Ed-Fi RFC 7 - Open API Spec

This page contains the full Open API spec listing for the proposed Ed-Fi RFC 7 - Core Student Data Management API.

Change History

- November 10, 2017: Initial Draft
- Core-Student-Data-Management-0-95.zip

API Specification

Core Student Data Management API

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "/",
  "apis": [
    {
      "path": "/calendarDates",
      "description": "This entity represents a day in the school calendar."
    },
    {
      "path": "/classPeriods",
      "description": "This entity represents the designation of a regularly scheduled series of class meetings
at designated times and days of the week."
    },
    {
      "path": "/cohorts",
      "description": "This entity represents any type of list of designated students for tracking, analysis, or
intervention."
    },
    {
      "path": "/courses",
      "description": "This educational entity represents the organization of subject matter and related
learning experiences provided for the instruction of students on a regular or systematic basis."
    },
    {
      "path": "/courseOfferings",
      "description": "This entity represents an entry in the course catalog of available courses offered by the
school during a session."
    },
    {
      "path": "/courseTranscripts",
      "description": "This entity is the final record of a student's performance in their courses at the end of
a semester or school year."
    },
    {
      "path": "/disciplineActions",
      "description": "This event entity represents actions taken by an education organization after a
disruptive event that is recorded as a discipline incident."
    },
    {
      "path": "/disciplineIncidents",
      "description": "This event entity represents an occurrence of an infraction ranging from a minor
heavioral problem that disrupts the orderly functioning of a school or classroom (such as tardiness) to a
criminal act that results in the involvement of a law enforcement official (such as robbery). A single event (e.
g., a fight) is one incident regardless of how many perpetrators or victims are involved. Discipline incidents
are events classified as warranting discipline action."
    },
    {
      "path": "/grades",
      "description": "This educational entity represents an overall score or assessment tied to a course over a
period of time (i.e., the grading period). Student grades are usually a compilation of marks and other scores."
    },
    {
      "path": "/gradingPeriods",
      "description": "This entity represents the time span for which grades are reported."
    },
    {
      "path": "/graduationPlans",
      "description": "This entity is a plan outlining the required credits, credits by subject,credits by
course, and other criteria required for graduation. A graduation plan may be one or more standard plans defined
by an education organization and/or individual plans for some or all students."
    },
    {
      "path": "/locations",
      "description": "This entity represents the physical space where students gather for a particular class
/section. The Location may be an indoor or outdoor area designated for the purpose of meeting the educational
needs of students."
    },
    {
      "path": "/parents",
      "description": "This entity represents a parent or guardian of a student, such as mother, father, or
caretaker."
    },
    {
      "path": "/programs",
      "description": "This entity represents any program designed to work in conjunction with, or as a
supplement to, the main academic program. Programs may provide instruction, training, services, or benefits
through federal, state, or local agencies. Programs may also include organized extracurricular activities for
students."
    },
    {
      "path": "/schools",
      "description": "This entity represents an educational organization that includes staff and students who
participate in classes and educational activity groups."
    },
    {
      "path": "/sections",
      "description": "This entity represents a setting in which organized instruction of course content is
provided, in-person or otherwise, to one or more students for a given period of time. A course offering may be
offered to more than one section."
    },
    {
      "path": "/sessions",
      "description": "This entity represents the prescribed span of time when an education institution is open,
instruction is provided and students are under the direction and guidance of teachers and/or education
institution administration. A session may be interrupted by one or more vacations."
    },
    {
      "path": "/staffs",
      "description": "This entity represents an individual who performs specified activities for any public or
private education institution or agency that provides instructional and/or support services to students or
staff at the early childhood level through high school completion. For example, this includes:      1. An \"
employee\" who performs services under the direction of the employing institution or agency is compensated for
such services by the employer and is eligible for employee benefits and wage or salary tax withholdings      2.
A \"contractor\" or \"consultant\" who performs services for an agreed upon fee or an employee of a management
service contracted to work on site      3. A \"volunteer\" who performs services on a voluntary and
uncompensated basis      4. An in-kind service provider      5. An independent contractor or businessperson
working at a school site."
    },
    {
      "path": "/staffCohortAssociations",
      "description": "This association indicates the Staff associated with a cohort of students."
    },
    {
      "path": "/staffEducationOrganizationAssignmentAssociations",
      "description": "This association indicates the education organization to which a staff member provides
services; also known as school of service."
    },
    {
      "path": "/staffSchoolAssociations",
      "description": "This association indicates the School(s) to which a staff member provides instructional
services."
    },
    {
      "path": "/staffSectionAssociations",
      "description": "This association indicates the class sections to which a staff member is assigned."
    },
    {
      "path": "/students",
      "description": "This entity represents an individual for whom instruction, services, and/or care are
provided in an early childhood, elementary, or secondary educational program under the jurisdiction of a
school, education agency or other institution or program. A student is a person who has been enrolled in a
school or other educational institution."
    },
    {
      "path": "/studentAcademicRecords",
      "description": "This educational entity represents the cumulative record of academic achievement for a
student."
    },
    {
      "path": "/studentCohortAssociations",
      "description": "This association represents the Cohort(s) for which a student is designated."
    },
    {
      "path": "/studentDisciplineIncidentAssociations",
      "description": "This association indicates those students who were victims, perpetrators, witnesses, and
reporters for a discipline incident."
    },
    {
      "path": "/studentEducationOrganizationAssociations",
      "description": "This association indicates any relationship between a student and an education
organization other than how the state views enrollment. Enrollment relationship semantics are covered by
StudentSchoolAssociation."
    },
    {
      "path": "/studentParentAssociations",
      "description": "This association relates students to their parents, guardians, or caretakers."
    },
    {
      "path": "/studentProgramAssociations",
      "description": "This association represents the Program(s) that a student participates in or is served
by."
    },
    {
      "path": "/studentSchoolAssociations",
      "description": "This association represents the School in which a student is enrolled. The semantics of
enrollment may differ slightly by state. Non-enrollment relationships between a student and an education
organization may be described using the StudentEducationOrganizationAssociation."
    },
    {
      "path": "/studentSchoolAttendanceEvents",
      "description": "This event entity represents the recording of whether a student is in attendance for a
school day."
    },
    {
      "path": "/studentSectionAssociations",
      "description": "This association indicates the course sections to which a student is assigned."
    },
    {
      "path": "/studentSectionAttendanceEvents",
      "description": "This event entity represents the recording of whether a student is in attendance for a
section."
    },
    {
      "path": "/studentSpecialEducationProgramAssociations",
      "description": "This association represents the special education program(s) that a student participates
in or receives services from. The association is an extension of the StudentProgramAssociation particular for
special education programs."
    },
    {
      "path": "/studentTitleIPartAProgramAssociations",
      "description": "This association represents the Title I Part A program(s) that a student participates in
or from which the Student receives services. The association is an extension of the StudentProgramAssociation
particular for Title I Part A programs."
    }
```

```
  ],
  "info": {
    "title": "Ed-Fi Core Student Data Management",
    "description": "The Core Student Data Management API Standards describes a RESTful API surface that covers
the core data domains typically managed by Student Information Systems in K&ndash;12 education. These standards
can be used to drive analysis of student performance, both alone and in combination with data from other
systems.",
    "termsOfServiceUrl": "",
    "contact": "",
    "license": "",
    "licenseUrl": "http://www.ed-fi.org/license/"
  },
  "schemes": [
    "http"
  ]
}
```

Resources

/calendarDates

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/calendarDates",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/calendarDates",
      "description": "This entity represents a day in the school calendar.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getCalendarDatesAll",
          "type": "array",
          "items": {
            "$ref": "calendarDate"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "calendarDate"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getCalendarDatesByExample",
          "type": "array",
          "items": {
            "$ref": "calendarDate"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "date",
              "description": "The month, day, and year of the CalendarDate.",
              "type": "date-time",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "calendarDate"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getCalendarDateByKey",
          "type": "calendarDate",
          "parameters": [
            {
              "paramType": "query",
              "name": "date",
              "description": "The month, day, and year of the CalendarDate.",
              "type": "date-time",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "calendarDate"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postCalendarDates",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "calendarDate",
              "description": "The JSON representation of the \"calendarDate\" resource to be created or
updated.",
              "type": "calendarDate",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
```

```
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/calendarDates/{id}",
      "description": "This entity represents a day in the school calendar.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getCalendarDatesById",
          "type": "calendarDate",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "calendarDate"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putCalendarDate",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "calendarDate",
              "description": "The JSON representation of the \"calendarDate\" resource to be updated.",
              "type": "calendarDate",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteCalendarDateById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "calendarDate": {
      "id": "calendarDate",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "schoolReference": {
          "type": "schoolReference",
          "required": true,
          "description": "A reference to the related School resource."
        },
        "date": {
          "type": "date-time",
          "required": true,
          "description": "The month, day, and year of the CalendarDate."
        },
        "calendarEvents": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of calendarDateCalendarEvents.  Additional description of the
date such as date classification and duration of the event.",
          "items": {
            "$ref": "calendarDateCalendarEvent"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "schoolReference": {
      "id": "schoolReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        }
      }
    },
    "calendarDateCalendarEvent": {
      "id": "calendarDateCalendarEvent",
      "properties": {
        "calendarEventDescriptor": {
          "type": "string",
          "required": true,
          "description": "The type of scheduled or unscheduled event for the day. For example:
Instructional day          Teacher only day          Holiday          Make-up day          Weather day
Student late arrival/early dismissal."
        },
        "eventDuration": {
          "type": "number",
          "required": true,
          "description": "The amount of time for the event as recognized by the school: 1 day = 1, 1/2 day =
0.5, 1/3 day = 0.33."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/classPeriods

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/classPeriods",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/classPeriods",
      "description": "This entity represents the designation of a regularly scheduled series of class meetings
at designated times and days of the week.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getClassPeriodsAll",
          "type": "array",
          "items": {
            "$ref": "classPeriod"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "classPeriod"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getClassPeriodsByExample",
          "type": "array",
          "items": {
            "$ref": "classPeriod"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "name",
              "description": "An indication of the portion of a typical daily session in which students receive
instruction in a specified subject (e.g., morning, sixth period, block period, or AB schedules).",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "classPeriod"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getClassPeriodByKey",
          "type": "classPeriod",
          "parameters": [
            {
              "paramType": "query",
              "name": "name",
              "description": "An indication of the portion of a typical daily session in which students receive
instruction in a specified subject (e.g., morning, sixth period, block period, or AB schedules).",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "classPeriod"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postClassPeriods",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "classPeriod",
              "description": "The JSON representation of the \"classPeriod\" resource to be created or
updated.",
              "type": "classPeriod",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/classPeriods/{id}",
      "description": "This entity represents the designation of a regularly scheduled series of class meetings
at designated times and days of the week.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getClassPeriodsById",
          "type": "classPeriod",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
```

```
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "classPeriod"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putClassPeriod",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "classPeriod",
              "description": "The JSON representation of the \"classPeriod\" resource to be updated.",
              "type": "classPeriod",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteClassPeriodById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource 
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource 
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to 
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body 
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was 
either not provided or is invalid.  The operation may succeed once authenication has been successfully 
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.  
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid 
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match 
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
```

```
  ],
  "models": {
    "classPeriod": {
      "id": "classPeriod",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "schoolReference": {
          "type": "schoolReference",
          "required": true,
          "description": "A reference to the related School resource."
        },
        "name": {
          "type": "string",
          "required": true,
          "description": "An indication of the portion of a typical daily session in which students receive
instruction in a specified subject (e.g., morning, sixth period, block period, or AB schedules)."
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "schoolReference": {
      "id": "schoolReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/cohorts

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/cohorts",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/cohorts",
      "description": "This entity represents any type of list of designated students for tracking, analysis, or
intervention.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getCohortsAll",
          "type": "array",
          "items": {
            "$ref": "cohort"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "cohort"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
```

```
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getCohortsByExample",
          "type": "array",
          "items": {
            "$ref": "cohort"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "identifier",
              "description": "The name or ID for the Cohort.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "cohort"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
```

```
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getCohortByKey",
          "type": "cohort",
          "parameters": [
            {
              "paramType": "query",
              "name": "identifier",
              "description": "The name or ID for the Cohort.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "cohort"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
```

```
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postCohorts",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "cohort",
              "description": "The JSON representation of the \"cohort\" resource to be created or updated.",
              "type": "cohort",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
```

```
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/cohorts/{id}",
      "description": "This entity represents any type of list of designated students for tracking, analysis, or
intervention.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getCohortsById",
          "type": "cohort",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "cohort"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putCohort",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "cohort",
              "description": "The JSON representation of the \"cohort\" resource to be updated.",
              "type": "cohort",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
```

```
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteCohortById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "cohort": {
      "id": "cohort",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "educationOrganizationReference": {
          "type": "educationOrganizationReference",
          "required": true,
          "description": "A reference to the related EducationOrganization resource."
        },
        "academicSubjectDescriptor": {
          "type": "string",
          "required": false,
          "description": "The academic subject associated with an academic intervention."
        },
        "description": {
          "type": "string",
          "required": true,
          "description": "The description of the Cohort and its purpose."
        },
        "identifier": {
          "type": "string",
          "required": true,
          "description": "The name or ID for the Cohort."
        },
        "scopeType": {
          "type": "string",
          "required": true,
          "description": "The scope of cohort (e.g., school, district, classroom)."
        },
        "type": {
          "type": "string",
          "required": true,
          "description": "The type of cohort (e.g., academic intervention, classroom breakout)."
        },
        "programs": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of cohortPrograms.  The (optional) program associated with
this Cohort (e.g., special education).",
          "items": {
            "$ref": "cohortProgram"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "educationOrganizationReference": {
      "id": "educationOrganizationReference",
      "properties": {
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        }
      }
    },
    "cohortProgram": {
      "id": "cohortProgram",
      "properties": {
        "programReference": {
          "type": "programReference",
          "required": true,
          "description": "A reference to the related Program resource."
        }
      }
    },
    "programReference": {
      "id": "programReference",
      "properties": {
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        },
        "type": {
          "type": "string",
          "required": true,
          "description": "The type of program."
        },
        "name": {
          "type": "string",
          "required": true,
          "description": "The formal name of the Program of instruction, training, services, or benefits
```

```
available through federal, state, or local agencies."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/courses

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/courses",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/courses",
      "description": "This educational entity represents the organization of subject matter and related
learning experiences provided for the instruction of students on a regular or systematic basis.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getCoursesAll",
          "type": "array",
          "items": {
            "$ref": "course"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "course"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getCoursesByExample",
          "type": "array",
          "items": {
            "$ref": "course"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "code",
              "description": "A unique alphanumeric code assigned to a course.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "course"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getCourseByKey",
          "type": "course",
          "parameters": [
            {
              "paramType": "query",
              "name": "code",
              "description": "A unique alphanumeric code assigned to a course.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "course"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postCourses",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "course",
              "description": "The JSON representation of the \"course\" resource to be created or updated.",
              "type": "course",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/courses/{id}",
      "description": "This educational entity represents the organization of subject matter and related
learning experiences provided for the instruction of students on a regular or systematic basis.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getCoursesById",
          "type": "course",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "course"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putCourse",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "course",
              "description": "The JSON representation of the \"course\" resource to be updated.",
              "type": "course",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteCourseById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "course": {
      "id": "course",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "educationOrganizationReference": {
          "type": "educationOrganizationReference",
          "required": true,
          "description": "A reference to the related EducationOrganization resource."
        },
        "academicSubjectDescriptor": {
          "type": "string",
          "required": true,
          "description": "The intended major subject area of the course."
        },
        "careerPathwayType": {
          "type": "string",
          "required": false,
          "description": "Indicates the career cluster or pathway the course is associated with as part of a
CTE curriculum."
        },
        "code": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a course."
        },
        "definedByType": {
          "type": "string",
          "required": false,
          "description": "Specifies whether the course was defined by the SEA, LEA, School, or national
organization."
        },
        "description": {
          "type": "string",
          "required": false,
          "description": "A description of the content standards and goals covered in the course. Reference may
be made to state or national content standards."
        },
        "gpaApplicabilityType": {
          "type": "string",
          "required": false,
          "description": "An indicator of whether or not the course being described is included in the
computation of the student's Grade Point Average, and if so, if it is weighted differently from regular
courses."
        },
        "title": {
          "type": "string",
          "required": true,
          "description": "The descriptive name given to a course of study offered in a school or other
institution or organization. In departmentalized classes at the elementary, secondary, and postsecondary levels
(and for staff development activities), this refers to the name by which a course is identified (e.g., American
History, English III). For elementary and other non-departmentalized classes, it refers to any portion of the
instruction for which a grade or report is assigned (e.g., reading, composition, spelling, and language arts)."
```

```
        },
        "dateCourseAdopted": {
          "type": "date-time",
          "required": false,
          "description": "Date the course was adopted by the education agency."
        },
        "highSchoolCourseRequirement": {
          "type": "boolean",
          "required": false,
          "description": "An indication that this course may satisfy high school graduation requirements in the
course's subject area."
        },
        "maximumAvailableCreditConversion": {
          "type": "number",
          "required": false,
          "description": "Conversion factor that when multiplied by the number of credits is equivalent to
Carnegie units."
        },
        "maximumAvailableCredits": {
          "type": "number",
          "required": false,
          "description": "The value of credits or units of value awarded for the completion of a course."
        },
        "maximumAvailableCreditType": {
          "type": "string",
          "required": false,
          "description": "The type of credits or units of value awarded for the completion of a course."
        },
        "minimumAvailableCreditConversion": {
          "type": "number",
          "required": false,
          "description": "Conversion factor that when multiplied by the number of credits is equivalent to
Carnegie units."
        },
        "minimumAvailableCredits": {
          "type": "number",
          "required": false,
          "description": "The value of credits or units of value awarded for the completion of a course."
        },
        "minimumAvailableCreditType": {
          "type": "string",
          "required": false,
          "description": "The type of credits or units of value awarded for the completion of a course."
        },
        "numberOfParts": {
          "type": "integer",
          "required": true,
          "description": "The number of parts identified for a course."
        },
        "timeRequiredForCompletion": {
          "type": "integer",
          "required": false,
          "description": "The actual or estimated number of clock minutes required for class completion.  This
number is especially important for career and technical education classes and may represent (in minutes) the
clock hour requirement of the class."
        },
        "competencyLevels": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of courseCompetencyLevels.  The competency levels defined to
rate the student for the course.",
          "items": {
            "$ref": "courseCompetencyLevel"
          }
        },
        "identificationCodes": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of courseIdentificationCodes.  The code that identifies the
organization of subject matter and related learning experiences provided for the instruction of students.",
          "items": {
            "$ref": "courseIdentificationCode"
          }
        },
        "learningObjectives": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of courseLearningObjectives.  Learning Objectives to be
mastered in the course.",
          "items": {
            "$ref": "courseLearningObjective"
          }
        },
        "learningStandards": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of courseLearningStandards.  Learning Standard(s) to be
taught by the course.",
          "items": {
            "$ref": "courseLearningStandard"
          }
        },
        "levelCharacteristics": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of courseLevelCharacteristics.  The type of specific program
or designation with which the course is associated (e.g., AP, IB, Dual Credit, CTE).",
          "items": {
            "$ref": "courseLevelCharacteristic"
          }
        },
        "offeredGradeLevels": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of courseOfferedGradeLevels.  The grade levels in which the
course is offered.",
          "items": {
            "$ref": "courseOfferedGradeLevel"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "educationOrganizationReference": {
      "id": "educationOrganizationReference",
      "properties": {
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        }
      }
    },
    "courseCompetencyLevel": {
      "id": "courseCompetencyLevel",
      "properties": {
        "competencyLevelDescriptor": {
          "type": "string",
          "required": true,
          "description": "The competency levels defined to rate the student for the course."
        }
      }
    },
    "courseIdentificationCode": {
      "id": "courseIdentificationCode",
      "properties": {
        "courseIdentificationSystemDescriptor": {
          "type": "string",
          "required": true,
          "description": "A system that is used to identify the organization of subject matter and related
learning experiences provided for the instruction of students."
        },
        "identificationCode": {
          "type": "string",
          "required": true,
          "description": "A unique number or alphanumeric code assigned to a course by a school, school system,
state, or other agency or entity. For multi-part course codes, concatenate the parts separated by a \"/\". For
example, consider the following SCED code-    subject = 20 Math    course = 272 Geometry    level = G
General    credits = 1.00   course sequence 1 of 1- would be entered as 20/272/G/1.00/1 of 1."
        },
        "assigningOrganizationIdentificationCode": {
          "type": "string",
          "required": false,
          "description": "The organization code or name assigning the staff Identification Code."
        }
      }
    },
    "courseLearningObjective": {
      "id": "courseLearningObjective",
      "properties": {
        "learningObjectiveReference": {
          "type": "learningObjectiveReference",
          "required": true,
          "description": "A reference to the related LearningObjective resource."
        }
      }
    },
    "courseLearningStandard": {
      "id": "courseLearningStandard",
      "properties": {
        "learningStandardReference": {
          "type": "learningStandardReference",
          "required": true,
          "description": "A reference to the related LearningStandard resource."
        }
      }
    },
    "courseLevelCharacteristic": {
      "id": "courseLevelCharacteristic",
      "properties": {
        "type": {
          "type": "string",
          "required": true,
          "description": "The type of specific program or designation with which the course is associated (e.
g., AP, IB, Dual Credit, CTE)."
        }
      }
    },
    "courseOfferedGradeLevel": {
      "id": "courseOfferedGradeLevel",
      "properties": {
        "gradeLevelDescriptor": {
          "type": "string",
          "required": true,
          "description": "The grade levels in which the course is offered."
        }
      }
    },
    "learningObjectiveReference": {
      "id": "learningObjectiveReference",
      "properties": {
        "objective": {
          "type": "string",
          "required": true,
          "description": "The designated title of the LearningObjective."
        },
        "academicSubjectDescriptor": {
          "type": "string",
          "required": true,
          "description": "The description of the content or subject area (e.g., arts, mathematics, reading,
stenography, or a foreign language) of an assessment."
        },
        "objectiveGradeLevelDescriptor": {
          "type": "string",
          "required": true,
          "description": "The grade level for which the LearningObjective is targeted."
        }
      }
    },
    "learningStandardReference": {
      "id": "learningStandardReference",
      "properties": {
        "learningStandardId": {
          "type": "string",
          "required": true,
          "description": "The identifier for the specific learning standard (e.g., 111.15.3.1.A)."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/courseOfferings

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/courseOfferings",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/courseOfferings",
      "description": "This entity represents an entry in the course catalog of available courses offered by the
school during a session.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getCourseOfferingsAll",
          "type": "array",
          "items": {
            "$ref": "courseOffering"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "courseOffering"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getCourseOfferingsByExample",
          "type": "array",
          "items": {
            "$ref": "courseOffering"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "localCourseCode",
              "description": "The local code assigned by the School that identifies the course offering
provided for the instruction of students.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolYear",
              "description": "The identifier for the school year.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "termDescriptor",
              "description": "The term for the Session during the school year.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "courseOffering"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getCourseOfferingByKey",
          "type": "courseOffering",
          "parameters": [
            {
              "paramType": "query",
              "name": "localCourseCode",
              "description": "The local code assigned by the School that identifies the course offering
provided for the instruction of students.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolYear",
              "description": "The identifier for the school year.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "termDescriptor",
              "description": "The term for the Session during the school year.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "courseOffering"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postCourseOfferings",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "courseOffering",
              "description": "The JSON representation of the \"courseOffering\" resource to be created or
updated.",
              "type": "courseOffering",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/courseOfferings/{id}",
      "description": "This entity represents an entry in the course catalog of available courses offered by the
school during a session.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getCourseOfferingsById",
          "type": "courseOffering",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "courseOffering"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putCourseOffering",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "courseOffering",
              "description": "The JSON representation of the \"courseOffering\" resource to be updated.",
              "type": "courseOffering",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
```

```
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteCourseOfferingById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "courseOffering": {
      "id": "courseOffering",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "courseReference": {
          "type": "courseReference",
          "required": true,
          "description": "A reference to the related Course resource."
        },
        "schoolReference": {
          "type": "schoolReference",
          "required": true,
          "description": "A reference to the related School resource."
        },
        "sessionReference": {
          "type": "sessionReference",
          "required": true,
          "description": "A reference to the related Session resource."
        },
        "instructionalTimePlanned": {
          "type": "integer",
          "required": false,
          "description": "The planned total number of clock minutes of instruction for this course offering.
Generally, this should be at least as many minutes as is required for completion by the related state- or
district-defined course."
        },
        "localCourseCode": {
          "type": "string",
          "required": true,
          "description": "The local code assigned by the School that identifies the course offering provided
for the instruction of students."
        },
        "localCourseTitle": {
          "type": "string",
          "required": true,
          "description": "The descriptive name given to a course of study offered in the school, if different
from the CourseTitle."
        },
        "curriculumUseds": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of courseOfferingCurriculumUseds.  The type of curriculum
used in an early learning classroom or group.",
          "items": {
            "$ref": "courseOfferingCurriculumUsed"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "courseReference": {
      "id": "courseReference",
      "properties": {
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        },
        "code": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a course."
        }
      }
    },
    "schoolReference": {
      "id": "schoolReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        }
      }
    },
    "sessionReference": {
      "id": "sessionReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        },
        "schoolYear": {
          "type": "integer",
          "required": true,
          "description": "The identifier for the school year."
        },
        "termDescriptor": {
          "type": "string",
          "required": true,
          "description": "The term for the Session during the school year."
        }
      }
    },
    "courseOfferingCurriculumUsed": {
      "id": "courseOfferingCurriculumUsed",
      "properties": {
        "curriculumUsedType": {
          "type": "string",
          "required": true,
          "description": "The type of curriculum used in an early learning classroom or group."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/courseTranscripts

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/courseTranscripts",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/courseTranscripts",
      "description": "This entity is the final record of a student's performance in their courses at the end of
a semester or school year.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getCourseTranscriptsAll",
          "type": "array",
          "items": {
            "$ref": "courseTranscript"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "courseTranscript"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getCourseTranscriptsByExample",
          "type": "array",
          "items": {
            "$ref": "courseTranscript"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "courseAttemptResultType",
              "description": "The result from the student's attempt to take the course, for example:
Pass          Fail          Incomplete          Withdrawn.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "courseCode",
              "description": "A unique alphanumeric code assigned to a course.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "courseEducationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolYear",
              "description": "The identifier for the school year.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "termDescriptor",
              "description": "The term for the session during the school year.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "courseTranscript"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getCourseTranscriptByKey",
          "type": "courseTranscript",
          "parameters": [
            {
              "paramType": "query",
              "name": "courseAttemptResultType",
              "description": "The result from the student's attempt to take the course, for example:
Pass          Fail          Incomplete          Withdrawn.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "courseCode",
              "description": "A unique alphanumeric code assigned to a course.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "courseEducationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolYear",
              "description": "The identifier for the school year.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "termDescriptor",
              "description": "The term for the session during the school year.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "courseTranscript"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was 
either not provided or is invalid.  The operation may succeed once authenication has been successfully 
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postCourseTranscripts",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "courseTranscript",
              "description": "The JSON representation of the \"courseTranscript\" resource to be created or 
updated.",
              "type": "courseTranscript",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is 
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\" 
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values 
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the 
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not 
yet completed due to current system load. Processing may still fail due to violation of referential integrity 
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of 
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body 
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was 
either not provided or is invalid.  The operation may succeed once authenication has been successfully 
completed."
```

```
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/courseTranscripts/{id}",
      "description": "This entity is the final record of a student's performance in their courses at the end of
a semester or school year.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getCourseTranscriptsById",
          "type": "courseTranscript",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "courseTranscript"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
```

```
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putCourseTranscript",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "courseTranscript",
              "description": "The JSON representation of the \"courseTranscript\" resource to be updated.",
              "type": "courseTranscript",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
```

```
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteCourseTranscriptById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "courseTranscript": {
      "id": "courseTranscript",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "courseReference": {
          "type": "courseReference",
          "required": true,
          "description": "A reference to the related Course resource."
        },
        "schoolReference": {
          "type": "schoolReference",
          "required": false,
          "description": "A reference to the related School resource."
        },
        "studentAcademicRecordReference": {
          "type": "studentAcademicRecordReference",
          "required": true,
          "description": "A reference to the related StudentAcademicRecord resource."
        },
        "alternativeCourseCode": {
          "type": "string",
          "required": false,
          "description": "The local code assigned by the school that identifies the course offering, the code
from an external educational organization, or other alternate course code."
        },
        "alternativeCourseTitle": {
          "type": "string",
          "required": false,
          "description": "The descriptive name given to a course of study offered in the school, if different
from the CourseTitle."
        },
        "attemptedCreditConversion": {
          "type": "number",
          "required": false,
          "description": "Conversion factor that when multiplied by the number of credits is equivalent to
Carnegie units."
        },
        "attemptedCredits": {
          "type": "number",
          "required": true,
          "description": "The value of credits or units of value awarded for the completion of a course."
        },
        "attemptedCreditType": {
          "type": "string",
          "required": false,
          "description": "The type of credits or units of value awarded for the completion of a course."
        },
        "courseAttemptResultType": {
          "type": "string",
          "required": true,
          "description": "The result from the student's attempt to take the course, for example:
Pass          Fail          Incomplete          Withdrawn."
        },
        "courseRepeatCodeType": {
          "type": "string",
          "required": false,
          "description": "Indicates that an academic course has been repeated by a student and how that repeat
is to be computed in the student's academic grade average."
        },
        "courseTitle": {
          "type": "string",
          "required": false,
          "description": "The descriptive name given to a course of study offered in a school or other
institution or organization. In departmentalized classes at the elementary, secondary, and postsecondary levels
(and for staff development activities), this refers to the name by which a course is identified (e.g., American
History, English III). For elementary and other non-departmentalized classes, it refers to any portion of the
instruction for which a grade or report is assigned (e.g., reading, composition, spelling, language arts)."
        },
        "earnedCreditConversion": {
          "type": "number",
          "required": false,
          "description": "Conversion factor that when multiplied by the number of credits is equivalent to
Carnegie units."
        },
        "earnedCredits": {
          "type": "number",
          "required": true,
          "description": "The value of credits or units of value awarded for the completion of a course."
        },
        "earnedCreditType": {
          "type": "string",
          "required": false,
          "description": "The type of credits or units of value awarded for the completion of a course."
        },
        "finalLetterGradeEarned": {
          "type": "string",
          "required": true,
          "description": "The final indicator of student performance in a class as submitted by the instructor."
        },
        "finalNumericGradeEarned": {
          "type": "number",
          "required": true,
          "description": "The final indicator of student performance in a class as submitted by the instructor."
        },
        "methodCreditEarnedType": {
          "type": "string",
          "required": false,
          "description": "The method the credits were earned (e.g., Classroom, Examination, Transfer)."
        },
        "whenTakenGradeLevelDescriptor": {
          "type": "string",
          "required": false,
          "description": "Student's grade level at time of course."
        },
        "earnedAdditionalCredits": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of courseTranscriptEarnedAdditionalCredits.  The number of
additional credits a student attempted and could earn for successfully completing a given course (e.g., dual
credit, AP, IB).",
          "items": {
            "$ref": "courseTranscriptEarnedAdditionalCredits"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "courseReference": {
      "id": "courseReference",
      "properties": {
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        },
        "code": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a course."
        }
      }
    },
    "schoolReference": {
      "id": "schoolReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        }
      }
    },
    "studentAcademicRecordReference": {
      "id": "studentAcademicRecordReference",
      "properties": {
        "studentUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a student."
        },
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        },
        "schoolYear": {
          "type": "integer",
          "required": true,
          "description": "The identifier for the school year."
        },
        "termDescriptor": {
          "type": "string",
          "required": true,
          "description": "The term for the session during the school year."
        }
      }
    },
    "courseTranscriptEarnedAdditionalCredits": {
      "id": "courseTranscriptEarnedAdditionalCredits",
      "properties": {
        "additionalCreditType": {
          "type": "string",
          "required": true,
          "description": "The type of credits or units of value awarded for the completion of a course."
        },
        "credits": {
          "type": "number",
          "required": true,
          "description": "The value of credits or units of value awarded for the completion of a course"
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/disciplineActions

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/disciplineActions",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/disciplineActions",
      "description": "This event entity represents actions taken by an education organization after a
disruptive event that is recorded as a discipline incident.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getDisciplineActionsAll",
          "type": "array",
          "items": {
            "$ref": "disciplineAction"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "disciplineAction"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getDisciplineActionsByExample",
          "type": "array",
          "items": {
            "$ref": "disciplineAction"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "identifier",
              "description": "Identifier assigned by the education organization to the DisciplineAction.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "disciplineDate",
              "description": "The date of the DisciplineAction.",
              "type": "date-time",
              "required": false
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "disciplineAction"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
```

```
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getDisciplineActionByKey",
          "type": "disciplineAction",
          "parameters": [
            {
              "paramType": "query",
              "name": "identifier",
              "description": "Identifier assigned by the education organization to the DisciplineAction.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "disciplineDate",
              "description": "The date of the DisciplineAction.",
              "type": "date-time",
              "required": true
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "disciplineAction"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
```

```
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postDisciplineActions",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "disciplineAction",
              "description": "The JSON representation of the \"disciplineAction\" resource to be created or
updated.",
              "type": "disciplineAction",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/disciplineActions/{id}",
      "description": "This event entity represents actions taken by an education organization after a
disruptive event that is recorded as a discipline incident.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getDisciplineActionsById",
          "type": "disciplineAction",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "disciplineAction"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putDisciplineAction",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "disciplineAction",
              "description": "The JSON representation of the \"disciplineAction\" resource to be updated.",
              "type": "disciplineAction",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteDisciplineActionById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to 
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body 
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was 
either not provided or is invalid.  The operation may succeed once authenication has been successfully 
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.  
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid 
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match 
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "disciplineAction": {
      "id": "disciplineAction",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "responsibilitySchoolReference": {
          "type": "schoolReference",
          "required": true,
          "description": "A reference to the related School resource."
        },
        "assignmentSchoolReference": {
          "type": "schoolReference",
          "required": false,
          "description": "A reference to the related School resource."
        },
        "studentReference": {
          "type": "studentReference",
          "required": true,
          "description": "A reference to the related Student resource."
        },
        "actualDisciplineActionLength": {
          "type": "integer",
          "required": true,
          "description": "Indicates the actual length in school days of a student's disciplinary assignment."
        },
        "identifier": {
          "type": "string",
          "required": true,
          "description": "Identifier assigned by the education organization to the DisciplineAction."
        },
        "length": {
          "type": "integer",
          "required": false,
          "description": "The length of time in school days for the DisciplineAction (e.g. removal, detention),
if applicable."
        },
        "lengthDifferenceReasonType": {
          "type": "string",
          "required": false,
          "description": "Indicates the reason for the difference, if any, between the official and actual
lengths of a student's disciplinary assignment."
        },
        "disciplineDate": {
          "type": "date-time",
          "required": true,
          "description": "The date of the DisciplineAction."
        },
        "relatedToZeroTolerancePolicy": {
          "type": "boolean",
          "required": false,
          "description": "An indication of whether or not this disciplinary action taken against a student was
imposed as a consequence of state or local zero tolerance policies."
        },
        "disciplines": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of disciplineActionDisciplines.  Type of action, such as
removal from the classroom, used to discipline the student involved as a perpetrator in a discipline incident.",
          "items": {
            "$ref": "disciplineActionDiscipline"
          }
        },
        "disciplineIncidents": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of disciplineActionDisciplineIncidents.  Reference to the
DisciplineIncident associated with the DisciplineAction.",
          "items": {
            "$ref": "disciplineActionDisciplineIncident"
          }
        },
        "staffs": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of disciplineActionStaffs.  The staff responsible for
enforcing the DisciplineAction.",
          "items": {
            "$ref": "disciplineActionStaff"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "schoolReference": {
      "id": "schoolReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        }
      }
    },
    "studentReference": {
      "id": "studentReference",
      "properties": {
        "studentUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a student."
        }
      }
    },
    "disciplineActionDiscipline": {
      "id": "disciplineActionDiscipline",
      "properties": {
        "disciplineDescriptor": {
          "type": "string",
          "required": true,
          "description": "Type of action, such as removal from the classroom, used to discipline the student
involved as a perpetrator in a discipline incident."
        }
      }
    },
    "disciplineActionDisciplineIncident": {
      "id": "disciplineActionDisciplineIncident",
      "properties": {
        "disciplineIncidentReference": {
          "type": "disciplineIncidentReference",
          "required": true,
          "description": "A reference to the related DisciplineIncident resource."
        }
      }
    },
    "disciplineActionStaff": {
      "id": "disciplineActionStaff",
      "properties": {
        "staffReference": {
          "type": "staffReference",
          "required": true,
          "description": "A reference to the related Staff resource."
        }
      }
    },
    "disciplineIncidentReference": {
      "id": "disciplineIncidentReference",
      "properties": {
        "incidentIdentifier": {
          "type": "string",
          "required": true,
          "description": "A locally assigned unique identifier (within the school or school district) to
identify each specific DisciplineIncident or occurrence. The same identifier should be used to document the
entire DisciplineIncident even if it included multiple offenses and multiple offenders."
        },
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        }
      }
    },
    "staffReference": {
      "id": "staffReference",
      "properties": {
        "staffUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a staff."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/disciplineIncidents

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/disciplineIncidents",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/disciplineIncidents",
      "description": "This event entity represents an occurrence of an infraction ranging from a minor 
heavioral problem that disrupts the orderly functioning of a school or classroom (such as tardiness) to a 
criminal act that results in the involvement of a law enforcement official (such as robbery). A single event (e.
g., a fight) is one incident regardless of how many perpetrators or victims are involved. Discipline incidents 
are events classified as warranting discipline action.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getDisciplineIncidentsAll",
          "type": "array",
          "items": {
            "$ref": "disciplineIncident"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "disciplineIncident"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getDisciplineIncidentsByExample",
          "type": "array",
          "items": {
            "$ref": "disciplineIncident"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "incidentIdentifier",
              "description": "A locally assigned unique identifier (within the school or school district) to
identify each specific DisciplineIncident or occurrence. The same identifier should be used to document the
entire DisciplineIncident even if it included multiple offenses and multiple offenders.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "disciplineIncident"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getDisciplineIncidentByKey",
          "type": "disciplineIncident",
          "parameters": [
            {
              "paramType": "query",
              "name": "incidentIdentifier",
              "description": "A locally assigned unique identifier (within the school or school district) to
```

```
identify each specific DisciplineIncident or occurrence. The same identifier should be used to document the
entire DisciplineIncident even if it included multiple offenses and multiple offenders.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "disciplineIncident"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postDisciplineIncidents",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "disciplineIncident",
              "description": "The JSON representation of the \"disciplineIncident\" resource to be created or
updated.",
              "type": "disciplineIncident",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/disciplineIncidents/{id}",
      "description": "This event entity represents an occurrence of an infraction ranging from a minor 
heavioral problem that disrupts the orderly functioning of a school or classroom (such as tardiness) to a 
criminal act that results in the involvement of a law enforcement official (such as robbery). A single event (e.
g., a fight) is one incident regardless of how many perpetrators or victims are involved. Discipline incidents 
are events classified as warranting discipline action.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getDisciplineIncidentsById",
          "type": "disciplineIncident",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary 
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\" 
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "disciplineIncident"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value 
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body 
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was 
either not provided or is invalid.  The operation may succeed once authenication has been successfully 
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.  
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putDisciplineIncident",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "disciplineIncident",
              "description": "The JSON representation of the \"disciplineIncident\" resource to be updated.",
              "type": "disciplineIncident",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteDisciplineIncidentById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
```

```
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "disciplineIncident": {
      "id": "disciplineIncident",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "schoolReference": {
          "type": "schoolReference",
          "required": true,
          "description": "A reference to the related School resource."
        },
        "staffReference": {
          "type": "staffReference",
          "required": false,
          "description": "A reference to the related Staff resource."
        },
        "caseNumber": {
          "type": "string",
          "required": false,
          "description": "The case number assigned to the DisciplineIncident by law enforcement or other
organization."
        },
        "incidentCost": {
          "type": "number",
          "required": false,
          "description": "The value of any quantifiable monetary loss directly resulting from the
DisciplineIncident. Examples include the value of repairs necessitated by vandalism of a school facility, or
the value of personnel resources used for repairs or consumed by the incident."
        },
        "incidentDate": {
          "type": "date-time",
          "required": true,
          "description": "The month, day, and year on which the DisciplineIncident occurred."
        },
        "incidentDescription": {
          "type": "string",
          "required": false,
          "description": "The description for an incident."
        },
        "incidentIdentifier": {
          "type": "string",
          "required": true,
          "description": "A locally assigned unique identifier (within the school or school district) to
```

```
identify each specific DisciplineIncident or occurrence. The same identifier should be used to document the
entire DisciplineIncident even if it included multiple offenses and multiple offenders."
        },
        "incidentLocationType": {
          "type": "string",
          "required": true,
          "description": "Identifies where the DisciplineIncident occurred and whether or not it occurred on
school, for example:          On school          Administrative offices area          Cafeteria area
Classroom          Hallway or stairs          ..."
        },
        "incidentTime": {
          "type": "string",
          "required": false,
          "description": "An indication of the time of day the incident took place."
        },
        "reportedToLawEnforcement": {
          "type": "boolean",
          "required": false,
          "description": "Indicator of whether the incident was reported to law enforcement."
        },
        "reporterDescriptionDescriptor": {
          "type": "string",
          "required": true,
          "description": "Information on the type of individual who reported the DisciplineIncident. When known
and/or if useful, use a more specific option code (e.g., \"Counselor\" rather than \"Professional Staff\"); for
example:          Student          Parent/guardian          Law enforcement officer          Nonschool
personnel          Representative of visiting school          ..."
        },
        "reporterName": {
          "type": "string",
          "required": true,
          "description": "Identifies the reporter of the DisciplineIncident by name."
        },
        "behaviors": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of disciplineIncidentBehaviors.  Describes behavior by
category and provides a detailed description.",
          "items": {
            "$ref": "disciplineIncidentBehavior"
          }
        },
        "weapons": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of disciplineIncidentWeapons.  Identifies the type of weapon
used during an incident. The Federal Gun-Free Schools Act requires states to report the number of students
expelled for bringing firearms to school by type of firearm.",
          "items": {
            "$ref": "disciplineIncidentWeapon"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "schoolReference": {
      "id": "schoolReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        }
      }
    },
    "staffReference": {
      "id": "staffReference",
      "properties": {
        "staffUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a staff."
        }
      }
    },
    "disciplineIncidentBehavior": {
      "id": "disciplineIncidentBehavior",
      "properties": {
        "behaviorDescriptor": {
          "type": "string",
          "required": true,
          "description": "Describes behavior by category and provides a detailed description."
        },
        "behaviorDetailedDescription": {
          "type": "string",
          "required": false,
          "description": "Specifies a more granular level of detail of a behavior involved in the incident."
        }
      }
    },
    "disciplineIncidentWeapon": {
      "id": "disciplineIncidentWeapon",
      "properties": {
        "weaponDescriptor": {
          "type": "string",
          "required": true,
          "description": "Identifies the type of weapon used during an incident. The Federal Gun-Free Schools
Act requires states to report the number of students expelled for bringing firearms to school by type of
firearm."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/grades

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/grades",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/grades",
      "description": "This educational entity represents an overall score or assessment tied to a course over a
period of time (i.e., the grading period). Student grades are usually a compilation of marks and other scores.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getGradesAll",
          "type": "array",
          "items": {
            "$ref": "grade"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "grade"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getGradesByExample",
          "type": "array",
          "items": {
            "$ref": "grade"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "beginDate",
              "description": "Month, day, and year of the Student's entry or assignment to the Section.",
              "type": "date-time",
              "required": false
            },
            {
              "paramType": "query",
              "name": "classPeriodName",
              "description": "An indication of the portion of a typical daily session in which students receive
instruction in a specified subject (e.g., morning, sixth period, block period, or AB schedules).",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "classroomIdentificationCode",
              "description": "A unique number or alphanumeric code assigned to a room by a school, school
system, state, or other agency or entity.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "type",
              "description": "The type of grade reported (e.g., Exam, Final, Grading Period).",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "gradingPeriodBeginDate",
              "description": "Month, day, and year of the first day of the GradingPeriod.",
              "type": "date-time",
              "required": false
            },
            {
              "paramType": "query",
              "name": "gradingPeriodDescriptor",
              "description": "The name of the period for which grades are reported.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "localCourseCode",
              "description": "The local code assigned by the School that identifies the course offering
provided for the instruction of students.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolYear",
              "description": "The identifier for the school year.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "sequenceOfCourse",
              "description": "When a section is part of a sequence of parts for a course, the number of the
sequence. If the course has only one part, the value of this section attribute should be 1.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "termDescriptor",
              "description": "The term for the Session during the school year.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "uniqueSectionCode",
              "description": "A unique identifier for the Section that is defined by the classroom, the
subjects taught, and the instructors who are assigned.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "grade"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getGradeByKey",
          "type": "grade",
          "parameters": [
            {
              "paramType": "query",
              "name": "beginDate",
              "description": "Month, day, and year of the Student's entry or assignment to the Section.",
              "type": "date-time",
              "required": true
            },
            {
              "paramType": "query",
              "name": "classPeriodName",
              "description": "An indication of the portion of a typical daily session in which students receive
instruction in a specified subject (e.g., morning, sixth period, block period, or AB schedules).",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "classroomIdentificationCode",
              "description": "A unique number or alphanumeric code assigned to a room by a school, school
system, state, or other agency or entity.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "type",
              "description": "The type of grade reported (e.g., Exam, Final, Grading Period).",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "gradingPeriodBeginDate",
              "description": "Month, day, and year of the first day of the GradingPeriod.",
              "type": "date-time",
              "required": true
            },
            {
              "paramType": "query",
              "name": "gradingPeriodDescriptor",
              "description": "The name of the period for which grades are reported.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "localCourseCode",
              "description": "The local code assigned by the School that identifies the course offering
provided for the instruction of students.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolYear",
              "description": "The identifier for the school year.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "sequenceOfCourse",
              "description": "When a section is part of a sequence of parts for a course, the number of the
sequence. If the course has only one part, the value of this section attribute should be 1.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "termDescriptor",
              "description": "The term for the Session during the school year.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "uniqueSectionCode",
              "description": "A unique identifier for the Section that is defined by the classroom, the
subjects taught, and the instructors who are assigned.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
```

```
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "grade"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postGrades",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "grade",
              "description": "The JSON representation of the \"grade\" resource to be created or updated.",
              "type": "grade",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/grades/{id}",
      "description": "This educational entity represents an overall score or assessment tied to a course over a
period of time (i.e., the grading period). Student grades are usually a compilation of marks and other scores.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getGradesById",
          "type": "grade",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "grade"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putGrade",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "grade",
              "description": "The JSON representation of the \"grade\" resource to be updated.",
              "type": "grade",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteGradeById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "grade": {
      "id": "grade",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "gradingPeriodReference": {
          "type": "gradingPeriodReference",
          "required": true,
          "description": "A reference to the related GradingPeriod resource."
        },
        "studentSectionAssociationReference": {
          "type": "studentSectionAssociationReference",
          "required": true,
          "description": "A reference to the related StudentSectionAssociation resource."
        },
        "diagnosticStatement": {
          "type": "string",
          "required": false,
          "description": "A statement provided by the teacher that provides information in addition to the
grade or assessment score."
        },
        "type": {
          "type": "string",
          "required": true,
          "description": "The type of grade reported (e.g., Exam, Final, Grading Period)."
        },
        "letterGradeEarned": {
          "type": "string",
          "required": true,
          "description": "A final or interim (grading period) indicator of student performance in a class as
submitted by the instructor."
        },
        "numericGradeEarned": {
          "type": "number",
          "required": true,
          "description": "A final or interim (grading period) indicator of student performance in a class as
submitted by the instructor."
        },
        "performanceBaseConversionType": {
          "type": "string",
          "required": false,
          "description": "A conversion of the level to a standard set of performance levels."
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "gradingPeriodReference": {
      "id": "gradingPeriodReference",
      "properties": {
        "descriptor": {
          "type": "string",
          "required": true,
          "description": "The name of the period for which grades are reported."
        },
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        },
        "beginDate": {
          "type": "date-time",
          "required": true,
          "description": "Month, day, and year of the first day of the GradingPeriod."
        }
      }
    },
    "studentSectionAssociationReference": {
      "id": "studentSectionAssociationReference",
      "properties": {
        "studentUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a student."
        },
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        },
        "classPeriodName": {
          "type": "string",
          "required": true,
          "description": "An indication of the portion of a typical daily session in which students receive 
instruction in a specified subject (e.g., morning, sixth period, block period, or AB schedules)."
        },
        "classroomIdentificationCode": {
          "type": "string",
          "required": true,
          "description": "A unique number or alphanumeric code assigned to a room by a school, school system, 
state, or other agency or entity."
        },
        "localCourseCode": {
          "type": "string",
          "required": true,
          "description": "The local code assigned by the School that identifies the course offering provided 
for the instruction of students."
        },
        "uniqueSectionCode": {
          "type": "string",
          "required": true,
          "description": "A unique identifier for the Section that is defined by the classroom, the subjects 
taught, and the instructors who are assigned."
        },
        "sequenceOfCourse": {
          "type": "integer",
          "required": true,
          "description": "When a section is part of a sequence of parts for a course, the number of the 
sequence. If the course has only one part, the value of this section attribute should be 1."
        },
        "schoolYear": {
          "type": "integer",
          "required": true,
          "description": "The identifier for the school year."
        },
        "termDescriptor": {
          "type": "string",
          "required": true,
          "description": "The term for the Session during the school year."
        },
        "beginDate": {
          "type": "date-time",
          "required": true,
          "description": "Month, day, and year of the Student's entry or assignment to the Section."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/gradingPeriods

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/gradingPeriods",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/gradingPeriods",
      "description": "This entity represents the time span for which grades are reported.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getGradingPeriodsAll",
          "type": "array",
          "items": {
            "$ref": "gradingPeriod"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "gradingPeriod"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getGradingPeriodsByExample",
          "type": "array",
          "items": {
            "$ref": "gradingPeriod"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "beginDate",
              "description": "Month, day, and year of the first day of the GradingPeriod.",
              "type": "date-time",
              "required": false
            },
            {
              "paramType": "query",
              "name": "descriptor",
              "description": "The name of the period for which grades are reported.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "gradingPeriod"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getGradingPeriodByKey",
          "type": "gradingPeriod",
          "parameters": [
            {
              "paramType": "query",
              "name": "beginDate",
              "description": "Month, day, and year of the first day of the GradingPeriod.",
              "type": "date-time",
              "required": true
            },
            {
              "paramType": "query",
              "name": "descriptor",
              "description": "The name of the period for which grades are reported.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "gradingPeriod"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postGradingPeriods",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "gradingPeriod",
              "description": "The JSON representation of the \"gradingPeriod\" resource to be created or
updated.",
              "type": "gradingPeriod",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/gradingPeriods/{id}",
      "description": "This entity represents the time span for which grades are reported.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getGradingPeriodsById",
          "type": "gradingPeriod",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "gradingPeriod"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putGradingPeriod",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
```

```
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "gradingPeriod",
              "description": "The JSON representation of the \"gradingPeriod\" resource to be updated.",
              "type": "gradingPeriod",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteGradingPeriodById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "gradingPeriod": {
      "id": "gradingPeriod",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "schoolReference": {
          "type": "schoolReference",
          "required": true,
          "description": "A reference to the related School resource."
        },
        "beginDate": {
          "type": "date-time",
          "required": true,
          "description": "Month, day, and year of the first day of the GradingPeriod."
        },
        "endDate": {
          "type": "date-time",
          "required": true,
          "description": "Month, day, and year of the last day of the GradingPeriod."
        },
        "descriptor": {
          "type": "string",
          "required": true,
          "description": "The name of the period for which grades are reported."
        },
        "periodSequence": {
          "type": "integer",
          "required": true,
          "description": "The sequential order of this period relative to other periods."
        },
        "totalInstructionalDays": {
          "type": "integer",
          "required": true,
          "description": "Total days available for educational instruction during the GradingPeriod."
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "schoolReference": {
      "id": "schoolReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/graduationPlans

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/graduationPlans",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/graduationPlans",
      "description": "This entity is a plan outlining the required credits, credits by subject,credits by
course, and other criteria required for graduation. A graduation plan may be one or more standard plans defined
by an education organization and/or individual plans for some or all students.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getGraduationPlansAll",
          "type": "array",
          "items": {
            "$ref": "graduationPlan"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
```

```
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "graduationPlan"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getGraduationPlansByExample",
          "type": "array",
          "items": {
            "$ref": "graduationPlan"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "typeDescriptor",
              "description": "The type of academic plan the student is following for graduation: for example,
Minimum, Recommended, Distinguished, or Standard.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "graduationSchoolYear",
              "description": "The school year the student is expected to graduate.",
              "type": "integer",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "graduationPlan"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getGraduationPlanByKey",
          "type": "graduationPlan",
          "parameters": [
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "typeDescriptor",
              "description": "The type of academic plan the student is following for graduation: for example,
Minimum, Recommended, Distinguished, or Standard.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "graduationSchoolYear",
              "description": "The school year the student is expected to graduate.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "graduationPlan"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postGraduationPlans",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "graduationPlan",
              "description": "The JSON representation of the \"graduationPlan\" resource to be created or
updated.",
              "type": "graduationPlan",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/graduationPlans/{id}",
      "description": "This entity is a plan outlining the required credits, credits by subject,credits by
course, and other criteria required for graduation. A graduation plan may be one or more standard plans defined
by an education organization and/or individual plans for some or all students.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getGraduationPlansById",
          "type": "graduationPlan",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "graduationPlan"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putGraduationPlan",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "graduationPlan",
              "description": "The JSON representation of the \"graduationPlan\" resource to be updated.",
              "type": "graduationPlan",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteGraduationPlanById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "graduationPlan": {
      "id": "graduationPlan",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "educationOrganizationReference": {
          "type": "educationOrganizationReference",
          "required": true,
          "description": "A reference to the related EducationOrganization resource."
        },
        "graduationSchoolYearTypeReference": {
          "type": "schoolYearTypeReference",
          "required": true,
          "description": "A reference to the related SchoolYearType resource."
        },
        "typeDescriptor": {
          "type": "string",
          "required": true,
          "description": "The type of academic plan the student is following for graduation: for example,
Minimum, Recommended, Distinguished, or Standard."
        },
        "individualPlan": {
          "type": "boolean",
          "required": false,
          "description": "An indicator of whether the GraduationPlan is tailored for an individual."
        },
        "totalRequiredCreditConversion": {
          "type": "number",
          "required": false,
          "description": "Conversion factor that when multiplied by the number of credits is equivalent to
Carnegie units."
        },
        "totalRequiredCredits": {
          "type": "number",
          "required": true,
          "description": "The value of credits or units of value awarded for the completion of a course."
        },
        "totalRequiredCreditType": {
          "type": "string",
          "required": false,
          "description": "The type of credits or units of value awarded for the completion of a course."
        },
        "creditsByCourses": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of graduationPlanCreditsByCourses.  The total credits
```

```
required for graduation by taking a specific course, or by taking one or more from a set of courses.",
          "items": {
            "$ref": "graduationPlanCreditsByCourse"
          }
        },
        "creditsBySubjects": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of graduationPlanCreditsBySubjects.  The total credits
required in subject to graduate. Only those courses identified as a high school course requirement are eligible
to meet subject credit requirements.",
          "items": {
            "$ref": "graduationPlanCreditsBySubject"
          }
        },
        "requiredAssessments": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of graduationPlanRequiredAssessments.  The assessments and
associated required score and performance level needed to satisfy graduation requirements.",
          "items": {
            "$ref": "graduationPlanRequiredAssessment"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "educationOrganizationReference": {
      "id": "educationOrganizationReference",
      "properties": {
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        }
      }
    },
    "schoolYearTypeReference": {
      "id": "schoolYearTypeReference",
      "properties": {
        "schoolYear": {
          "type": "integer",
          "required": true,
          "description": "Key for School Year"
        }
      }
    },
    "graduationPlanCreditsByCourse": {
      "id": "graduationPlanCreditsByCourse",
      "properties": {
        "courseSetName": {
          "type": "string",
          "required": true,
          "description": "Identifying name given to a collection of courses."
        },
        "creditType": {
          "type": "string",
          "required": false,
          "description": "The type of credits or units of value awarded for the completion of a course."
        },
        "whenTakenGradeLevelDescriptor": {
          "type": "string",
          "required": false,
          "description": "The grade level when the student is planned to take the course."
        },
        "credits": {
          "type": "number",
          "required": true,
          "description": "The value of credits or units of value awarded for the completion of a course."
        },
        "creditConversion": {
          "type": "number",
          "required": false,
          "description": "Conversion factor that when multiplied by the number of credits is equivalent to
Carnegie units."
        },
        "courses": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of graduationPlanCreditsByCourseCourses.  The course
reference that identifies the organization of subject matter and related learning experiences provided for the
instruction of students.",
          "items": {
            "$ref": "graduationPlanCreditsByCourseCourse"
          }
        }
      }
    },
    "graduationPlanCreditsBySubject": {
      "id": "graduationPlanCreditsBySubject",
      "properties": {
        "academicSubjectDescriptor": {
          "type": "string",
          "required": true,
          "description": "The intended major subject area of the graduation requirement."
        },
        "creditType": {
          "type": "string",
          "required": false,
          "description": "The type of credits or units of value awarded for the completion of a course."
        },
        "credits": {
          "type": "number",
          "required": true,
          "description": "The value of credits or units of value awarded for the completion of a course."
        },
        "creditConversion": {
          "type": "number",
          "required": false,
          "description": "Conversion factor that when multiplied by the number of credits is equivalent to
Carnegie units."
        }
      }
    },
    "graduationPlanRequiredAssessment": {
      "id": "graduationPlanRequiredAssessment",
      "properties": {
        "assessmentReference": {
          "type": "assessmentReference",
          "required": true,
          "description": "A reference to the related Assessment resource."
        },
        "assessmentPerformanceLevel": {
          "type": "graduationPlanRequiredAssessmentAssessmentPerformanceLevel",
          "required": false,
          "description": "Performance level required to be met or exceeded."
        },
        "scores": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of graduationPlanRequiredAssessmentScores.  Score required to
be met or exceeded.",
          "items": {
            "$ref": "graduationPlanRequiredAssessmentScore"
          }
        }
      }
    },
    "graduationPlanCreditsByCourseCourse": {
      "id": "graduationPlanCreditsByCourseCourse",
      "properties": {
        "courseReference": {
          "type": "courseReference",
          "required": true,
          "description": "A reference to the related Course resource."
        }
      }
    },
    "graduationPlanRequiredAssessmentScore": {
      "id": "graduationPlanRequiredAssessmentScore",
      "properties": {
        "assessmentReportingMethodType": {
          "type": "string",
          "required": true,
          "description": "The method that the administrator of the assessment uses to report the performance
and achievement of all students. It may be a qualitative method such as performance level descriptors or a
quantitative method such as a numerical grade or cut score. More than one type of reporting method may be used."
        },
        "resultDatatypeType": {
          "type": "string",
          "required": false,
          "description": "The datatype of the result. The results can be expressed as a number, percentile,
range, level, etc."
        },
        "minimumScore": {
          "type": "string",
          "required": false,
          "description": "The minimum score possible on the assessment."
        },
        "maximumScore": {
          "type": "string",
          "required": false,
          "description": "The maximum score possible on the assessment."
        }
      }
    },
    "graduationPlanRequiredAssessmentAssessmentPerformanceLevel": {
      "id": "graduationPlanRequiredAssessmentAssessmentPerformanceLevel",
      "properties": {
        "assessmentReportingMethodType": {
          "type": "string",
          "required": true,
          "description": "The method that the instructor of the class uses to report the performance and
achievement of all students. It may be a qualitative method such as individualized teacher comments or a
quantitative method such as a letter or numerical grade. In some cases, more than one type of reporting method
may be used."
        },
        "performanceLevelDescriptor": {
          "type": "string",
          "required": true,
          "description": "The performance level(s) defined for the assessment."
        },
        "resultDatatypeType": {
          "type": "string",
          "required": false,
          "description": "The datatype of the result. The results can be expressed as a number, percentile,
range, level, etc."
        },
        "minimumScore": {
          "type": "string",
          "required": false,
          "description": "The minimum score required to make the indicated level of performance."
        },
        "maximumScore": {
          "type": "string",
          "required": false,
          "description": "The maximum score to make the indicated level of performance."
        }
      }
    },
    "assessmentReference": {
      "id": "assessmentReference",
      "properties": {
        "title": {
          "type": "string",
          "required": true,
          "description": "The title or name of the Assessment."
        },
        "assessedGradeLevelDescriptor": {
          "type": "string",
          "required": true,
          "description": "The typical grade level for which an assessment is designed. If the assessment spans
a range of grades, then this attribute holds the highest grade assessed. If only one grade level is assessed
then only this attribute is used. For example:          Adult          Prekindergarten          First
grade          Second grade          ..."
        },
        "academicSubjectDescriptor": {
          "type": "string",
          "required": true,
          "description": "The description of the content or subject area (e.g., arts, mathematics, reading,
stenography, or a foreign language) of an assessment."
        },
        "version": {
          "type": "integer",
          "required": true,
          "description": "The version identifier for the assessment."
        }
      }
    },
    "courseReference": {
      "id": "courseReference",
      "properties": {
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        },
        "code": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a course."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/locations

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/locations",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/locations",
      "description": "This entity represents the physical space where students gather for a particular class
/section. The Location may be an indoor or outdoor area designated for the purpose of meeting the educational
needs of students.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getLocationsAll",
          "type": "array",
          "items": {
            "$ref": "location"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "location"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getLocationsByExample",
          "type": "array",
          "items": {
            "$ref": "location"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "classroomIdentificationCode",
              "description": "A unique number or alphanumeric code assigned to a room by a school, school
system, state, or other agency or entity.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
```

```
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "location"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getLocationByKey",
          "type": "location",
          "parameters": [
            {
              "paramType": "query",
              "name": "classroomIdentificationCode",
              "description": "A unique number or alphanumeric code assigned to a room by a school, school
system, state, or other agency or entity.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "location"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postLocations",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "location",
              "description": "The JSON representation of the \"location\" resource to be created or updated.",
              "type": "location",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
```

```
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/locations/{id}",
      "description": "This entity represents the physical space where students gather for a particular class
/section. The Location may be an indoor or outdoor area designated for the purpose of meeting the educational
needs of students.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getLocationsById",
          "type": "location",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "location"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putLocation",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "location",
              "description": "The JSON representation of the \"location\" resource to be updated.",
              "type": "location",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
```

```
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteLocationById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "location": {
      "id": "location",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "schoolReference": {
          "type": "schoolReference",
          "required": true,
          "description": "A reference to the related School resource."
        },
        "classroomIdentificationCode": {
          "type": "string",
          "required": true,
          "description": "A unique number or alphanumeric code assigned to a room by a school, school system,
state, or other agency or entity."
        },
        "maximumNumberOfSeats": {
          "type": "integer",
          "required": true,
          "description": "The most number of seats the class can maintain."
        },
        "optimalNumberOfSeats": {
          "type": "integer",
          "required": false,
          "description": "The number of seats that is most favorable to the class."
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "schoolReference": {
      "id": "schoolReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/parents

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/parents",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/parents",
      "description": "This entity represents a parent or guardian of a student, such as mother, father, or
caretaker.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getParentsAll",
          "type": "array",
          "items": {
            "$ref": "parent"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "parent"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getParentsByExample",
          "type": "array",
          "items": {
            "$ref": "parent"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "parentUniqueId",
              "description": "A unique alphanumeric code assigned to a parent.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "parent"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
```

```
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getParentByKey",
          "type": "parent",
          "parameters": [
            {
              "paramType": "query",
              "name": "parentUniqueId",
              "description": "A unique alphanumeric code assigned to a parent.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "parent"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postParents",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "parent",
              "description": "The JSON representation of the \"parent\" resource to be created or updated.",
              "type": "parent",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/parents/{id}",
      "description": "This entity represents a parent or guardian of a student, such as mother, father, or
caretaker.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getParentsById",
          "type": "parent",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "parent"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putParent",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "parent",
              "description": "The JSON representation of the \"parent\" resource to be updated.",
              "type": "parent",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteParentById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "parent": {
      "id": "parent",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "firstName": {
          "type": "string",
          "required": true,
          "description": "A name given to an individual at birth, baptism, or during another naming ceremony,
or through legal change."
        },
        "generationCodeSuffix": {
          "type": "string",
          "required": false,
          "description": "An appendage, if any, used to denote an individual's generation in his family (e.g.,
Jr., Sr., III)."
        },
        "lastSurname": {
          "type": "string",
          "required": true,
          "description": "The name borne in common by members of a family."
        },
        "loginId": {
          "type": "string",
          "required": false,
          "description": "The login ID for the user; used for security access control interface."
        },
        "maidenName": {
          "type": "string",
          "required": false,
          "description": "The person's maiden name."
        },
        "middleName": {
          "type": "string",
          "required": false,
          "description": "A secondary name given to an individual at birth, baptism, or during another naming
ceremony."
        },
        "parentUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a parent."
        },
        "personalTitlePrefix": {
          "type": "string",
          "required": true,
          "description": "A prefix used to denote the title, degree, position, or seniority of the person."
        },
        "sexType": {
          "type": "string",
          "required": true,
          "description": "A person's gender."
        },
        "addresses": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of parentAddresses.  Parent's address, if different from the
student address.",
          "items": {
            "$ref": "parentAddress"
          }
        },
        "electronicMails": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of parentElectronicMails.  The numbers, letters, and symbols
used to identify an electronic mail (e-mail) user within the network to which the individual or organization
belongs.",
          "items": {
            "$ref": "parentElectronicMail"
          }
        },
        "identificationDocuments": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of parentIdentificationDocuments.  The documents presented as
evident to verify one's personal identity; for example: drivers license, passport, birth certificate, etc.",
          "items": {
            "$ref": "parentIdentificationDocument"
          }
        },
        "internationalAddresses": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of parentInternationalAddresses.  The set of elements that
describes an international address.",
          "items": {
            "$ref": "parentInternationalAddress"
          }
        },
        "otherNames": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of parentOtherNames.  Other names (e.g., alias, nickname,
previous legal name) associated with a person.",
          "items": {
            "$ref": "parentOtherName"
          }
        },
        "telephones": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of parentTelephones.  The 10-digit telephone number,
including the area code, for the person.",
          "items": {
            "$ref": "parentTelephone"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "parentAddress": {
      "id": "parentAddress",
      "properties": {
        "addressType": {
          "type": "string",
          "required": true,
          "description": "The type of address listed for an individual or organization.    For example:
Physical Address, Mailing Address, Home Address, etc.)"
        },
        "stateAbbreviationType": {
          "type": "string",
          "required": true,
          "description": "The abbreviation for the state (within the United States) or outlying area in which
an address is located."
        },
        "streetNumberName": {
          "type": "string",
          "required": true,
          "description": "The street number and street name or post office box number of an address."
        },
        "apartmentRoomSuiteNumber": {
          "type": "string",
          "required": true,
          "description": "The apartment, room, or suite number of an address."
        },
        "buildingSiteNumber": {
          "type": "string",
          "required": false,
          "description": "The number of the building on the site, if more than one building shares the same
address."
        },
        "city": {
          "type": "string",
          "required": true,
          "description": "The name of the city in which an address is located."
        },
        "postalCode": {
          "type": "string",
          "required": true,
          "description": "The five or nine digit zip code or overseas postal code portion of an address."
        },
        "nameOfCounty": {
          "type": "string",
          "required": true,
          "description": "The name of the county, parish, borough, or comparable unit (within a state)
in                        'which an address is located."
        },
        "countyFIPSCode": {
          "type": "string",
          "required": false,
          "description": "The Federal Information Processing Standards (FIPS) numeric code for the county
issued by the National Institute of Standards and Technology (NIST). Counties are considered to be the \"first-
order subdivisions\" of each State and statistically equivalent entity, regardless of their local designations
(county, parish, borough, etc.) Counties in different States will have the same code. A unique county number is
created when combined with the 2-digit FIPS State Code."
        },
        "latitude": {
          "type": "string",
          "required": false,
          "description": "The geographic latitude of the physical address."
        },
        "longitude": {
          "type": "string",
          "required": false,
          "description": "The geographic longitude of the physical address."
        },
        "beginDate": {
          "type": "date-time",
          "required": false,
          "description": "The first date the address is valid. For physical addresses, the date the person
moved to that address."
        },
        "endDate": {
          "type": "date-time",
          "required": false,
          "description": "The last date the address is valid. For physical addresses, this would be the date
the person moved from that address."
        }
      }
    },
    "parentElectronicMail": {
      "id": "parentElectronicMail",
      "properties": {
        "electronicMailType": {
          "type": "string",
          "required": true,
          "description": "The type of email listed for an individual or organization. For example: Home
/Personal, Work, etc.)"
        },
        "electronicMailAddress": {
          "type": "string",
          "required": true,
          "description": "The electronic mail (e-mail) address listed for an individual or organization."
        },
        "primaryEmailAddressIndicator": {
          "type": "boolean",
          "required": false,
          "description": "An indication that the electronic mail address should be used as the principal
electronic mail address for an individual or organization."
        }
      }
    },
    "parentIdentificationDocument": {
      "id": "parentIdentificationDocument",
      "properties": {
        "identificationDocumentUseType": {
          "type": "string",
          "required": true,
          "description": "The primary function of the document used for establishing identity."
        },
        "personalInformationVerificationType": {
          "type": "string",
          "required": true,
          "description": "The category of the document relative to its purpose."
        },
        "issuerCountryDescriptor": {
          "type": "string",
          "required": false,
          "description": "Country of origin of the document."
        },
        "documentTitle": {
          "type": "string",
          "required": false,
          "description": "The title of the document given by the issuer."
        },
        "documentExpirationDate": {
          "type": "date-time",
          "required": false,
          "description": "The day when the document  expires, if null then never expires."
        },
        "issuerDocumentIdentificationCode": {
          "type": "string",
          "required": false,
          "description": "The unique identifier on the issuer's identification system."
        },
        "issuerName": {
          "type": "string",
          "required": false,
          "description": "Name of the entity or institution that issued the document."
        }
      }
    },
    "parentInternationalAddress": {
      "id": "parentInternationalAddress",
      "properties": {
        "addressType": {
          "type": "string",
          "required": true,
          "description": "The type of address listed for an individual or organization. For example:  Physical
Address, Mailing Address, Home Address, etc.)"
        },
        "countryDescriptor": {
          "type": "string",
          "required": true,
          "description": "The name of the country."
        },
        "addressLine1": {
          "type": "string",
          "required": true,
          "description": "The first line of the address."
        },
        "addressLine2": {
          "type": "string",
          "required": false,
          "description": "The second line of the address."
        },
        "addressLine3": {
          "type": "string",
          "required": false,
          "description": "The third line of the address."
        },
        "addressLine4": {
          "type": "string",
          "required": false,
          "description": "The fourth line of the address."
        },
        "latitude": {
          "type": "string",
          "required": false,
          "description": "The geographic latitude of the physical address."
        },
        "longitude": {
          "type": "string",
          "required": false,
          "description": "The geographic longitude of the physical address."
        },
        "beginDate": {
          "type": "date-time",
          "required": false,
          "description": "The first date the address is valid. For physical addresses, the date the person
moved to that address."
        },
        "endDate": {
          "type": "date-time",
          "required": false,
          "description": "The last date the address is valid. For physical addresses, this would be the date
the person moved from that address."
        }
      }
    },
    "parentOtherName": {
      "id": "parentOtherName",
      "properties": {
        "otherNameType": {
          "type": "string",
          "required": true,
          "description": "The types of alternate names for a person."
        },
        "personalTitlePrefix": {
          "type": "string",
          "required": false,
          "description": "A prefix used to denote the title, degree, position, or seniority of the person."
        },
        "firstName": {
          "type": "string",
          "required": true,
          "description": "A name given to an individual at birth, baptism, or during another naming ceremony,
or through legal change."
        },
        "middleName": {
          "type": "string",
          "required": false,
          "description": "A secondary name given to an individual at birth, baptism, or during another naming
ceremony."
        },
        "lastSurname": {
          "type": "string",
          "required": true,
          "description": "The name borne in common by members of a family."
        },
        "generationCodeSuffix": {
          "type": "string",
          "required": false,
          "description": "An appendage, if any, used to denote an individual's generation in his family (e.g.,
Jr., Sr., III)."
        }
      }
    },
    "parentTelephone": {
      "id": "parentTelephone",
      "properties": {
        "telephoneNumberType": {
          "type": "string",
          "required": true,
          "description": "The type of communication number listed for an individual or organization."
        },
        "telephoneNumber": {
          "type": "string",
          "required": true,
          "description": "The telephone number including the area code, and extension, if applicable."
        },
        "orderOfPriority": {
          "type": "integer",
          "required": false,
          "description": "The order of priority assigned to telephone numbers to define which number to attempt
first, second, etc."
        },
        "textMessageCapabilityIndicator": {
          "type": "boolean",
          "required": false,
          "description": "An indication that the telephone number is technically capable of sending and
receiving Short Message Service (SMS) text messages."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/programs

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/programs",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/programs",
      "description": "This entity represents any program designed to work in conjunction with, or as a
supplement to, the main academic program. Programs may provide instruction, training, services, or benefits
through federal, state, or local agencies. Programs may also include organized extracurricular activities for
students.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getProgramsAll",
          "type": "array",
          "items": {
            "$ref": "program"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "program"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getProgramsByExample",
          "type": "array",
          "items": {
            "$ref": "program"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "name",
              "description": "The formal name of the Program of instruction, training, services, or benefits
available through federal, state, or local agencies.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "type",
              "description": "The type of program.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "program"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getProgramByKey",
          "type": "program",
          "parameters": [
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "name",
              "description": "The formal name of the Program of instruction, training, services, or benefits
available through federal, state, or local agencies.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "type",
              "description": "The type of program.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "program"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postPrograms",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "program",
              "description": "The JSON representation of the \"program\" resource to be created or updated.",
              "type": "program",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/programs/{id}",
      "description": "This entity represents any program designed to work in conjunction with, or as a
supplement to, the main academic program. Programs may provide instruction, training, services, or benefits
through federal, state, or local agencies. Programs may also include organized extracurricular activities for
students.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getProgramsById",
          "type": "program",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "program"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putProgram",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "program",
              "description": "The JSON representation of the \"program\" resource to be updated.",
              "type": "program",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
```

```
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteProgramById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
```

```
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "program": {
      "id": "program",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "educationOrganizationReference": {
          "type": "educationOrganizationReference",
          "required": true,
          "description": "A reference to the related EducationOrganization resource."
        },
        "programId": {
          "type": "string",
          "required": true,
          "description": "A unique number or alphanumeric code assigned to a program by a school, school
system, a state, or other agency or entity."
        },
        "name": {
          "type": "string",
          "required": true,
          "description": "The formal name of the Program of instruction, training, services, or benefits
available through federal, state, or local agencies."
        },
        "sponsorType": {
          "type": "string",
          "required": false,
          "description": "Ultimate and intermediate providers of funds for a particular educational or service
program or activity, or for an individual's participation in the program or activity (e.g., Federal, State,
ESC, District, School, Private Organization)."
        },
        "type": {
          "type": "string",
          "required": true,
          "description": "The type of program."
        },
        "characteristics": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of programCharacteristics.  Reflects important
characteristics of the Program, such as categories or particular indications.",
          "items": {
            "$ref": "programCharacteristic"
          }
        },
        "learningObjectives": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of programLearningObjectives.  References the
LearningObjective(s) with which the Program is associated.",
          "items": {
            "$ref": "programLearningObjective"
          }
        },
        "learningStandards": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of programLearningStandards.  LearningStandard followed by
this program.",
          "items": {
            "$ref": "programLearningStandard"
          }
        },
        "services": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of programServices.  Defines the services this program
provides to students.",
          "items": {
            "$ref": "programService"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "educationOrganizationReference": {
      "id": "educationOrganizationReference",
      "properties": {
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        }
      }
    },
    "programCharacteristic": {
      "id": "programCharacteristic",
      "properties": {
        "descriptor": {
          "type": "string",
          "required": true,
          "description": "Reflects important characteristics of the Program, such as categories or particular
indications."
        }
      }
    },
    "programLearningObjective": {
      "id": "programLearningObjective",
      "properties": {
        "learningObjectiveReference": {
          "type": "learningObjectiveReference",
          "required": true,
          "description": "A reference to the related LearningObjective resource."
        }
      }
    },
    "programLearningStandard": {
      "id": "programLearningStandard",
      "properties": {
        "learningStandardReference": {
          "type": "learningStandardReference",
          "required": true,
          "description": "A reference to the related LearningStandard resource."
        }
      }
    },
    "programService": {
      "id": "programService",
      "properties": {
        "serviceDescriptor": {
          "type": "string",
          "required": true,
          "description": "Defines the services this program provides to students."
        }
      }
    },
    "learningObjectiveReference": {
      "id": "learningObjectiveReference",
      "properties": {
        "objective": {
          "type": "string",
          "required": true,
          "description": "The designated title of the LearningObjective."
        },
        "academicSubjectDescriptor": {
          "type": "string",
          "required": true,
          "description": "The description of the content or subject area (e.g., arts, mathematics, reading,
stenography, or a foreign language) of an assessment."
        },
        "objectiveGradeLevelDescriptor": {
          "type": "string",
          "required": true,
          "description": "The grade level for which the LearningObjective is targeted."
        }
      }
    },
    "learningStandardReference": {
      "id": "learningStandardReference",
      "properties": {
        "learningStandardId": {
          "type": "string",
          "required": true,
          "description": "The identifier for the specific learning standard (e.g., 111.15.3.1.A)."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/schools

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/schools",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/schools",
      "description": "This entity represents an educational organization that includes staff and students who
participate in classes and educational activity groups.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getSchoolsAll",
          "type": "array",
          "items": {
            "$ref": "school"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "school"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getSchoolsByExample",
          "type": "array",
          "items": {
            "$ref": "school"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "school"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
```

```
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getSchoolByKey",
          "type": "school",
          "parameters": [
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "school"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postSchools",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "school",
              "description": "The JSON representation of the \"school\" resource to be created or updated.",
              "type": "school",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/schools/{id}",
      "description": "This entity represents an educational organization that includes staff and students who
participate in classes and educational activity groups.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getSchoolsById",
          "type": "school",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "school"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putSchool",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "school",
              "description": "The JSON representation of the \"school\" resource to be updated.",
              "type": "school",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteSchoolById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "school": {
      "id": "school",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "localEducationAgencyReference": {
          "type": "localEducationAgencyReference",
          "required": true,
          "description": "A reference to the related LocalEducationAgency resource."
        },
        "charterApprovalSchoolYearTypeReference": {
          "type": "schoolYearTypeReference",
          "required": false,
          "description": "A reference to the related SchoolYearType resource."
        },
        "administrativeFundingControlDescriptor": {
          "type": "string",
          "required": false,
          "description": "The type of education institution as classified by its funding source, for example
public or private."
        },
        "charterApprovalAgencyType": {
          "type": "string",
          "required": false,
          "description": "The type of agency that approved the establishment or continuation of a charter
school."
        },
        "charterStatusType": {
          "type": "string",
          "required": false,
          "description": "A school or agency providing free public elementary or secondary education to
eligible students under a specific charter granted by the state legislature or other appropriate authority and
designated by such authority to be a charter school."
```

```
        },
        "internetAccessType": {
          "type": "string",
          "required": false,
          "description": "The type of Internet access available."
        },
        "magnetSpecialProgramEmphasisSchoolType": {
          "type": "string",
          "required": false,
          "description": "A school that has been designed: 1) to attract students of different racial/ethnic
backgrounds for the purpose of reducing, preventing, or eliminating racial isolation; and/or 2) to provide an
academic or social focus on a particular theme (e.g., science/math, performing arts, gifted/talented, or
foreign language)."
        },
        "nameOfInstitution": {
          "type": "string",
          "required": true,
          "description": "The full, legally accepted name of the institution."
        },
        "operationalStatusType": {
          "type": "string",
          "required": false,
          "description": "The current operational status of the EducationOrganization (e.g., active, inactive)."
        },
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        },
        "type": {
          "type": "string",
          "required": false,
          "description": "The type of education institution as classified by its primary focus."
        },
        "shortNameOfInstitution": {
          "type": "string",
          "required": true,
          "description": "A short name for the institution."
        },
        "stateOrganizationId": {
          "type": "string",
          "required": true,
          "description": "The identifier assigned to an education organization by the StateEducationAgency
(SEA)."
        },
        "titleIPartASchoolDesignationType": {
          "type": "string",
          "required": false,
          "description": "Denotes the Title I Part A designation for the school."
        },
        "webSite": {
          "type": "string",
          "required": false,
          "description": "The public web site address (URL) for the EducationOrganization."
        },
        "schoolCategories": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of schoolCategories.  The one or more categories of school.
For example: High School, Middle School, and/or Elementary School.",
          "items": {
            "$ref": "schoolCategory"
          }
        },
        "gradeLevels": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of schoolGradeLevels.  The grade levels served at the
school.",
          "items": {
            "$ref": "schoolGradeLevel"
          }
        },
        "addresses": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of educationOrganizationAddresses.  The set of elements that
describes the physical location of the education entity, including the street address, city, state, ZIP code,
and ZIP code + 4.",
          "items": {
            "$ref": "educationOrganizationAddress"
          }
        },
        "educationOrganizationCategories": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of educationOrganizationCategories.  The classification of
the education agency within the geographic boundaries of a state according to the level of administrative and
operational control granted by the state.",
          "items": {
            "$ref": "educationOrganizationCategory"
          }
        },
        "identificationCodes": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of educationOrganizationIdentificationCodes.  A unique number
or alphanumeric code assigned to an education organization by a school, school system, a state, or other agency
or entity.",
          "items": {
            "$ref": "educationOrganizationIdentificationCode"
          }
        },
        "institutionTelephones": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of educationOrganizationInstitutionTelephones.  The 10-digit
telephone number, including the area code, for the education entity.",
          "items": {
            "$ref": "educationOrganizationInstitutionTelephone"
          }
        },
        "internationalAddresses": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of educationOrganizationInternationalAddresses.  The set of
elements that describes the international physical location of the education entity.",
          "items": {
            "$ref": "educationOrganizationInternationalAddress"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "localEducationAgencyReference": {
      "id": "localEducationAgencyReference",
      "properties": {
        "localEducationAgencyId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a local education agency by the State Education Agency
(SEA)."
        }
      }
    },
    "schoolYearTypeReference": {
      "id": "schoolYearTypeReference",
      "properties": {
        "schoolYear": {
          "type": "integer",
          "required": true,
          "description": "Key for School Year"
        }
      }
    },
    "schoolCategory": {
      "id": "schoolCategory",
      "properties": {
        "type": {
          "type": "string",
          "required": true,
          "description": "The one or more categories of school. For example: High School, Middle School, and/or
Elementary School."
        }
      }
    },
    "schoolGradeLevel": {
      "id": "schoolGradeLevel",
      "properties": {
        "gradeLevelDescriptor": {
          "type": "string",
          "required": true,
          "description": "The grade levels served at the school."
        }
      }
    },
    "educationOrganizationAddress": {
      "id": "educationOrganizationAddress",
      "properties": {
        "addressType": {
          "type": "string",
          "required": true,
          "description": "The type of address listed for an individual or organization.    For example:
Physical Address, Mailing Address, Home Address, etc.)"
        },
        "stateAbbreviationType": {
          "type": "string",
          "required": true,
          "description": "The abbreviation for the state (within the United States) or outlying area in which
an address is located."
        },
        "streetNumberName": {
          "type": "string",
          "required": true,
          "description": "The street number and street name or post office box number of an address."
        },
        "apartmentRoomSuiteNumber": {
          "type": "string",
          "required": false,
          "description": "The apartment, room, or suite number of an address."
        },
        "buildingSiteNumber": {
          "type": "string",
          "required": false,
          "description": "The number of the building on the site, if more than one building shares the same
address."
        },
        "city": {
          "type": "string",
          "required": true,
          "description": "The name of the city in which an address is located."
        },
        "postalCode": {
          "type": "string",
          "required": true,
          "description": "The five or nine digit zip code or overseas postal code portion of an address."
        },
        "nameOfCounty": {
          "type": "string",
          "required": false,
          "description": "The name of the county, parish, borough, or comparable unit (within a state)
in                         'which an address is located."
        },
        "countyFIPSCode": {
          "type": "string",
          "required": false,
          "description": "The Federal Information Processing Standards (FIPS) numeric code for the county
issued by the National Institute of Standards and Technology (NIST). Counties are considered to be the \"first-
order subdivisions\" of each State and statistically equivalent entity, regardless of their local designations
(county, parish, borough, etc.) Counties in different States will have the same code. A unique county number is
created when combined with the 2-digit FIPS State Code."
        },
        "latitude": {
          "type": "string",
          "required": false,
          "description": "The geographic latitude of the physical address."
        },
        "longitude": {
          "type": "string",
          "required": false,
          "description": "The geographic longitude of the physical address."
        },
        "beginDate": {
          "type": "date-time",
          "required": false,
          "description": "The first date the address is valid. For physical addresses, the date the person
moved to that address."
        },
        "endDate": {
          "type": "date-time",
          "required": false,
          "description": "The last date the address is valid. For physical addresses, this would be the date
the person moved from that address."
        }
      }
    },
    "educationOrganizationCategory": {
      "id": "educationOrganizationCategory",
      "properties": {
        "type": {
          "type": "string",
          "required": true,
          "description": "The classification of the education agency within the geographic boundaries of a
state according to the level of administrative and operational control granted by the state."
        }
      }
    },
    "educationOrganizationIdentificationCode": {
      "id": "educationOrganizationIdentificationCode",
      "properties": {
        "educationOrganizationIdentificationSystemDescriptor": {
          "type": "string",
          "required": true,
          "description": "The school system, state, or agency assigning the identification code."
        },
        "identificationCode": {
          "type": "string",
          "required": true,
          "description": "A unique number or alphanumeric code that is assigned to an education organization by
a school, school system, state, or other agency or entity."
        }
      }
    },
    "educationOrganizationInstitutionTelephone": {
      "id": "educationOrganizationInstitutionTelephone",
      "properties": {
        "institutionTelephoneNumberType": {
          "type": "string",
          "required": true,
          "description": "The type of communication number listed for an individual or organization."
        },
        "telephoneNumber": {
          "type": "string",
          "required": true,
          "description": "The telephone number including the area code, and extension, if applicable."
        }
      }
    },
    "educationOrganizationInternationalAddress": {
      "id": "educationOrganizationInternationalAddress",
      "properties": {
        "addressType": {
          "type": "string",
          "required": true,
          "description": "The type of address listed for an individual or organization. For example:  Physical
Address, Mailing Address, Home Address, etc.)"
        },
        "countryDescriptor": {
          "type": "string",
          "required": true,
          "description": "The name of the country."
        },
        "addressLine1": {
          "type": "string",
          "required": true,
          "description": "The first line of the address."
        },
        "addressLine2": {
          "type": "string",
          "required": false,
          "description": "The second line of the address."
        },
        "addressLine3": {
          "type": "string",
          "required": false,
          "description": "The third line of the address."
        },
        "addressLine4": {
          "type": "string",
          "required": false,
          "description": "The fourth line of the address."
        },
        "latitude": {
          "type": "string",
          "required": false,
          "description": "The geographic latitude of the physical address."
        },
        "longitude": {
          "type": "string",
          "required": false,
          "description": "The geographic longitude of the physical address."
        },
        "beginDate": {
          "type": "date-time",
          "required": false,
          "description": "The first date the address is valid. For physical addresses, the date the person
moved to that address."
        },
        "endDate": {
          "type": "date-time",
          "required": false,
          "description": "The last date the address is valid. For physical addresses, this would be the date
the person moved from that address."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/sections

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/sections",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/sections",
      "description": "This entity represents a setting in which organized instruction of course content is
provided, in-person or otherwise, to one or more students for a given period of time. A course offering may be
offered to more than one section.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getSectionsAll",
          "type": "array",
          "items": {
            "$ref": "section"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "section"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getSectionsByExample",
          "type": "array",
          "items": {
            "$ref": "section"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "classPeriodName",
              "description": "An indication of the portion of a typical daily session in which students receive
instruction in a specified subject (e.g., morning, sixth period, block period, or AB schedules).",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "classroomIdentificationCode",
              "description": "A unique number or alphanumeric code assigned to a room by a school, school
system, state, or other agency or entity.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "localCourseCode",
              "description": "The local code assigned by the School that identifies the course offering
provided for the instruction of students.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolYear",
              "description": "The identifier for the school year.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "sequenceOfCourse",
              "description": "When a section is part of a sequence of parts for a course, the number of the
sequence. If the course has only one part, the value of this section attribute should be 1.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "termDescriptor",
              "description": "The term for the Session during the school year.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "uniqueSectionCode",
              "description": "A unique identifier for the Section that is defined by the classroom, the
subjects taught, and the instructors who are assigned.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "section"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getSectionByKey",
          "type": "section",
          "parameters": [
            {
              "paramType": "query",
              "name": "classPeriodName",
              "description": "An indication of the portion of a typical daily session in which students receive
instruction in a specified subject (e.g., morning, sixth period, block period, or AB schedules).",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "classroomIdentificationCode",
              "description": "A unique number or alphanumeric code assigned to a room by a school, school
system, state, or other agency or entity.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "localCourseCode",
              "description": "The local code assigned by the School that identifies the course offering
provided for the instruction of students.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolYear",
              "description": "The identifier for the school year.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "sequenceOfCourse",
              "description": "When a section is part of a sequence of parts for a course, the number of the
sequence. If the course has only one part, the value of this section attribute should be 1.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "termDescriptor",
              "description": "The term for the Session during the school year.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "uniqueSectionCode",
              "description": "A unique identifier for the Section that is defined by the classroom, the
subjects taught, and the instructors who are assigned.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "section"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postSections",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "section",
              "description": "The JSON representation of the \"section\" resource to be created or updated.",
              "type": "section",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/sections/{id}",
      "description": "This entity represents a setting in which organized instruction of course content is
provided, in-person or otherwise, to one or more students for a given period of time. A course offering may be
offered to more than one section.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getSectionsById",
          "type": "section",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "section"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putSection",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "section",
              "description": "The JSON representation of the \"section\" resource to be updated.",
              "type": "section",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteSectionById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "section": {
      "id": "section",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "classPeriodReference": {
          "type": "classPeriodReference",
          "required": true,
          "description": "A reference to the related ClassPeriod resource."
        },
        "courseOfferingReference": {
          "type": "courseOfferingReference",
          "required": true,
          "description": "A reference to the related CourseOffering resource."
        },
        "locationReference": {
          "type": "locationReference",
          "required": true,
          "description": "A reference to the related Location resource."
        },
        "schoolReference": {
          "type": "schoolReference",
          "required": true,
          "description": "A reference to the related School resource."
        },
        "availableCreditConversion": {
          "type": "number",
          "required": false,
          "description": "Conversion factor that when multiplied by the number of credits is equivalent to
Carnegie units."
        },
        "availableCredits": {
          "type": "number",
          "required": true,
          "description": "The value of credits or units of value awarded for the completion of a course."
        },
        "availableCreditType": {
          "type": "string",
          "required": false,
          "description": "The type of credits or units of value awarded for the completion of a course."
        },
        "educationalEnvironmentType": {
          "type": "string",
          "required": true,
          "description": "The setting in which a child receives education and related services; for
example:          Center-based instruction          Home-based instruction          Hospital class
Mainstream          Residential care and treatment facility          ..."
        },
        "instructionLanguageDescriptor": {
          "type": "string",
          "required": false,
          "description": "The primary language of instruction, if omitted English is assumed."
        },
        "mediumOfInstructionType": {
          "type": "string",
          "required": false,
          "description": "The media through which teachers provide instruction to students and students and
teachers communicate about instructional matters; for example:          Technology-based instruction in
classroom          Correspondence instruction          Face-to-face instruction          Virtual/On-line
Distance learning          Center-based instruction          ..."
        },
        "populationServedType": {
          "type": "string",
          "required": false,
          "description": "The type of students the Section is offered and tailored to; for example:
Bilingual students          Remedial education students          Gifted and talented students          Career
and Technical Education students          Special education students          ..."
        },
        "sequenceOfCourse": {
          "type": "integer",
          "required": true,
          "description": "When a section is part of a sequence of parts for a course, the number of the
sequence. If the course has only one part, the value of this section attribute should be 1."
        },
        "uniqueSectionCode": {
          "type": "string",
          "required": true,
          "description": "A unique identifier for the Section that is defined by the classroom, the subjects
taught, and the instructors who are assigned."
        },
        "characteristics": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of sectionCharacteristics.  Reflects important
characteristics of the Section, such as whether or not attendance is taken and the Section is graded.",
          "items": {
            "$ref": "sectionCharacteristic"
          }
        },
        "programs": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of sectionPrograms.  Optional reference to program (e.g.,
CTE) to which the Section is associated.",
          "items": {
            "$ref": "sectionProgram"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "classPeriodReference": {
      "id": "classPeriodReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        },
        "name": {
          "type": "string",
          "required": true,
          "description": "An indication of the portion of a typical daily session in which students receive
instruction in a specified subject (e.g., morning, sixth period, block period, or AB schedules)."
        }
      }
    },
    "courseOfferingReference": {
      "id": "courseOfferingReference",
      "properties": {
        "localCourseCode": {
          "type": "string",
          "required": true,
          "description": "The local code assigned by the School that identifies the course offering provided
for the instruction of students."
        },
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        },
        "schoolYear": {
          "type": "integer",
          "required": true,
          "description": "The identifier for the school year."
        },
        "termDescriptor": {
          "type": "string",
          "required": true,
          "description": "The term for the Session during the school year."
        }
      }
    },
    "locationReference": {
      "id": "locationReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        },
        "classroomIdentificationCode": {
          "type": "string",
          "required": true,
          "description": "A unique number or alphanumeric code assigned to a room by a school, school system,
state, or other agency or entity."
        }
      }
    },
    "schoolReference": {
      "id": "schoolReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        }
      }
    },
    "sectionCharacteristic": {
      "id": "sectionCharacteristic",
      "properties": {
        "descriptor": {
          "type": "string",
          "required": true,
          "description": "Reflects important characteristics of the Section, such as whether or not attendance
is taken and the Section is graded."
```

```
        }
      }
    },
    "sectionProgram": {
      "id": "sectionProgram",
      "properties": {
        "programReference": {
          "type": "programReference",
          "required": true,
          "description": "A reference to the related Program resource."
        }
      }
    },
    "programReference": {
      "id": "programReference",
      "properties": {
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        },
        "type": {
          "type": "string",
          "required": true,
          "description": "The type of program."
        },
        "name": {
          "type": "string",
          "required": true,
          "description": "The formal name of the Program of instruction, training, services, or benefits
available through federal, state, or local agencies."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/sessions

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/sessions",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/sessions",
      "description": "This entity represents the prescribed span of time when an education institution is open,
instruction is provided and students are under the direction and guidance of teachers and/or education
institution administration. A session may be interrupted by one or more vacations.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getSessionsAll",
          "type": "array",
          "items": {
            "$ref": "session"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "session"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getSessionsByExample",
          "type": "array",
          "items": {
            "$ref": "session"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolYear",
              "description": "The identifier for the school year.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "termDescriptor",
              "description": "The term for the Session during the school year.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "session"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getSessionByKey",
          "type": "session",
          "parameters": [
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolYear",
              "description": "The identifier for the school year.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "termDescriptor",
              "description": "The term for the Session during the school year.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "session"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value 
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body 
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was 
either not provided or is invalid.  The operation may succeed once authenication has been successfully 
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.  
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postSessions",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "session",
              "description": "The JSON representation of the \"session\" resource to be created or updated.",
              "type": "session",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is 
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\" 
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values 
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the 
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not 
yet completed due to current system load. Processing may still fail due to violation of referential integrity 
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of 
the response."
```

```
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/sessions/{id}",
      "description": "This entity represents the prescribed span of time when an education institution is open,
instruction is provided and students are under the direction and guidance of teachers and/or education
institution administration. A session may be interrupted by one or more vacations.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getSessionsById",
          "type": "session",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "session"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putSession",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "session",
              "description": "The JSON representation of the \"session\" resource to be updated.",
              "type": "session",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
```

```
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteSessionById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "session": {
      "id": "session",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "schoolReference": {
          "type": "schoolReference",
          "required": true,
          "description": "A reference to the related School resource."
        },
        "schoolYearTypeReference": {
          "type": "schoolYearTypeReference",
          "required": true,
          "description": "A reference to the related SchoolYearType resource."
        },
        "beginDate": {
          "type": "date-time",
          "required": true,
          "description": "Month, day, and year of the first day of the Session."
        },
        "endDate": {
          "type": "date-time",
          "required": true,
          "description": "Month, day and year of the last day of the Session."
        },
        "name": {
          "type": "string",
          "required": true,
          "description": "The identifier for the calendar for the academic session (e.g., 2010/11, 2011
Summer)."
        },
        "termDescriptor": {
          "type": "string",
          "required": true,
          "description": "The term for the Session during the school year."
        },
        "totalInstructionalDays": {
          "type": "integer",
          "required": true,
          "description": "The total number of instructional days in the school calendar."
        },
        "academicWeeks": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of sessionAcademicWeeks.  The academic weeks associated with
the school year.",
          "items": {
            "$ref": "sessionAcademicWeek"
          }
        },
        "gradingPeriods": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of sessionGradingPeriods.  Grading periods associated with
the calendar.",
          "items": {
            "$ref": "sessionGradingPeriod"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "schoolReference": {
      "id": "schoolReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        }
      }
    },
    "schoolYearTypeReference": {
      "id": "schoolYearTypeReference",
      "properties": {
        "schoolYear": {
          "type": "integer",
          "required": true,
          "description": "Key for School Year"
        }
      }
    },
    "sessionAcademicWeek": {
      "id": "sessionAcademicWeek",
      "properties": {
        "academicWeekReference": {
          "type": "academicWeekReference",
          "required": true,
          "description": "A reference to the related AcademicWeek resource."
        }
      }
    },
    "sessionGradingPeriod": {
      "id": "sessionGradingPeriod",
      "properties": {
        "gradingPeriodReference": {
          "type": "gradingPeriodReference",
          "required": true,
          "description": "A reference to the related GradingPeriod resource."
        }
      }
    },
    "academicWeekReference": {
      "id": "academicWeekReference",
      "properties": {
        "weekIdentifier": {
          "type": "string",
          "required": true,
          "description": "The school label for the week."
        },
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        }
      }
    },
    "gradingPeriodReference": {
      "id": "gradingPeriodReference",
      "properties": {
        "descriptor": {
          "type": "string",
          "required": true,
          "description": "The name of the period for which grades are reported."
        },
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        },
        "beginDate": {
          "type": "date-time",
          "required": true,
          "description": "Month, day, and year of the first day of the GradingPeriod."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/staffs

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/staffs",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/staffs",
      "description": "This entity represents an individual who performs specified activities for any public or
private education institution or agency that provides instructional and/or support services to students or
staff at the early childhood level through high school completion. For example, this includes:      1. An \"
employee\" who performs services under the direction of the employing institution or agency is compensated for
such services by the employer and is eligible for employee benefits and wage or salary tax withholdings      2.
A \"contractor\" or \"consultant\" who performs services for an agreed upon fee or an employee of a management
service contracted to work on site      3. A \"volunteer\" who performs services on a voluntary and
uncompensated basis      4. An in-kind service provider      5. An independent contractor or businessperson
working at a school site.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStaffsAll",
          "type": "array",
          "items": {
            "$ref": "staff"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "staff"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStaffsByExample",
          "type": "array",
          "items": {
            "$ref": "staff"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "staffUniqueId",
              "description": "A unique alphanumeric code assigned to a staff.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "staff"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStaffByKey",
          "type": "staff",
          "parameters": [
            {
              "paramType": "query",
              "name": "staffUniqueId",
              "description": "A unique alphanumeric code assigned to a staff.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "staff"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postStaffs",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "staff",
              "description": "The JSON representation of the \"staff\" resource to be created or updated.",
              "type": "staff",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
```

```
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/staffs/{id}",
      "description": "This entity represents an individual who performs specified activities for any public or
private education institution or agency that provides instructional and/or support services to students or
staff at the early childhood level through high school completion. For example, this includes:      1. An \"
employee\" who performs services under the direction of the employing institution or agency is compensated for
such services by the employer and is eligible for employee benefits and wage or salary tax withholdings      2.
A \"contractor\" or \"consultant\" who performs services for an agreed upon fee or an employee of a management
service contracted to work on site      3. A \"volunteer\" who performs services on a voluntary and
uncompensated basis      4. An in-kind service provider      5. An independent contractor or businessperson
working at a school site.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStaffsById",
          "type": "staff",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "staff"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putStaff",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "staff",
              "description": "The JSON representation of the \"staff\" resource to be updated.",
              "type": "staff",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteStaffById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "staff": {
      "id": "staff",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "birthDate": {
          "type": "date-time",
          "required": true,
          "description": "The month, day, and year on which an individual was born."
        },
        "citizenshipStatusType": {
          "type": "string",
          "required": false,
          "description": "An indicator of whether or not the person is a U.S. citizen."
        },
        "firstName": {
          "type": "string",
          "required": true,
          "description": "A name given to an individual at birth, baptism, or during another naming ceremony,
or through legal change."
        },
        "generationCodeSuffix": {
          "type": "string",
          "required": true,
          "description": "An appendage, if any, used to denote an individual's generation in his family (e.g.,
Jr., Sr., III)."
        },
        "highestCompletedLevelOfEducationDescriptor": {
          "type": "string",
          "required": true,
          "description": "The extent of formal instruction an individual has received (e.g., the highest grade
in school completed or its equivalent or the highest degree received)."
        },
        "highlyQualifiedTeacher": {
          "type": "boolean",
          "required": true,
          "description": "An indication of whether a teacher is classified as highly qualified for his/her
assignment according to state definition. This attribute indicates the teacher is highly qualified for ALL
Sections being taught."
        },
        "hispanicLatinoEthnicity": {
          "type": "boolean",
          "required": true,
          "description": "An indication that the individual traces his or her origin or descent to Mexico,
Puerto Rico, Cuba, Central, and South America, and other Spanish cultures, regardless of race. The term, \"
Spanish origin,\" can be used in addition to \"Hispanic or Latino.\""
        },
        "lastSurname": {
          "type": "string",
          "required": true,
          "description": "The name borne in common by members of a family."
        },
        "loginId": {
          "type": "string",
          "required": false,
          "description": "The login ID for the user; used for security access control interface."
        },
        "maidenName": {
          "type": "string",
          "required": false,
          "description": "The person's maiden name."
        },
        "middleName": {
          "type": "string",
          "required": true,
          "description": "A secondary name given to an individual at birth, baptism, or during another naming
ceremony."
        },
        "oldEthnicityType": {
          "type": "string",
          "required": false,
          "description": "Previous definition of Ethnicity combining Hispanic/Latino and race:          1 -
American Indian or Alaskan Native          2 - Asian or Pacific Islander          3 - Black, not of Hispanic
origin          4 - Hispanic          5 - White, not of Hispanic origin."
        },
        "personalTitlePrefix": {
          "type": "string",
          "required": true,
          "description": "A prefix used to denote the title, degree, position, or seniority of the person."
        },
        "sexType": {
          "type": "string",
          "required": true,
          "description": "A person's gender."
        },
        "staffUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a staff."
        },
        "yearsOfPriorProfessionalExperience": {
          "type": "number",
          "required": false,
          "description": "The total number of years that an individual has previously held a similar
professional position in one or more education institutions."
        },
        "yearsOfPriorTeachingExperience": {
          "type": "number",
          "required": false,
          "description": "The total number of years that an individual has previously held a teaching position
in one or more education institutions."
        },
        "addresses": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of staffAddresses.  The set of elements that describes an
address, including the street address, city, state, and ZIP code.",
          "items": {
            "$ref": "staffAddress"
          }
        },
        "credentials": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of staffCredentials.  The legal document or authorization
giving authorization to perform teaching assignment services.",
          "items": {
            "$ref": "staffCredential"
          }
        },
        "electronicMails": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of staffElectronicMails.  The numbers, letters, and symbols
used to identify an electronic mail (e-mail) user within the network to which the individual or organization
belongs.",
          "items": {
            "$ref": "staffElectronicMail"
          }
        },
        "identificationCodes": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of staffIdentificationCodes.  A unique number or alphanumeric
code assigned to a staff member by a school, school system, a state, or other agency or entity.",
          "items": {
            "$ref": "staffIdentificationCode"
          }
        },
        "identificationDocuments": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of staffIdentificationDocuments.  The documents presented as
evident to verify one's personal identity; for example: drivers license, passport, birth certificate, etc.",
          "items": {
            "$ref": "staffIdentificationDocument"
          }
        },
        "internationalAddresses": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of staffInternationalAddresses.  The set of elements that
describes an international address.",
          "items": {
            "$ref": "staffInternationalAddress"
          }
        },
        "languages": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of staffLanguages.  The language(s) the individual uses to
communicate.",
          "items": {
            "$ref": "staffLanguage"
          }
        },
        "otherNames": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of staffOtherNames.  Other names (e.g., alias, nickname,
previous legal name) associated with a person.",
          "items": {
            "$ref": "staffOtherName"
          }
        },
        "races": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of staffRaces.  The general racial category which most
clearly reflects the individual's recognition of his or her community or with which the individual most
identifies. The way this data element is listed, it must allow for multiple entries so that each individual can
specify all appropriate races.",
          "items": {
            "$ref": "staffRace"
          }
        },
        "telephones": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of staffTelephones.  The 10-digit telephone number, including
the area code, for the person.",
          "items": {
            "$ref": "staffTelephone"
          }
        },
        "visas": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of staffVisas.  An indicator of a non-US citizen's Visa
type.",
          "items": {
            "$ref": "staffVisa"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
```

```
    },
    "staffAddress": {
      "id": "staffAddress",
      "properties": {
        "addressType": {
          "type": "string",
          "required": true,
          "description": "The type of address listed for an individual or organization.    For example:
Physical Address, Mailing Address, Home Address, etc.)"
        },
        "stateAbbreviationType": {
          "type": "string",
          "required": true,
          "description": "The abbreviation for the state (within the United States) or outlying area in which
an address is located."
        },
        "streetNumberName": {
          "type": "string",
          "required": true,
          "description": "The street number and street name or post office box number of an address."
        },
        "apartmentRoomSuiteNumber": {
          "type": "string",
          "required": false,
          "description": "The apartment, room, or suite number of an address."
        },
        "buildingSiteNumber": {
          "type": "string",
          "required": false,
          "description": "The number of the building on the site, if more than one building shares the same
address."
        },
        "city": {
          "type": "string",
          "required": true,
          "description": "The name of the city in which an address is located."
        },
        "postalCode": {
          "type": "string",
          "required": true,
          "description": "The five or nine digit zip code or overseas postal code portion of an address."
        },
        "nameOfCounty": {
          "type": "string",
          "required": false,
          "description": "The name of the county, parish, borough, or comparable unit (within a state)
in                         'which an address is located."
        },
        "countyFIPSCode": {
          "type": "string",
          "required": false,
          "description": "The Federal Information Processing Standards (FIPS) numeric code for the county
issued by the National Institute of Standards and Technology (NIST). Counties are considered to be the \"first-
order subdivisions\" of each State and statistically equivalent entity, regardless of their local designations
(county, parish, borough, etc.) Counties in different States will have the same code. A unique county number is
created when combined with the 2-digit FIPS State Code."
        },
        "latitude": {
          "type": "string",
          "required": false,
          "description": "The geographic latitude of the physical address."
        },
        "longitude": {
          "type": "string",
          "required": false,
          "description": "The geographic longitude of the physical address."
        },
        "beginDate": {
          "type": "date-time",
          "required": false,
          "description": "The first date the address is valid. For physical addresses, the date the person
moved to that address."
        },
        "endDate": {
          "type": "date-time",
          "required": false,
          "description": "The last date the address is valid. For physical addresses, this would be the date
the person moved from that address."
        }
      }
    },
    "staffCredential": {
      "id": "staffCredential",
      "properties": {
        "credentialFieldDescriptor": {
          "type": "string",
          "required": true,
          "description": "The field of certification for the certificate (e.g., Mathematics, Music)."
        },
        "credentialType": {
          "type": "string",
          "required": true,
          "description": "An indication of the category of credential an individual holds."
        },
        "levelDescriptor": {
          "type": "string",
          "required": true,
          "description": "The grade level(s) certified for teaching."
        },
        "teachingCredentialDescriptor": {
          "type": "string",
          "required": true,
          "description": "An indication of the category of a legal document giving authorization to perform
teaching assignment services."
        },
        "credentialIssuanceDate": {
          "type": "date-time",
          "required": true,
          "description": "The month, day, and year on which an active credential was issued to an individual."
        },
        "stateOfIssueStateAbbreviationType": {
          "type": "string",
          "required": false,
          "description": "The abbreviation for the name of the state (within the United States) or extra-state
jurisdiction in which a license/credential was issued."
        },
        "teachingCredentialBasisType": {
          "type": "string",
          "required": false,
          "description": "An indication of the pre-determined criteria for granting the teaching credential
that an individual holds."
        },
        "credentialExpirationDate": {
          "type": "date-time",
          "required": false,
          "description": "The month, day, and year on which an active credential held by an individual will
expire."
        }
      }
    },
    "staffElectronicMail": {
      "id": "staffElectronicMail",
      "properties": {
        "electronicMailType": {
          "type": "string",
          "required": true,
          "description": "The type of email listed for an individual or organization. For example: Home
/Personal, Work, etc.)"
        },
        "electronicMailAddress": {
          "type": "string",
          "required": true,
          "description": "The electronic mail (e-mail) address listed for an individual or organization."
        },
        "primaryEmailAddressIndicator": {
          "type": "boolean",
          "required": false,
          "description": "An indication that the electronic mail address should be used as the principal
electronic mail address for an individual or organization."
        }
      }
    },
    "staffIdentificationCode": {
      "id": "staffIdentificationCode",
      "properties": {
        "staffIdentificationSystemDescriptor": {
          "type": "string",
          "required": true,
          "description": "A coding scheme that is used for identification and record-keeping purposes by
schools, social services, or other agencies to refer to a staff member."
        },
        "identificationCode": {
          "type": "string",
          "required": true,
          "description": "A unique number or alphanumeric code assigned to a staff member by a school, school
system, a state, or other agency or entity."
        },
        "assigningOrganizationIdentificationCode": {
          "type": "string",
          "required": false,
          "description": "The organization code or name assigning the staff Identification Code."
        }
      }
    },
    "staffIdentificationDocument": {
      "id": "staffIdentificationDocument",
      "properties": {
        "identificationDocumentUseType": {
          "type": "string",
          "required": true,
          "description": "The primary function of the document used for establishing identity."
        },
        "personalInformationVerificationType": {
          "type": "string",
          "required": true,
          "description": "The category of the document relative to its purpose."
        },
        "issuerCountryDescriptor": {
          "type": "string",
          "required": false,
          "description": "Country of origin of the document."
        },
        "documentTitle": {
          "type": "string",
          "required": false,
          "description": "The title of the document given by the issuer."
        },
        "documentExpirationDate": {
          "type": "date-time",
          "required": false,
          "description": "The day when the document  expires, if null then never expires."
        },
        "issuerDocumentIdentificationCode": {
          "type": "string",
          "required": false,
          "description": "The unique identifier on the issuer's identification system."
        },
        "issuerName": {
          "type": "string",
          "required": false,
          "description": "Name of the entity or institution that issued the document."
        }
      }
    },
    "staffInternationalAddress": {
      "id": "staffInternationalAddress",
      "properties": {
        "addressType": {
          "type": "string",
          "required": true,
          "description": "The type of address listed for an individual or organization. For example:  Physical
Address, Mailing Address, Home Address, etc.)"
        },
        "countryDescriptor": {
          "type": "string",
          "required": true,
          "description": "The name of the country."
        },
        "addressLine1": {
          "type": "string",
          "required": true,
          "description": "The first line of the address."
        },
        "addressLine2": {
          "type": "string",
          "required": false,
          "description": "The second line of the address."
        },
        "addressLine3": {
          "type": "string",
          "required": false,
          "description": "The third line of the address."
        },
        "addressLine4": {
          "type": "string",
          "required": false,
          "description": "The fourth line of the address."
        },
        "latitude": {
          "type": "string",
          "required": false,
          "description": "The geographic latitude of the physical address."
        },
        "longitude": {
          "type": "string",
          "required": false,
          "description": "The geographic longitude of the physical address."
        },
        "beginDate": {
          "type": "date-time",
          "required": false,
          "description": "The first date the address is valid. For physical addresses, the date the person
moved to that address."
        },
        "endDate": {
          "type": "date-time",
          "required": false,
          "description": "The last date the address is valid. For physical addresses, this would be the date
the person moved from that address."
        }
      }
    },
    "staffLanguage": {
      "id": "staffLanguage",
      "properties": {
        "languageDescriptor": {
          "type": "string",
          "required": true,
          "description": "A specification of which written or spoken communication is being used."
        },
        "uses": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of staffLanguageUses.  A description of how the language is
```

```
used (e.g. Home Language, Native Language, Spoken Language).",
          "items": {
            "$ref": "staffLanguageUse"
          }
        }
      }
    },
    "staffOtherName": {
      "id": "staffOtherName",
      "properties": {
        "otherNameType": {
          "type": "string",
          "required": true,
          "description": "The types of alternate names for a person."
        },
        "personalTitlePrefix": {
          "type": "string",
          "required": false,
          "description": "A prefix used to denote the title, degree, position, or seniority of the person."
        },
        "firstName": {
          "type": "string",
          "required": true,
          "description": "A name given to an individual at birth, baptism, or during another naming ceremony,
or through legal change."
        },
        "middleName": {
          "type": "string",
          "required": false,
          "description": "A secondary name given to an individual at birth, baptism, or during another naming
ceremony."
        },
        "lastSurname": {
          "type": "string",
          "required": true,
          "description": "The name borne in common by members of a family."
        },
        "generationCodeSuffix": {
          "type": "string",
          "required": false,
          "description": "An appendage, if any, used to denote an individual's generation in his family (e.g.,
Jr., Sr., III)."
        }
      }
    },
    "staffRace": {
      "id": "staffRace",
      "properties": {
        "raceType": {
          "type": "string",
          "required": true,
          "description": "The general racial category which most clearly reflects the individual's recognition
of his or her community or with which the individual most identifies. The way this data element is listed, it
must allow for multiple entries so that each individual can specify all appropriate races."
        }
      }
    },
    "staffTelephone": {
      "id": "staffTelephone",
      "properties": {
        "telephoneNumberType": {
          "type": "string",
          "required": true,
          "description": "The type of communication number listed for an individual or organization."
        },
        "telephoneNumber": {
          "type": "string",
          "required": true,
          "description": "The telephone number including the area code, and extension, if applicable."
        },
        "orderOfPriority": {
          "type": "integer",
          "required": false,
          "description": "The order of priority assigned to telephone numbers to define which number to attempt
first, second, etc."
        },
        "textMessageCapabilityIndicator": {
          "type": "boolean",
          "required": false,
          "description": "An indication that the telephone number is technically capable of sending and
receiving Short Message Service (SMS) text messages."
        }
      }
    },
    "staffVisa": {
      "id": "staffVisa",
      "properties": {
        "visaType": {
          "type": "string",
          "required": true,
          "description": "An indicator of a non-US citizen's Visa type."
        }
      }
    },
    "staffLanguageUse": {
      "id": "staffLanguageUse",
      "properties": {
        "languageUseType": {
          "type": "string",
          "required": true,
          "description": "A description of how the language is used (e.g. Home Language, Native Language,
Spoken Language)."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/staffCohortAssociations

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/staffCohortAssociations",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/staffCohortAssociations",
      "description": "This association indicates the Staff associated with a cohort of students.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStaffCohortAssociationsAll",
          "type": "array",
          "items": {
            "$ref": "staffCohortAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "staffCohortAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStaffCohortAssociationsByExample",
          "type": "array",
          "items": {
            "$ref": "staffCohortAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "beginDate",
              "description": "Start date for the association of staff to this cohort.",
              "type": "date-time",
              "required": false
            },
            {
              "paramType": "query",
              "name": "cohortIdentifier",
              "description": "The name or ID for the Cohort.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "staffUniqueId",
              "description": "A unique alphanumeric code assigned to a staff.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "staffCohortAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStaffCohortAssociationByKey",
          "type": "staffCohortAssociation",
          "parameters": [
            {
              "paramType": "query",
              "name": "beginDate",
              "description": "Start date for the association of staff to this cohort.",
              "type": "date-time",
              "required": true
            },
            {
              "paramType": "query",
              "name": "cohortIdentifier",
              "description": "The name or ID for the Cohort.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "staffUniqueId",
              "description": "A unique alphanumeric code assigned to a staff.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "staffCohortAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postStaffCohortAssociations",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "staffCohortAssociation",
              "description": "The JSON representation of the \"staffCohortAssociation\" resource to be created
or updated.",
              "type": "staffCohortAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/staffCohortAssociations/{id}",
      "description": "This association indicates the Staff associated with a cohort of students.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStaffCohortAssociationsById",
          "type": "staffCohortAssociation",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "staffCohortAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putStaffCohortAssociation",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "staffCohortAssociation",
              "description": "The JSON representation of the \"staffCohortAssociation\" resource to be
updated.",
              "type": "staffCohortAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteStaffCohortAssociationById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
```

```
  ],
  "models": {
    "staffCohortAssociation": {
      "id": "staffCohortAssociation",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "cohortReference": {
          "type": "cohortReference",
          "required": true,
          "description": "A reference to the related Cohort resource."
        },
        "staffReference": {
          "type": "staffReference",
          "required": true,
          "description": "A reference to the related Staff resource."
        },
        "beginDate": {
          "type": "date-time",
          "required": true,
          "description": "Start date for the association of staff to this cohort."
        },
        "endDate": {
          "type": "date-time",
          "required": true,
          "description": "End date for the association of staff to this cohort."
        },
        "studentRecordAccess": {
          "type": "boolean",
          "required": false,
          "description": "Indicator of whether the staff has access to the student records of the cohort per
district interpretation of FERPA and other privacy laws, regulations, and policies."
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "cohortReference": {
      "id": "cohortReference",
      "properties": {
        "identifier": {
          "type": "string",
          "required": true,
          "description": "The name or ID for the Cohort."
        },
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        }
      }
    },
    "staffReference": {
      "id": "staffReference",
      "properties": {
        "staffUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a staff."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/staffEducationOrganizationAssignmentAssociations

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/staffEducationOrganizationAssignmentAssociations",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/staffEducationOrganizationAssignmentAssociations",
      "description": "This association indicates the education organization to which a staff member provides
services; also known as school of service.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStaffEducationOrganizationAssignmentAssociationsAll",
          "type": "array",
          "items": {
            "$ref": "staffEducationOrganizationAssignmentAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "staffEducationOrganizationAssignmentAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStaffEducationOrganizationAssignmentAssociationsByExample",
          "type": "array",
          "items": {
            "$ref": "staffEducationOrganizationAssignmentAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "beginDate",
              "description": "Month, day, and year of the start or effective date of a staff member's
employment, contract, or relationship with the LEA.",
              "type": "date-time",
              "required": false
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "staffClassificationDescriptor",
              "description": "The titles of employment, official status, or rank of education staff.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "staffUniqueId",
              "description": "A unique alphanumeric code assigned to a staff.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "staffEducationOrganizationAssignmentAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStaffEducationOrganizationAssignmentAssociationByKey",
          "type": "staffEducationOrganizationAssignmentAssociation",
          "parameters": [
            {
              "paramType": "query",
              "name": "beginDate",
              "description": "Month, day, and year of the start or effective date of a staff member's
employment, contract, or relationship with the LEA.",
              "type": "date-time",
              "required": true
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "staffClassificationDescriptor",
              "description": "The titles of employment, official status, or rank of education staff.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "staffUniqueId",
              "description": "A unique alphanumeric code assigned to a staff.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "staffEducationOrganizationAssignmentAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
```

```
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postStaffEducationOrganizationAssignmentAssociations",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "staffEducationOrganizationAssignmentAssociation",
              "description": "The JSON representation of the \"
staffEducationOrganizationAssignmentAssociation\" resource to be created or updated.",
              "type": "staffEducationOrganizationAssignmentAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/staffEducationOrganizationAssignmentAssociations/{id}",
      "description": "This association indicates the education organization to which a staff member provides
services; also known as school of service.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStaffEducationOrganizationAssignmentAssociationsById",
          "type": "staffEducationOrganizationAssignmentAssociation",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "staffEducationOrganizationAssignmentAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putStaffEducationOrganizationAssignmentAssociation",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "staffEducationOrganizationAssignmentAssociation",
              "description": "The JSON representation of the \"
staffEducationOrganizationAssignmentAssociation\" resource to be updated.",
              "type": "staffEducationOrganizationAssignmentAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteStaffEducationOrganizationAssignmentAssociationById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "staffEducationOrganizationAssignmentAssociation": {
      "id": "staffEducationOrganizationAssignmentAssociation",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "educationOrganizationReference": {
          "type": "educationOrganizationReference",
          "required": true,
          "description": "A reference to the related EducationOrganization resource."
        },
        "staffReference": {
          "type": "staffReference",
          "required": true,
          "description": "A reference to the related Staff resource."
        },
        "employmentStaffEducationOrganizationEmploymentAssociationReference": {
          "type": "staffEducationOrganizationEmploymentAssociationReference",
          "required": false,
          "description": "A reference to the related StaffEducationOrganizationEmploymentAssociation resource."
        },
        "beginDate": {
          "type": "date-time",
          "required": true,
          "description": "Month, day, and year of the start or effective date of a staff member's employment,
contract, or relationship with the LEA."
        },
        "endDate": {
          "type": "date-time",
          "required": true,
          "description": "Month, day, and year of the end or termination date of a staff member's employment,
contract, or relationship with the LEA."
        },
        "orderOfAssignment": {
          "type": "integer",
          "required": false,
          "description": "Describes whether the assignment is this the staff member's primary assignment,
secondary assignment, etc."
        },
        "positionTitle": {
          "type": "string",
          "required": true,
          "description": "The descriptive name of an individual's position."
        },
        "staffClassificationDescriptor": {
          "type": "string",
          "required": true,
          "description": "The titles of employment, official status, or rank of education staff."
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "educationOrganizationReference": {
      "id": "educationOrganizationReference",
      "properties": {
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        }
      }
    },
    "staffReference": {
      "id": "staffReference",
      "properties": {
        "staffUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a staff."
        }
      }
    },
    "staffEducationOrganizationEmploymentAssociationReference": {
      "id": "staffEducationOrganizationEmploymentAssociationReference",
      "properties": {
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        },
        "staffUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a staff."
        },
        "employmentStatusDescriptor": {
          "type": "string",
          "required": true,
          "description": "Reflects the type of employment or contract; for example:
Probationary          Contractual          Substitute/temporary          Tenured or permanent          Volunteer
/no contract          ..."
        },
        "hireDate": {
          "type": "date-time",
          "required": true,
          "description": "The month, day, and year on which an individual was hired for a position."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/staffSchoolAssociations

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/staffSchoolAssociations",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/staffSchoolAssociations",
      "description": "This association indicates the School(s) to which a staff member provides instructional
services.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStaffSchoolAssociationsAll",
          "type": "array",
          "items": {
            "$ref": "staffSchoolAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "staffSchoolAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStaffSchoolAssociationsByExample",
          "type": "array",
          "items": {
            "$ref": "staffSchoolAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "programAssignmentDescriptor",
              "description": "The name of the program for which the individual is assigned; for
example:          Regular education          Title I-Academic          Title I-Non-Academic          Special
Education          Bilingual/English as a Second Language.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "staffUniqueId",
              "description": "A unique alphanumeric code assigned to a staff.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "staffSchoolAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStaffSchoolAssociationByKey",
          "type": "staffSchoolAssociation",
          "parameters": [
            {
              "paramType": "query",
              "name": "programAssignmentDescriptor",
              "description": "The name of the program for which the individual is assigned; for
example:          Regular education          Title I-Academic          Title I-Non-Academic          Special
Education          Bilingual/English as a Second Language.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "staffUniqueId",
              "description": "A unique alphanumeric code assigned to a staff.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "staffSchoolAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postStaffSchoolAssociations",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "staffSchoolAssociation",
              "description": "The JSON representation of the \"staffSchoolAssociation\" resource to be created
or updated.",
              "type": "staffSchoolAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
```

```
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/staffSchoolAssociations/{id}",
      "description": "This association indicates the School(s) to which a staff member provides instructional
services.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStaffSchoolAssociationsById",
          "type": "staffSchoolAssociation",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "staffSchoolAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putStaffSchoolAssociation",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "staffSchoolAssociation",
              "description": "The JSON representation of the \"staffSchoolAssociation\" resource to be
updated.",
              "type": "staffSchoolAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteStaffSchoolAssociationById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "staffSchoolAssociation": {
      "id": "staffSchoolAssociation",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "schoolReference": {
          "type": "schoolReference",
          "required": true,
          "description": "A reference to the related School resource."
        },
        "schoolYearTypeReference": {
          "type": "schoolYearTypeReference",
          "required": false,
          "description": "A reference to the related SchoolYearType resource."
        },
        "staffReference": {
          "type": "staffReference",
          "required": true,
          "description": "A reference to the related Staff resource."
        },
        "programAssignmentDescriptor": {
          "type": "string",
          "required": true,
          "description": "The name of the program for which the individual is assigned; for example:
Regular education          Title I-Academic          Title I-Non-Academic          Special Education
Bilingual/English as a Second Language."
        },
        "academicSubjects": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of staffSchoolAssociationAcademicSubjects.  The teaching
field taught by an individual, for example English/Language Arts, Reading, Mathematics, Science, Social
Sciences, etc.",
          "items": {
            "$ref": "staffSchoolAssociationAcademicSubject"
          }
        },
        "gradeLevels": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of staffSchoolAssociationGradeLevels.  The set of grade
levels for which the individual's assignment is responsible.",
          "items": {
            "$ref": "staffSchoolAssociationGradeLevel"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "schoolReference": {
      "id": "schoolReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        }
      }
    },
    "schoolYearTypeReference": {
      "id": "schoolYearTypeReference",
      "properties": {
        "schoolYear": {
          "type": "integer",
          "required": true,
          "description": "Key for School Year"
        }
      }
    },
    "staffReference": {
      "id": "staffReference",
      "properties": {
        "staffUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a staff."
        }
      }
    },
    "staffSchoolAssociationAcademicSubject": {
      "id": "staffSchoolAssociationAcademicSubject",
      "properties": {
        "academicSubjectDescriptor": {
          "type": "string",
          "required": true,
          "description": "The teaching field taught by an individual, for example English/Language Arts,
Reading, Mathematics, Science, Social Sciences, etc."
        }
      }
    },
    "staffSchoolAssociationGradeLevel": {
      "id": "staffSchoolAssociationGradeLevel",
      "properties": {
        "gradeLevelDescriptor": {
          "type": "string",
          "required": true,
          "description": "The set of grade levels for which the individual's assignment is responsible."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/staffSectionAssociations

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/staffSectionAssociations",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/staffSectionAssociations",
      "description": "This association indicates the class sections to which a staff member is assigned.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStaffSectionAssociationsAll",
          "type": "array",
          "items": {
            "$ref": "staffSectionAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "staffSectionAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStaffSectionAssociationsByExample",
          "type": "array",
          "items": {
            "$ref": "staffSectionAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "classPeriodName",
              "description": "An indication of the portion of a typical daily session in which students receive
instruction in a specified subject (e.g., morning, sixth period, block period, or AB schedules).",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "classroomIdentificationCode",
              "description": "A unique number or alphanumeric code assigned to a room by a school, school
system, state, or other agency or entity.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "localCourseCode",
              "description": "The local code assigned by the School that identifies the course offering
provided for the instruction of students.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolYear",
              "description": "The identifier for the school year.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "sequenceOfCourse",
              "description": "When a section is part of a sequence of parts for a course, the number of the
sequence. If the course has only one part, the value of this section attribute should be 1.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "staffUniqueId",
              "description": "A unique alphanumeric code assigned to a staff.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "termDescriptor",
              "description": "The term for the Session during the school year.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "uniqueSectionCode",
              "description": "A unique identifier for the Section that is defined by the classroom, the
subjects taught, and the instructors who are assigned.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "staffSectionAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStaffSectionAssociationByKey",
          "type": "staffSectionAssociation",
          "parameters": [
            {
              "paramType": "query",
              "name": "classPeriodName",
              "description": "An indication of the portion of a typical daily session in which students receive
instruction in a specified subject (e.g., morning, sixth period, block period, or AB schedules).",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "classroomIdentificationCode",
              "description": "A unique number or alphanumeric code assigned to a room by a school, school
system, state, or other agency or entity.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "localCourseCode",
              "description": "The local code assigned by the School that identifies the course offering
provided for the instruction of students.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolYear",
              "description": "The identifier for the school year.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "sequenceOfCourse",
              "description": "When a section is part of a sequence of parts for a course, the number of the
sequence. If the course has only one part, the value of this section attribute should be 1.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "staffUniqueId",
              "description": "A unique alphanumeric code assigned to a staff.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "termDescriptor",
              "description": "The term for the Session during the school year.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "uniqueSectionCode",
              "description": "A unique identifier for the Section that is defined by the classroom, the
subjects taught, and the instructors who are assigned.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "staffSectionAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
```

```
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postStaffSectionAssociations",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "staffSectionAssociation",
              "description": "The JSON representation of the \"staffSectionAssociation\" resource to be created
or updated.",
              "type": "staffSectionAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/staffSectionAssociations/{id}",
      "description": "This association indicates the class sections to which a staff member is assigned.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStaffSectionAssociationsById",
          "type": "staffSectionAssociation",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "staffSectionAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putStaffSectionAssociation",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "staffSectionAssociation",
              "description": "The JSON representation of the \"staffSectionAssociation\" resource to be
updated.",
              "type": "staffSectionAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteStaffSectionAssociationById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
```

```
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "staffSectionAssociation": {
      "id": "staffSectionAssociation",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "sectionReference": {
          "type": "sectionReference",
          "required": true,
          "description": "A reference to the related Section resource."
        },
        "staffReference": {
          "type": "staffReference",
          "required": true,
          "description": "A reference to the related Staff resource."
        },
        "beginDate": {
          "type": "date-time",
          "required": true,
          "description": "Month, day, and year of a teacher's assignment to the Section. If blank, defaults to
the first day of the first grading period for the Section."
        },
        "classroomPositionDescriptor": {
          "type": "string",
          "required": true,
          "description": "The type of position the Staff member holds in the specific class/section; for
example:          Teacher of Record, Assistant Teacher, Support Teacher, Substitute Teacher..."
        },
        "endDate": {
          "type": "date-time",
          "required": false,
          "description": "Month, day, and year of the last day of a staff member's assignment to the Section."
        },
        "highlyQualifiedTeacher": {
          "type": "boolean",
          "required": false,
          "description": "An indication of whether a teacher is classified as highly qualified for his/her
assignment according to state definition. This attribute indicates the teacher is highly qualified for this
section being taught."
        },
        "percentageContribution": {
          "type": "number",
          "required": false,
          "description": "Indicates the percentage of the total scheduled course time, academic standards, and
/or learning activities delivered in this section by this staff member. A teacher of record designation may be
based solely or partially on this contribution percentage."
        },
        "teacherStudentDataLinkExclusion": {
          "type": "boolean",
          "required": false,
          "description": "Indicates that the entire section is excluded from calculation of value-added or
growth attribution calculations used for a particular teacher evaluation."
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "sectionReference": {
      "id": "sectionReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        },
        "classPeriodName": {
          "type": "string",
          "required": true,
          "description": "An indication of the portion of a typical daily session in which students receive
instruction in a specified subject (e.g., morning, sixth period, block period, or AB schedules)."
        },
        "classroomIdentificationCode": {
          "type": "string",
          "required": true,
          "description": "A unique number or alphanumeric code assigned to a room by a school, school system,
state, or other agency or entity."
        },
        "localCourseCode": {
          "type": "string",
          "required": true,
          "description": "The local code assigned by the School that identifies the course offering provided
for the instruction of students."
        },
        "termDescriptor": {
          "type": "string",
          "required": true,
          "description": "The term for the Session during the school year."
        },
        "schoolYear": {
          "type": "integer",
          "required": true,
          "description": "The identifier for the school year."
        },
        "uniqueSectionCode": {
          "type": "string",
          "required": true,
          "description": "A unique identifier for the Section that is defined by the classroom, the subjects
taught, and the instructors who are assigned."
        },
        "sequenceOfCourse": {
          "type": "integer",
          "required": true,
          "description": "When a section is part of a sequence of parts for a course, the number of the
sequence. If the course has only one part, the value of this section attribute should be 1."
        }
      }
    },
    "staffReference": {
      "id": "staffReference",
      "properties": {
        "staffUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a staff."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/students

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/students",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/students",
      "description": "This entity represents an individual for whom instruction, services, and/or care are
```

```
provided in an early childhood, elementary, or secondary educational program under the jurisdiction of a
school, education agency or other institution or program. A student is a person who has been enrolled in a
school or other educational institution.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentsAll",
          "type": "array",
          "items": {
            "$ref": "student"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "student"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentsByExample",
          "type": "array",
          "items": {
            "$ref": "student"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "student"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
```

```
        },
        {
          "method": "GET",
          "nickname": "getStudentByKey",
          "type": "student",
          "parameters": [
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "student"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postStudents",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "student",
              "description": "The JSON representation of the \"student\" resource to be created or updated.",
              "type": "student",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/students/{id}",
      "description": "This entity represents an individual for whom instruction, services, and/or care are
provided in an early childhood, elementary, or secondary educational program under the jurisdiction of a
school, education agency or other institution or program. A student is a person who has been enrolled in a
school or other educational institution.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentsById",
          "type": "student",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "student"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putStudent",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "student",
              "description": "The JSON representation of the \"student\" resource to be updated.",
              "type": "student",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteStudentById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "student": {
      "id": "student",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "birthCity": {
          "type": "string",
          "required": true,
          "description": "The city the student was born in."
        },
        "birthCountryDescriptor": {
          "type": "string",
          "required": true,
          "description": "The country in which an individual is born."
        },
        "birthDate": {
          "type": "date-time",
          "required": true,
          "description": "The month, day, and year on which an individual was born."
        },
        "birthInternationalProvince": {
          "type": "string",
          "required": false,
          "description": "For students born outside of the U.S., the Province or jurisdiction in which an
individual is born."
        },
        "birthStateAbbreviationType": {
          "type": "string",
          "required": true,
          "description": "The abbreviation for the name of the state (within the United States) or extra-state
jurisdiction in which an individual was born."
        },
        "citizenshipStatusType": {
          "type": "string",
          "required": false,
          "description": "An indicator of whether or not the person is a U.S. citizen."
        },
        "dateEnteredUS": {
          "type": "date-time",
          "required": true,
          "description": "For students born outside of the U.S., the date the student entered the U.S."
        },
        "displacementStatus": {
          "type": "string",
          "required": false,
          "description": "Indicates a state health or weather related event that displaces a group of students,
and may require additional funding, educational, or social services."
        },
        "economicDisadvantaged": {
          "type": "boolean",
          "required": true,
          "description": "An indication of inadequate financial condition of an individual's family, as
determined by family income, number of family members/dependents, participation in public assistance programs,
and/or other characteristics considered relevant by federal, state, and local policy."
        },
        "firstName": {
          "type": "string",
          "required": true,
          "description": "A name given to an individual at birth, baptism, or during another naming ceremony,
or through legal change."
        },
        "generationCodeSuffix": {
          "type": "string",
          "required": true,
          "description": "An appendage, if any, used to denote an individual's generation in his family (e.g.,
Jr., Sr., III)."
        },
        "hispanicLatinoEthnicity": {
          "type": "boolean",
          "required": true,
          "description": "An indication that the individual traces his or her origin or descent to Mexico,
Puerto Rico, Cuba, Central, and South America, and other Spanish cultures, regardless of race. The term, \"
Spanish origin,\" can be used in addition to \"Hispanic or Latino.\""
        },
        "lastSurname": {
          "type": "string",
          "required": true,
          "description": "The name borne in common by members of a family."
        },
        "limitedEnglishProficiencyDescriptor": {
          "type": "string",
          "required": true,
          "description": "An indication that the student has been identified as limited English proficient by
the Language Proficiency Assessment Committee (LPAC), or English proficient."
        },
        "loginId": {
          "type": "string",
          "required": false,
          "description": "The login ID for the user; used for security access control interface."
        },
        "maidenName": {
          "type": "string",
          "required": false,
          "description": "The person's maiden name."
        },
        "middleName": {
          "type": "string",
          "required": true,
          "description": "A secondary name given to an individual at birth, baptism, or during another naming
ceremony."
        },
        "multipleBirthStatus": {
          "type": "boolean",
          "required": false,
          "description": "Indicator of whether the student was born with other siblings (i.e., twins, triplets,
etc.)"
        },
        "oldEthnicityType": {
          "type": "string",
          "required": false,
          "description": "Previous definition of Ethnicity combining Hispanic/Latino and race:          1 -
American Indian or Alaskan Native          2 - Asian or Pacific Islander          3 - Black, not of Hispanic
origin          4 - Hispanic          5 - White, not of Hispanic origin."
        },
        "personalTitlePrefix": {
          "type": "string",
          "required": true,
          "description": "A prefix used to denote the title, degree, position, or seniority of the person."
        },
        "profileThumbnail": {
          "type": "string",
          "required": false,
          "description": "Locator for the student photo."
        },
        "schoolFoodServicesEligibilityDescriptor": {
          "type": "string",
          "required": true,
          "description": "An indication of a student's level of eligibility for breakfast, lunch, snack,
supper, and milk programs."
        },
        "sexType": {
          "type": "string",
          "required": true,
          "description": "A person's gender."
        },
        "studentUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a student."
        },
        "learningStyle": {
          "type": "studentLearningStyle",
          "required": false,
          "description": "The student's relative preference to visual, auditory, and tactile learning expressed
as percentages."
        },
        "addresses": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of studentAddresses.  The set of elements that describes an
address, including the street address, city, state, and ZIP code.",
          "items": {
            "$ref": "studentAddress"
          }
        },
        "characteristics": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of studentCharacteristics.  Reflects important
characteristics of the student's home situation:          Displaced Homemaker, Immigrant, Migratory, Military
Parent, Pregnant Teen, Single Parent, and Unaccompanied Youth.",
          "items": {
            "$ref": "studentCharacteristic"
          }
        },
        "cohortYears": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of studentCohortYears.  The type and year of a cohort (e.g.,
9th grade) the student belongs to as determined by the year that student entered a specific grade.",
          "items": {
            "$ref": "studentCohortYear"
          }
        },
        "disabilities": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of studentDisabilities.  The disability condition(s) that
best describes an individual's impairment.",
          "items": {
            "$ref": "studentDisability"
          }
        },
        "electronicMails": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of studentElectronicMails.  The numbers, letters, and symbols
used to identify an electronic mail (e-mail) user within the network to which the individual or organization
belongs.",
          "items": {
            "$ref": "studentElectronicMail"
          }
        },
        "identificationCodes": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of studentIdentificationCodes.  A coding scheme that is used
for identification and record-keeping purposes by schools, social services, or other agencies to refer to a
student.",
          "items": {
            "$ref": "studentIdentificationCode"
          }
        },
        "identificationDocuments": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of studentIdentificationDocuments.  The documents presented
as evident to verify one's personal identity; for example: drivers license, passport, birth certificate, etc.",
          "items": {
            "$ref": "studentIdentificationDocument"
          }
        },
        "indicators": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of studentIndicators.  Indicator(s) or metric(s) computed for
the student (e.g., at risk) to influence more effective education or direct specific interventions.",
          "items": {
            "$ref": "studentIndicator"
          }
        },
        "internationalAddresses": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of studentInternationalAddresses.  The set of elements that
describes an international address.",
          "items": {
            "$ref": "studentInternationalAddress"
          }
        },
        "languages": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of studentLanguages.  The language(s) the individual uses to
communicate.",
          "items": {
            "$ref": "studentLanguage"
          }
        },
        "otherNames": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of studentOtherNames.  Other names (e.g., alias, nickname,
previous legal name) associated with a person.",
          "items": {
            "$ref": "studentOtherName"
          }
        },
        "programParticipations": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of studentProgramParticipations.  Key programs the student is
participating in or receives services from.",
          "items": {
            "$ref": "studentProgramParticipation"
          }
        },
        "races": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of studentRaces.  The general racial category which most
clearly reflects the individual's recognition of his or her community or with which the individual most
identifies. The data model allows for multiple entries so that each individual can specify all appropriate
races.",
          "items": {
            "$ref": "studentRace"
          }
        },
        "telephones": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of studentTelephones.  The 10-digit telephone number,
including the area code, for the person.",
          "items": {
            "$ref": "studentTelephone"
          }
        },
        "visas": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of studentVisas.  An indicator of a non-US citizen's Visa
type.",
          "items": {
            "$ref": "studentVisa"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "studentAddress": {
      "id": "studentAddress",
      "properties": {
        "addressType": {
          "type": "string",
          "required": true,
          "description": "The type of address listed for an individual or organization.    For example:
Physical Address, Mailing Address, Home Address, etc.)"
        },
        "stateAbbreviationType": {
          "type": "string",
          "required": true,
          "description": "The abbreviation for the state (within the United States) or outlying area in which
an address is located."
        },
        "streetNumberName": {
          "type": "string",
          "required": true,
          "description": "The street number and street name or post office box number of an address."
        },
        "apartmentRoomSuiteNumber": {
          "type": "string",
          "required": false,
          "description": "The apartment, room, or suite number of an address."
        },
        "buildingSiteNumber": {
          "type": "string",
          "required": false,
          "description": "The number of the building on the site, if more than one building shares the same
address."
        },
        "city": {
          "type": "string",
          "required": true,
          "description": "The name of the city in which an address is located."
        },
        "postalCode": {
          "type": "string",
          "required": true,
          "description": "The five or nine digit zip code or overseas postal code portion of an address."
        },
        "nameOfCounty": {
          "type": "string",
          "required": false,
          "description": "The name of the county, parish, borough, or comparable unit (within a state)
in                         'which an address is located."
        },
        "countyFIPSCode": {
          "type": "string",
          "required": false,
          "description": "The Federal Information Processing Standards (FIPS) numeric code for the county
issued by the National Institute of Standards and Technology (NIST). Counties are considered to be the \"first-
order subdivisions\" of each State and statistically equivalent entity, regardless of their local designations
(county, parish, borough, etc.) Counties in different States will have the same code. A unique county number is
created when combined with the 2-digit FIPS State Code."
        },
        "latitude": {
          "type": "string",
          "required": false,
          "description": "The geographic latitude of the physical address."
        },
        "longitude": {
          "type": "string",
          "required": false,
          "description": "The geographic longitude of the physical address."
        },
        "beginDate": {
          "type": "date-time",
          "required": false,
          "description": "The first date the address is valid. For physical addresses, the date the person
moved to that address."
        },
        "endDate": {
          "type": "date-time",
          "required": false,
          "description": "The last date the address is valid. For physical addresses, this would be the date
the person moved from that address."
        }
      }
    },
    "studentCharacteristic": {
      "id": "studentCharacteristic",
      "properties": {
        "descriptor": {
          "type": "string",
          "required": true,
          "description": "The characteristic designated for the Student."
        },
        "beginDate": {
          "type": "date-time",
          "required": false,
          "description": "The date the characteristic was designated."
        },
        "endDate": {
          "type": "date-time",
          "required": false,
          "description": "The date the characteristic was removed."
        },
        "designatedBy": {
          "type": "string",
          "required": false,
          "description": "The person, organization, or department that designated the characteristic."
        }
      }
    },
    "studentCohortYear": {
      "id": "studentCohortYear",
      "properties": {
        "schoolYearTypeReference": {
          "type": "schoolYearTypeReference",
          "required": true,
          "description": "A reference to the related SchoolYearType resource."
        },
        "cohortYearType": {
          "type": "string",
          "required": true,
          "description": "The type of cohort year (9th grade, graduation)."
        }
      }
    },
    "studentDisability": {
      "id": "studentDisability",
      "properties": {
        "disabilityDescriptor": {
          "type": "string",
          "required": true,
          "description": "A disability category that describes a child's impairment."
        },
        "disabilityDeterminationSourceType": {
          "type": "string",
          "required": false,
          "description": "The source that provided the disability determination."
        },
        "disabilityDiagnosis": {
          "type": "string",
          "required": false,
          "description": "A description of the disability diagnosis."
        },
        "orderOfDisability": {
          "type": "integer",
          "required": false,
          "description": "The order by severity of student's disabilities: 1- Primary, 2 -  Secondary, 3 -
Tertiary, etc."
        }
      }
    },
    "studentElectronicMail": {
      "id": "studentElectronicMail",
      "properties": {
        "electronicMailType": {
          "type": "string",
          "required": true,
          "description": "The type of email listed for an individual or organization. For example: Home
/Personal, Work, etc.)"
        },
        "electronicMailAddress": {
          "type": "string",
          "required": true,
          "description": "The electronic mail (e-mail) address listed for an individual or organization."
        },
        "primaryEmailAddressIndicator": {
          "type": "boolean",
          "required": false,
          "description": "An indication that the electronic mail address should be used as the principal
electronic mail address for an individual or organization."
        }
      }
    },
    "studentIdentificationCode": {
      "id": "studentIdentificationCode",
      "properties": {
        "studentIdentificationSystemDescriptor": {
          "type": "string",
          "required": true,
          "description": "A coding scheme that is used for identification and record-keeping purposes by
schools, social services, or other agencies to refer to a student."
```

```
        },
        "assigningOrganizationIdentificationCode": {
          "type": "string",
          "required": true,
          "description": "The organization code or name assigning the StudentIdentificationCode."
        },
        "identificationCode": {
          "type": "string",
          "required": true,
          "description": "A unique number or alphanumeric code assigned to a student by a school, school
system, a state, or other agency or entity."
        }
      }
    },
    "studentIdentificationDocument": {
      "id": "studentIdentificationDocument",
      "properties": {
        "identificationDocumentUseType": {
          "type": "string",
          "required": true,
          "description": "The primary function of the document used for establishing identity."
        },
        "personalInformationVerificationType": {
          "type": "string",
          "required": true,
          "description": "The category of the document relative to its purpose."
        },
        "issuerCountryDescriptor": {
          "type": "string",
          "required": false,
          "description": "Country of origin of the document."
        },
        "documentTitle": {
          "type": "string",
          "required": false,
          "description": "The title of the document given by the issuer."
        },
        "documentExpirationDate": {
          "type": "date-time",
          "required": false,
          "description": "The day when the document  expires, if null then never expires."
        },
        "issuerDocumentIdentificationCode": {
          "type": "string",
          "required": false,
          "description": "The unique identifier on the issuer's identification system."
        },
        "issuerName": {
          "type": "string",
          "required": false,
          "description": "Name of the entity or institution that issued the document."
        }
      }
    },
    "studentIndicator": {
      "id": "studentIndicator",
      "properties": {
        "indicatorName": {
          "type": "string",
          "required": true,
          "description": "The name of the indicator or metric."
        },
        "indicatorGroup": {
          "type": "string",
          "required": false,
          "description": "The name for a group of indicators."
        },
        "indicator": {
          "type": "string",
          "required": true,
          "description": "The value of the indicator or metric."
        },
        "beginDate": {
          "type": "date-time",
          "required": false,
          "description": "The date when the indicator was assigned or computed."
        },
        "endDate": {
          "type": "date-time",
          "required": false,
          "description": "The date the indicator or metric was sunset or removed."
        },
        "designatedBy": {
          "type": "string",
          "required": false,
          "description": "The person, organization, or department that designated the program association."
        }
      }
    },
    "studentInternationalAddress": {
      "id": "studentInternationalAddress",
      "properties": {
        "addressType": {
          "type": "string",
          "required": true,
          "description": "The type of address listed for an individual or organization. For example:  Physical
Address, Mailing Address, Home Address, etc.)"
        },
        "countryDescriptor": {
          "type": "string",
          "required": true,
          "description": "The name of the country."
        },
        "addressLine1": {
          "type": "string",
          "required": true,
          "description": "The first line of the address."
        },
        "addressLine2": {
          "type": "string",
          "required": false,
          "description": "The second line of the address."
        },
        "addressLine3": {
          "type": "string",
          "required": false,
          "description": "The third line of the address."
        },
        "addressLine4": {
          "type": "string",
          "required": false,
          "description": "The fourth line of the address."
        },
        "latitude": {
          "type": "string",
          "required": false,
          "description": "The geographic latitude of the physical address."
        },
        "longitude": {
          "type": "string",
          "required": false,
          "description": "The geographic longitude of the physical address."
        },
        "beginDate": {
          "type": "date-time",
          "required": false,
          "description": "The first date the address is valid. For physical addresses, the date the person
moved to that address."
        },
        "endDate": {
          "type": "date-time",
          "required": false,
          "description": "The last date the address is valid. For physical addresses, this would be the date 
the person moved from that address."
        }
      }
    },
    "studentLanguage": {
      "id": "studentLanguage",
      "properties": {
        "languageDescriptor": {
          "type": "string",
          "required": true,
          "description": "A specification of which written or spoken communication is being used."
        },
        "uses": {
          "type": "array",
          "required": true,
          "description": "An unordered collection of studentLanguageUses.  A description of how the language is 
used (e.g. Home Language, Native Language, Spoken Language).",
          "items": {
            "$ref": "studentLanguageUse"
          }
        }
      }
    },
    "studentOtherName": {
      "id": "studentOtherName",
      "properties": {
        "otherNameType": {
          "type": "string",
          "required": true,
          "description": "The types of alternate names for a person."
        },
        "personalTitlePrefix": {
          "type": "string",
          "required": false,
          "description": "A prefix used to denote the title, degree, position, or seniority of the person."
        },
        "firstName": {
          "type": "string",
          "required": true,
          "description": "A name given to an individual at birth, baptism, or during another naming ceremony, 
or through legal change."
        },
        "middleName": {
          "type": "string",
          "required": false,
          "description": "A secondary name given to an individual at birth, baptism, or during another naming 
ceremony."
        },
        "lastSurname": {
          "type": "string",
          "required": true,
          "description": "The name borne in common by members of a family."
        },
        "generationCodeSuffix": {
          "type": "string",
          "required": false,
          "description": "An appendage, if any, used to denote an individual's generation in his family (e.g., 
Jr., Sr., III)."
        }
      }
    },
    "studentProgramParticipation": {
      "id": "studentProgramParticipation",
      "properties": {
        "programType": {
          "type": "string",
          "required": true,
          "description": "The type of program."
        },
        "beginDate": {
          "type": "date-time",
          "required": false,
          "description": "The date the Student was associated with the Program or service."
        },
        "endDate": {
          "type": "date-time",
          "required": false,
          "description": "The date the Program participation ended."
        },
        "designatedBy": {
          "type": "string",
          "required": false,
          "description": "The person, organization, or department that designated the program association."
        },
        "programCharacteristics": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of studentProgramParticipationProgramCharacteristics.
Reflects important characteristics of the Program, such as categories or particular indications.",
          "items": {
            "$ref": "studentProgramParticipationProgramCharacteristic"
          }
        }
      }
    },
    "studentRace": {
      "id": "studentRace",
      "properties": {
        "raceType": {
          "type": "string",
          "required": true,
          "description": "The general racial category which most clearly reflects the individual's recognition
of his or her community or with which the individual most identifies. The data model allows for multiple
entries so that each individual can specify all appropriate races."
        }
      }
    },
    "studentTelephone": {
      "id": "studentTelephone",
      "properties": {
        "telephoneNumberType": {
          "type": "string",
          "required": true,
          "description": "The type of communication number listed for an individual or organization."
        },
        "telephoneNumber": {
          "type": "string",
          "required": true,
          "description": "The telephone number including the area code, and extension, if applicable."
        },
        "orderOfPriority": {
          "type": "integer",
          "required": false,
          "description": "The order of priority assigned to telephone numbers to define which number to attempt
first, second, etc."
        },
        "textMessageCapabilityIndicator": {
          "type": "boolean",
          "required": false,
          "description": "An indication that the telephone number is technically capable of sending and
receiving Short Message Service (SMS) text messages."
        }
      }
    },
    "studentVisa": {
      "id": "studentVisa",
      "properties": {
        "visaType": {
          "type": "string",
          "required": true,
          "description": "An indicator of a non-US citizen's Visa type."
        }
      }
    },
    "studentLanguageUse": {
      "id": "studentLanguageUse",
      "properties": {
        "languageUseType": {
          "type": "string",
          "required": true,
          "description": "A description of how the language is used (e.g. Home Language, Native Language, 
Spoken Language)."
        }
      }
    },
    "studentProgramParticipationProgramCharacteristic": {
      "id": "studentProgramParticipationProgramCharacteristic",
      "properties": {
        "programCharacteristicDescriptor": {
          "type": "string",
          "required": true,
          "description": "Reflects important characteristics of the Program, such as categories or particular 
indications."
        }
      }
    },
    "studentLearningStyle": {
      "id": "studentLearningStyle",
      "properties": {
        "visualLearning": {
          "type": "number",
          "required": true,
          "description": "The student's relative preference expressed as a percent to visual learning."
        },
        "auditoryLearning": {
          "type": "number",
          "required": true,
          "description": "The student's relative preference expressed as a percent to auditory learning."
        },
        "tactileLearning": {
          "type": "number",
          "required": true,
          "description": "The student's relative preference expressed as a percent to kinesthetic or tactile 
learning."
        }
      }
    },
    "schoolYearTypeReference": {
      "id": "schoolYearTypeReference",
      "properties": {
        "schoolYear": {
          "type": "integer",
          "required": true,
          "description": "Key for School Year"
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/studentAcademicRecords

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/studentAcademicRecords",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/studentAcademicRecords",
      "description": "This educational entity represents the cumulative record of academic achievement for a
student.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentAcademicRecordsAll",
          "type": "array",
          "items": {
            "$ref": "studentAcademicRecord"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentAcademicRecord"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentAcademicRecordsByExample",
          "type": "array",
          "items": {
            "$ref": "studentAcademicRecord"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolYear",
              "description": "The identifier for the school year.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "termDescriptor",
              "description": "The term for the session during the school year.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentAcademicRecord"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentAcademicRecordByKey",
          "type": "studentAcademicRecord",
          "parameters": [
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolYear",
              "description": "The identifier for the school year.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "termDescriptor",
              "description": "The term for the session during the school year.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentAcademicRecord"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postStudentAcademicRecords",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "studentAcademicRecord",
              "description": "The JSON representation of the \"studentAcademicRecord\" resource to be created
or updated.",
              "type": "studentAcademicRecord",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
```

```
      ]
    },
    {
      "path": "/studentAcademicRecords/{id}",
      "description": "This educational entity represents the cumulative record of academic achievement for a
student.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentAcademicRecordsById",
          "type": "studentAcademicRecord",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentAcademicRecord"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putStudentAcademicRecord",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "studentAcademicRecord",
              "description": "The JSON representation of the \"studentAcademicRecord\" resource to be updated.",
              "type": "studentAcademicRecord",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteStudentAcademicRecordById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "studentAcademicRecord": {
      "id": "studentAcademicRecord",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "educationOrganizationReference": {
          "type": "educationOrganizationReference",
          "required": true,
          "description": "A reference to the related EducationOrganization resource."
        },
        "schoolYearTypeReference": {
          "type": "schoolYearTypeReference",
          "required": true,
          "description": "A reference to the related SchoolYearType resource."
        },
        "studentReference": {
          "type": "studentReference",
          "required": true,
          "description": "A reference to the related Student resource."
        },
        "cumulativeAttemptedCreditConversion": {
          "type": "number",
          "required": false,
          "description": "Conversion factor that when multiplied by the number of credits is equivalent to
Carnegie units."
        },
        "cumulativeAttemptedCredits": {
          "type": "number",
          "required": true,
          "description": "The value of credits or units of value awarded for the completion of a course."
        },
        "cumulativeAttemptedCreditType": {
          "type": "string",
          "required": false,
          "description": "The type of credits or units of value awarded for the completion of a course."
        },
        "cumulativeEarnedCreditConversion": {
          "type": "number",
          "required": false,
          "description": "Conversion factor that when multiplied by the number of credits is equivalent to
Carnegie units."
        },
        "cumulativeEarnedCredits": {
          "type": "number",
          "required": true,
          "description": "The value of credits or units of value awarded for the completion of a course."
        },
        "cumulativeEarnedCreditType": {
          "type": "string",
          "required": false,
          "description": "The type of credits or units of value awarded for the completion of a course."
        },
        "cumulativeGradePointAverage": {
          "type": "number",
          "required": false,
          "description": "A measure of average performance in all courses taken by an individual during his or
her school career as determined for record-keeping purposes. This is obtained by dividing the total grade
points received by the total number of credits attempted. This usually includes grade points received and
credits attempted in his or her current school as well as those transferred from schools in which the
individual was previously enrolled."
        },
        "cumulativeGradePointsEarned": {
          "type": "number",
          "required": false,
          "description": "The cumulative number of grade points an individual earns by successfully completing
courses or examinations during his or her enrollment in the current school as well as those transferred from
schools in which the individual had been previously enrolled."
        },
        "gradeValueQualifier": {
          "type": "string",
          "required": false,
          "description": "The scale of equivalents, if applicable, for grades awarded as indicators of
performance in schoolwork. For example, numerical equivalents for letter grades used in determining a student's
Grade Point Average (A=4, B=3, C=2, D=1 in a four-point system) or letter equivalents for percentage grades (90-
100%=A, 80-90%=B, etc.)"
        },
        "projectedGraduationDate": {
          "type": "date-time",
          "required": false,
          "description": "The month and year the student is projected to graduate."
        },
        "sessionAttemptedCreditConversion": {
          "type": "number",
          "required": false,
          "description": "Conversion factor that when multiplied by the number of credits is equivalent to
Carnegie units."
        },
        "sessionAttemptedCredits": {
          "type": "number",
          "required": true,
          "description": "The value of credits or units of value awarded for the completion of a course."
        },
        "sessionAttemptedCreditType": {
          "type": "string",
          "required": false,
          "description": "The type of credits or units of value awarded for the completion of a course."
        },
        "sessionEarnedCreditConversion": {
          "type": "number",
          "required": false,
          "description": "Conversion factor that when multiplied by the number of credits is equivalent to
Carnegie units."
        },
        "sessionEarnedCredits": {
          "type": "number",
          "required": true,
          "description": "The value of credits or units of value awarded for the completion of a course."
        },
        "sessionEarnedCreditType": {
          "type": "string",
          "required": false,
          "description": "The type of credits or units of value awarded for the completion of a course."
        },
        "sessionGradePointAverage": {
          "type": "number",
          "required": false,
          "description": "The grade point average for an individual computed as the grade points earned during
the session divided by the number of credits attempted."
        },
        "sessionGradePointsEarned": {
          "type": "number",
          "required": false,
          "description": "The number of grade points an individual earned for this session."
        },
        "termDescriptor": {
          "type": "string",
          "required": true,
          "description": "The term for the session during the school year."
        },
        "classRanking": {
          "type": "studentAcademicRecordClassRanking",
          "required": false,
          "description": "The academic rank information of a student in relation to his or her graduating
class."
        },
        "academicHonors": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of studentAcademicRecordAcademicHonors.  Academic
distinctions earned by or awarded to the student.",
          "items": {
            "$ref": "studentAcademicRecordAcademicHonor"
          }
        },
        "diplomas": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of studentAcademicRecordDiplomas.  Diploma(s) earned by the
student.",
          "items": {
            "$ref": "studentAcademicRecordDiploma"
          }
        },
        "recognitions": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of studentAcademicRecordRecognitions.  Recognitions given to
the student for accomplishments in a co-curricular or extracurricular activity.",
          "items": {
            "$ref": "studentAcademicRecordRecognition"
          }
        },
        "reportCards": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of studentAcademicRecordReportCards.  Report cards for the
student.",
          "items": {
            "$ref": "studentAcademicRecordReportCard"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "educationOrganizationReference": {
      "id": "educationOrganizationReference",
      "properties": {
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
```

```
Also known as the State LEA ID."
        }
      }
    },
    "schoolYearTypeReference": {
      "id": "schoolYearTypeReference",
      "properties": {
        "schoolYear": {
          "type": "integer",
          "required": true,
          "description": "Key for School Year"
        }
      }
    },
    "studentReference": {
      "id": "studentReference",
      "properties": {
        "studentUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a student."
        }
      }
    },
    "studentAcademicRecordAcademicHonor": {
      "id": "studentAcademicRecordAcademicHonor",
      "properties": {
        "academicHonorCategoryType": {
          "type": "string",
          "required": true,
          "description": "A designation of the type of academic distinctions earned by or awarded to the
student."
        },
        "achievementCategoryDescriptor": {
          "type": "string",
          "required": false,
          "description": "The category of achievement attributed to the learner."
        },
        "achievementTitle": {
          "type": "string",
          "required": false,
          "description": "The title assigned to the achievement."
        },
        "achievementCategorySystem": {
          "type": "string",
          "required": false,
          "description": "The system that defines the categories by which an achievement is attributed to the
learner."
        },
        "issuerName": {
          "type": "string",
          "required": false,
          "description": "The name of the agent, entity, or institution issuing the element."
        },
        "issuerOriginURL": {
          "type": "string",
          "required": false,
          "description": "The Uniform Resource Locator (URL) from which the award was issued."
        },
        "criteria": {
          "type": "string",
          "required": false,
          "description": "The criteria for competency-based completion of the achievement/award."
        },
        "criteriaURL": {
          "type": "string",
          "required": false,
          "description": "The Uniform Resource Locator (URL) for the unique address of a web page describing
the competency-based completion criteria for the achievement/award."
        },
        "evidenceStatement": {
          "type": "string",
          "required": false,
          "description": "A statement or reference describing the evidence that the learner met the criteria
for attainment of the Achievement."
        },
        "imageURL": {
          "type": "string",
          "required": false,
          "description": "The Uniform Resource Locator (URL) for the unique address of an image representing an
award or badge associated with the Achievement."
        },
        "honorDescription": {
          "type": "string",
          "required": false,
          "description": "A description of the type of academic distinctions earned by or awarded to the
individual."
        },
        "honorAwardDate": {
          "type": "date-time",
          "required": false,
          "description": "The date the honor was awarded or earned."
        },
        "honorAwardExpiresDate": {
          "type": "date-time",
          "required": false,
          "description": "Date on which the award expires."
        }
      }
    },
    "studentAcademicRecordDiploma": {
      "id": "studentAcademicRecordDiploma",
      "properties": {
        "diplomaType": {
          "type": "string",
          "required": true,
          "description": "The type of diploma/credential that is awarded to a student in recognition of his/her
completion of the curricular requirements."
        },
        "diplomaAwardDate": {
          "type": "date-time",
          "required": true,
          "description": "The month, day, and year on which the student met  graduation requirements and was
awarded a diploma."
        },
        "achievementCategoryDescriptor": {
          "type": "string",
          "required": false,
          "description": "The category of achievement attributed to the learner."
        },
        "diplomaLevelType": {
          "type": "string",
          "required": false,
          "description": "The level of diploma/credential that is awarded to a student in recognition of his
/her completion of the curricular requirements.          Minimum high school program          Recommended high
school program          Distinguished Achievement Program."
        },
        "achievementTitle": {
          "type": "string",
          "required": false,
          "description": "The title assigned to the achievement."
        },
        "achievementCategorySystem": {
          "type": "string",
          "required": false,
          "description": "The system that defines the categories by which an achievement is attributed to the
learner."
        },
        "issuerName": {
          "type": "string",
          "required": false,
          "description": "The name of the agent, entity, or institution issuing the element."
        },
        "issuerOriginURL": {
          "type": "string",
          "required": false,
          "description": "The Uniform Resource Locator (URL) from which the award was issued."
        },
        "criteria": {
          "type": "string",
          "required": false,
          "description": "The criteria for competency-based completion of the achievement/award."
        },
        "criteriaURL": {
          "type": "string",
          "required": false,
          "description": "The Uniform Resource Locator (URL) for the unique address of a web page describing
the competency-based completion criteria for the achievement/award."
        },
        "evidenceStatement": {
          "type": "string",
          "required": false,
          "description": "A statement or reference describing the evidence that the learner met the criteria
for attainment of the Achievement."
        },
        "imageURL": {
          "type": "string",
          "required": false,
          "description": "The Uniform Resource Locator (URL) for the unique address of an image representing an
award or badge associated with the Achievement."
        },
        "cteCompleter": {
          "type": "boolean",
          "required": false,
          "description": "Indicated a student who reached a state-defined threshold of vocational education and
who attained a high school diploma or its recognized state equivalent or GED."
        },
        "diplomaDescription": {
          "type": "string",
          "required": false,
          "description": "The description of diploma given to the student for accomplishments."
        },
        "diplomaAwardExpiresDate": {
          "type": "date-time",
          "required": false,
          "description": "Date on which the award expires."
        }
      }
    },
    "studentAcademicRecordRecognition": {
      "id": "studentAcademicRecordRecognition",
      "properties": {
        "recognitionType": {
          "type": "string",
          "required": true,
          "description": "The nature of recognition given to the student for accomplishments in a co-
curricular, or extra-curricular activity."
        },
        "achievementCategoryDescriptor": {
          "type": "string",
          "required": false,
          "description": "The category of achievement attributed to the learner."
        },
        "achievementTitle": {
          "type": "string",
          "required": false,
          "description": "The title assigned to the achievement."
        },
        "achievementCategorySystem": {
          "type": "string",
          "required": false,
          "description": "The system that defines the categories by which an achievement is attributed to the
learner."
```

```
        },
        "issuerName": {
          "type": "string",
          "required": false,
          "description": "The name of the agent, entity, or institution issuing the element."
        },
        "issuerOriginURL": {
          "type": "string",
          "required": false,
          "description": "The Uniform Resource Locator (URL) from which the award was issued."
        },
        "criteria": {
          "type": "string",
          "required": false,
          "description": "The criteria for competency-based completion of the achievement/award."
        },
        "criteriaURL": {
          "type": "string",
          "required": false,
          "description": "The Uniform Resource Locator (URL) for the unique address of a web page describing
the competency-based completion criteria for the achievement/award."
        },
        "evidenceStatement": {
          "type": "string",
          "required": false,
          "description": "A statement or reference describing the evidence that the learner met the criteria
for attainment of the Achievement."
        },
        "imageURL": {
          "type": "string",
          "required": false,
          "description": "The Uniform Resource Locator (URL) for the unique address of an image representing an
award or badge associated with the Achievement."
        },
        "recognitionDescription": {
          "type": "string",
          "required": false,
          "description": "A description of the type of academic distinctions earned by or awarded to the
individual."
        },
        "recognitionAwardDate": {
          "type": "date-time",
          "required": false,
          "description": "The date the recognition was awarded or earned."
        },
        "recognitionAwardExpiresDate": {
          "type": "date-time",
          "required": false,
          "description": "Date on which the award expires."
        }
      }
    },
    "studentAcademicRecordReportCard": {
      "id": "studentAcademicRecordReportCard",
      "properties": {
        "reportCardReference": {
          "type": "reportCardReference",
          "required": true,
          "description": "A reference to the related ReportCard resource."
        }
      }
    },
    "studentAcademicRecordClassRanking": {
      "id": "studentAcademicRecordClassRanking",
      "properties": {
        "classRank": {
          "type": "integer",
          "required": true,
          "description": "The academic rank of a student in relation to his or her graduating class (e.g., 1st,
2nd, 3rd)."
        },
        "totalNumberInClass": {
          "type": "integer",
          "required": true,
          "description": "The total number of students in the student's graduating class."
        },
        "percentageRanking": {
          "type": "integer",
          "required": false,
          "description": "The academic percentage rank of a student in relation to his or her graduating class
(e.g., 95%, 80%, 50%)."
        },
        "classRankingDate": {
          "type": "date-time",
          "required": false,
          "description": "Date class ranking was determined."
        }
      }
    },
    "reportCardReference": {
      "id": "reportCardReference",
      "properties": {
        "studentUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a student."
        },
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        },
        "gradingPeriodDescriptor": {
          "type": "string",
          "required": true,
          "description": "The name of the period for which grades are reported."
        },
        "gradingPeriodBeginDate": {
          "type": "date-time",
          "required": true,
          "description": "Month, day, and year of the first day of the GradingPeriod."
        },
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/studentCohortAssociations

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/studentCohortAssociations",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/studentCohortAssociations",
      "description": "This association represents the Cohort(s) for which a student is designated.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentCohortAssociationsAll",
          "type": "array",
          "items": {
            "$ref": "studentCohortAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentCohortAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentCohortAssociationsByExample",
          "type": "array",
          "items": {
            "$ref": "studentCohortAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "beginDate",
              "description": "The month, day, and year on which the Student was first identified as part of the
Cohort.",
              "type": "date-time",
              "required": false
            },
            {
              "paramType": "query",
              "name": "cohortIdentifier",
              "description": "The name or ID for the Cohort.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentCohortAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentCohortAssociationByKey",
          "type": "studentCohortAssociation",
          "parameters": [
            {
              "paramType": "query",
              "name": "beginDate",
              "description": "The month, day, and year on which the Student was first identified as part of the
Cohort.",
              "type": "date-time",
              "required": true
            },
            {
              "paramType": "query",
              "name": "cohortIdentifier",
              "description": "The name or ID for the Cohort.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentCohortAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postStudentCohortAssociations",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "studentCohortAssociation",
              "description": "The JSON representation of the \"studentCohortAssociation\" resource to be
created or updated.",
              "type": "studentCohortAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/studentCohortAssociations/{id}",
      "description": "This association represents the Cohort(s) for which a student is designated.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentCohortAssociationsById",
          "type": "studentCohortAssociation",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentCohortAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putStudentCohortAssociation",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "studentCohortAssociation",
              "description": "The JSON representation of the \"studentCohortAssociation\" resource to be
updated.",
              "type": "studentCohortAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
```

```
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteStudentCohortAssociationById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "studentCohortAssociation": {
      "id": "studentCohortAssociation",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "cohortReference": {
          "type": "cohortReference",
          "required": true,
          "description": "A reference to the related Cohort resource."
        },
        "studentReference": {
          "type": "studentReference",
          "required": true,
          "description": "A reference to the related Student resource."
        },
        "beginDate": {
          "type": "date-time",
          "required": true,
          "description": "The month, day, and year on which the Student was first identified as part of the
Cohort."
        },
        "endDate": {
          "type": "date-time",
          "required": true,
          "description": "The month, day, and year on which the Student was removed as part of the Cohort."
        },
        "sections": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of studentCohortAssociationSections.  The Cohort representing
the subdivision of students within one or more sections. For example, a group of students may be given
additional instruction and tracked as a cohort.",
          "items": {
            "$ref": "studentCohortAssociationSection"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "cohortReference": {
      "id": "cohortReference",
      "properties": {
        "identifier": {
          "type": "string",
          "required": true,
          "description": "The name or ID for the Cohort."
        },
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        }
      }
    },
    "studentReference": {
      "id": "studentReference",
      "properties": {
        "studentUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a student."
        }
      }
    },
    "studentCohortAssociationSection": {
      "id": "studentCohortAssociationSection",
      "properties": {
        "sectionReference": {
          "type": "sectionReference",
          "required": true,
          "description": "A reference to the related Section resource."
        }
      }
    },
    "sectionReference": {
      "id": "sectionReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        },
        "classPeriodName": {
          "type": "string",
          "required": true,
          "description": "An indication of the portion of a typical daily session in which students receive
instruction in a specified subject (e.g., morning, sixth period, block period, or AB schedules)."
        },
        "classroomIdentificationCode": {
          "type": "string",
          "required": true,
          "description": "A unique number or alphanumeric code assigned to a room by a school, school system,
state, or other agency or entity."
        },
        "localCourseCode": {
          "type": "string",
          "required": true,
          "description": "The local code assigned by the School that identifies the course offering provided
for the instruction of students."
        },
        "termDescriptor": {
          "type": "string",
          "required": true,
          "description": "The term for the Session during the school year."
        },
        "schoolYear": {
          "type": "integer",
          "required": true,
          "description": "The identifier for the school year."
        },
        "uniqueSectionCode": {
          "type": "string",
          "required": true,
          "description": "A unique identifier for the Section that is defined by the classroom, the subjects
taught, and the instructors who are assigned."
```

```
        },
        "sequenceOfCourse": {
          "type": "integer",
          "required": true,
          "description": "When a section is part of a sequence of parts for a course, the number of the
sequence. If the course has only one part, the value of this section attribute should be 1."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/studentDisciplineIncidentAssociations

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/studentDisciplineIncidentAssociations",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/studentDisciplineIncidentAssociations",
      "description": "This association indicates those students who were victims, perpetrators, witnesses, and
reporters for a discipline incident.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentDisciplineIncidentAssociationsAll",
          "type": "array",
          "items": {
            "$ref": "studentDisciplineIncidentAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentDisciplineIncidentAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentDisciplineIncidentAssociationsByExample",
          "type": "array",
          "items": {
            "$ref": "studentDisciplineIncidentAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "incidentIdentifier",
              "description": "A locally assigned unique identifier (within the school or school district) to
identify each specific DisciplineIncident or occurrence. The same identifier should be used to document the
entire DisciplineIncident even if it included multiple offenses and multiple offenders.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentDisciplineIncidentAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentDisciplineIncidentAssociationByKey",
          "type": "studentDisciplineIncidentAssociation",
          "parameters": [
            {
              "paramType": "query",
              "name": "incidentIdentifier",
              "description": "A locally assigned unique identifier (within the school or school district) to
identify each specific DisciplineIncident or occurrence. The same identifier should be used to document the
entire DisciplineIncident even if it included multiple offenses and multiple offenders.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentDisciplineIncidentAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postStudentDisciplineIncidentAssociations",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "studentDisciplineIncidentAssociation",
              "description": "The JSON representation of the \"studentDisciplineIncidentAssociation\" resource
to be created or updated.",
              "type": "studentDisciplineIncidentAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/studentDisciplineIncidentAssociations/{id}",
      "description": "This association indicates those students who were victims, perpetrators, witnesses, and
reporters for a discipline incident.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentDisciplineIncidentAssociationsById",
          "type": "studentDisciplineIncidentAssociation",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentDisciplineIncidentAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
```

```
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putStudentDisciplineIncidentAssociation",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "studentDisciplineIncidentAssociation",
              "description": "The JSON representation of the \"studentDisciplineIncidentAssociation\" resource
to be updated.",
              "type": "studentDisciplineIncidentAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
```

```
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteStudentDisciplineIncidentAssociationById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
```

```
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "studentDisciplineIncidentAssociation": {
      "id": "studentDisciplineIncidentAssociation",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "disciplineIncidentReference": {
          "type": "disciplineIncidentReference",
          "required": true,
          "description": "A reference to the related DisciplineIncident resource."
        },
        "studentReference": {
          "type": "studentReference",
          "required": true,
          "description": "A reference to the related Student resource."
        },
        "studentParticipationCodeType": {
          "type": "string",
          "required": true,
          "description": "The role or type of participation of a student in a discipline incident;
for example:          Victim          Perpetrator          Witness          Reporter."
        },
        "behaviors": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of studentDisciplineIncidentAssociationBehaviors.  Describes
behavior by category and provides a detailed description.",
          "items": {
            "$ref": "studentDisciplineIncidentAssociationBehavior"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "disciplineIncidentReference": {
      "id": "disciplineIncidentReference",
      "properties": {
        "incidentIdentifier": {
          "type": "string",
          "required": true,
          "description": "A locally assigned unique identifier (within the school or school district) to
identify each specific DisciplineIncident or occurrence. The same identifier should be used to document the
entire DisciplineIncident even if it included multiple offenses and multiple offenders."
        },
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        }
      }
    },
    "studentReference": {
      "id": "studentReference",
      "properties": {
        "studentUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a student."
        }
      }
    },
    "studentDisciplineIncidentAssociationBehavior": {
      "id": "studentDisciplineIncidentAssociationBehavior",
      "properties": {
        "behaviorDescriptor": {
          "type": "string",
          "required": true,
          "description": "Describes behavior by category and provides a detailed description."
        },
        "behaviorDetailedDescription": {
          "type": "string",
          "required": false,
          "description": "Specifies a more granular level of detail of a behavior involved in the incident."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/studentEducationOrganizationAssociations

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/studentEducationOrganizationAssociations",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/studentEducationOrganizationAssociations",
      "description": "This association indicates any relationship between a student and an education
organization other than how the state views enrollment. Enrollment relationship semantics are covered by
StudentSchoolAssociation.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentEducationOrganizationAssociationsAll",
          "type": "array",
          "items": {
            "$ref": "studentEducationOrganizationAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentEducationOrganizationAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentEducationOrganizationAssociationsByExample",
          "type": "array",
          "items": {
            "$ref": "studentEducationOrganizationAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "responsibilityDescriptor",
              "description": "Indications of an education organization's responsibility for a student, such as
accountability, attendance, funding, etc.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentEducationOrganizationAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentEducationOrganizationAssociationByKey",
          "type": "studentEducationOrganizationAssociation",
          "parameters": [
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "responsibilityDescriptor",
              "description": "Indications of an education organization's responsibility for a student, such as
accountability, attendance, funding, etc.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentEducationOrganizationAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postStudentEducationOrganizationAssociations",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "studentEducationOrganizationAssociation",
              "description": "The JSON representation of the \"studentEducationOrganizationAssociation\"
resource to be created or updated.",
              "type": "studentEducationOrganizationAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
```

```
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/studentEducationOrganizationAssociations/{id}",
      "description": "This association indicates any relationship between a student and an education
organization other than how the state views enrollment. Enrollment relationship semantics are covered by
StudentSchoolAssociation.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentEducationOrganizationAssociationsById",
          "type": "studentEducationOrganizationAssociation",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentEducationOrganizationAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putStudentEducationOrganizationAssociation",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
```

```
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "studentEducationOrganizationAssociation",
              "description": "The JSON representation of the \"studentEducationOrganizationAssociation\"
resource to be updated.",
              "type": "studentEducationOrganizationAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteStudentEducationOrganizationAssociationById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "studentEducationOrganizationAssociation": {
      "id": "studentEducationOrganizationAssociation",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "educationOrganizationReference": {
          "type": "educationOrganizationReference",
          "required": true,
          "description": "A reference to the related EducationOrganization resource."
        },
        "studentReference": {
          "type": "studentReference",
          "required": true,
          "description": "A reference to the related Student resource."
        },
        "responsibilityDescriptor": {
          "type": "string",
          "required": true,
          "description": "Indications of an education organization's responsibility for a student, such as
accountability, attendance, funding, etc."
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "educationOrganizationReference": {
      "id": "educationOrganizationReference",
      "properties": {
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        }
      }
    },
    "studentReference": {
      "id": "studentReference",
      "properties": {
        "studentUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a student."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/studentParentAssociations

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/studentParentAssociations",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/studentParentAssociations",
      "description": "This association relates students to their parents, guardians, or caretakers.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentParentAssociationsAll",
          "type": "array",
          "items": {
            "$ref": "studentParentAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentParentAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentParentAssociationsByExample",
          "type": "array",
          "items": {
            "$ref": "studentParentAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "parentUniqueId",
              "description": "A unique alphanumeric code assigned to a parent.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentParentAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentParentAssociationByKey",
          "type": "studentParentAssociation",
          "parameters": [
            {
              "paramType": "query",
              "name": "parentUniqueId",
              "description": "A unique alphanumeric code assigned to a parent.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentParentAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postStudentParentAssociations",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "studentParentAssociation",
              "description": "The JSON representation of the \"studentParentAssociation\" resource to be
created or updated.",
              "type": "studentParentAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
```

```
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/studentParentAssociations/{id}",
      "description": "This association relates students to their parents, guardians, or caretakers.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentParentAssociationsById",
          "type": "studentParentAssociation",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentParentAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putStudentParentAssociation",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "studentParentAssociation",
              "description": "The JSON representation of the \"studentParentAssociation\" resource to be
updated.",
              "type": "studentParentAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteStudentParentAssociationById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "studentParentAssociation": {
      "id": "studentParentAssociation",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "parentReference": {
          "type": "parentReference",
          "required": true,
          "description": "A reference to the related Parent resource."
        },
        "studentReference": {
          "type": "studentReference",
          "required": true,
          "description": "A reference to the related Student resource."
        },
        "contactPriority": {
          "type": "integer",
          "required": false,
          "description": "The numeric order of the preferred sequence or priority of contact."
        },
        "contactRestrictions": {
          "type": "string",
          "required": false,
          "description": "Restrictions for student and/or teacher contact with the individual (e.g., the
student may not be picked up by the individual)."
        },
        "emergencyContactStatus": {
          "type": "boolean",
          "required": true,
          "description": "Indicator of whether the person is a designated emergency contact for the Student."
        },
        "livesWith": {
          "type": "boolean",
          "required": false,
          "description": "Indicator of whether the Student lives with the associated parent."
        },
        "primaryContactStatus": {
          "type": "boolean",
          "required": true,
          "description": "Indicator of whether the person is a primary parental contact for the Student."
        },
        "relationType": {
          "type": "string",
          "required": false,
          "description": "The nature of an individual's relationship to a student; for example:
Father, Mother, Step Father, Step Mother, Foster Father, Foster Mother, Guardian, etc."
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "parentReference": {
      "id": "parentReference",
      "properties": {
        "parentUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a parent."
        }
      }
    },
    "studentReference": {
      "id": "studentReference",
      "properties": {
        "studentUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a student."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/studentProgramAssocations

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/studentProgramAssociations",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/studentProgramAssociations",
      "description": "This association represents the Program(s) that a student participates in or is served
by.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentProgramAssociationsAll",
          "type": "array",
          "items": {
            "$ref": "studentProgramAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentProgramAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentProgramAssociationsByExample",
          "type": "array",
          "items": {
            "$ref": "studentProgramAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "beginDate",
              "description": "The month, day, and year on which the Student first received services.",
              "type": "date-time",
              "required": false
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "programEducationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "programName",
              "description": "The formal name of the Program of instruction, training, services, or benefits
available through federal, state, or local agencies.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "programType",
              "description": "The type of program.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentProgramAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentProgramAssociationByKey",
          "type": "studentProgramAssociation",
          "parameters": [
            {
              "paramType": "query",
              "name": "beginDate",
              "description": "The month, day, and year on which the Student first received services.",
              "type": "date-time",
              "required": true
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "programEducationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "programName",
              "description": "The formal name of the Program of instruction, training, services, or benefits
available through federal, state, or local agencies.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "programType",
              "description": "The type of program.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentProgramAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postStudentProgramAssociations",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "studentProgramAssociation",
              "description": "The JSON representation of the \"studentProgramAssociation\" resource to be
created or updated.",
              "type": "studentProgramAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/studentProgramAssociations/{id}",
      "description": "This association represents the Program(s) that a student participates in or is served
by.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentProgramAssociationsById",
          "type": "studentProgramAssociation",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentProgramAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putStudentProgramAssociation",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "studentProgramAssociation",
              "description": "The JSON representation of the \"studentProgramAssociation\" resource to be
updated.",
              "type": "studentProgramAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteStudentProgramAssociationById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "studentProgramAssociation": {
      "id": "studentProgramAssociation",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "educationOrganizationReference": {
          "type": "educationOrganizationReference",
          "required": true,
          "description": "A reference to the related EducationOrganization resource."
        },
        "programReference": {
          "type": "programReference",
          "required": true,
          "description": "A reference to the related Program resource."
        },
        "studentReference": {
          "type": "studentReference",
          "required": true,
          "description": "A reference to the related Student resource."
        },
        "beginDate": {
          "type": "date-time",
          "required": true,
          "description": "The month, day, and year on which the Student first received services."
        },
        "endDate": {
          "type": "date-time",
          "required": true,
          "description": "The month, day, and year on which the Student exited the Program or stopped receiving
services."
        },
        "reasonExitedDescriptor": {
          "type": "string",
          "required": false,
          "description": "The reason the child left the Program within a school or district."
        },
        "servedOutsideOfRegularSession": {
          "type": "boolean",
          "required": false,
          "description": "Indicates whether the Student received services during the summer session or between
sessions."
        },
        "services": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of studentProgramAssociationServices.  Indicates the Service
(s) being provided to the Student by the Program.",
          "items": {
            "$ref": "studentProgramAssociationService"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "educationOrganizationReference": {
      "id": "educationOrganizationReference",
      "properties": {
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        }
      }
    },
    "programReference": {
      "id": "programReference",
      "properties": {
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        },
        "type": {
          "type": "string",
          "required": true,
          "description": "The type of program."
        },
        "name": {
          "type": "string",
          "required": true,
          "description": "The formal name of the Program of instruction, training, services, or benefits
available through federal, state, or local agencies."
        }
      }
    },
    "studentReference": {
      "id": "studentReference",
      "properties": {
        "studentUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a student."
        }
      }
    },
    "studentProgramAssociationService": {
      "id": "studentProgramAssociationService",
      "properties": {
        "serviceDescriptor": {
          "type": "string",
          "required": true,
          "description": "Indicates the Service being provided to the student by the Program."
        },
        "primaryIndicator": {
          "type": "boolean",
          "required": false,
          "description": "True if service is a primary service."
        },
        "serviceBeginDate": {
          "type": "date-time",
          "required": false,
          "description": "First date the Student was in this option for the current school year."
        },
        "serviceEndDate": {
          "type": "date-time",
          "required": false,
          "description": "Last date the Student was in this option for the current school year."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/studentSchoolAssociations

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/studentSchoolAssociations",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/studentSchoolAssociations",
      "description": "This association represents the School in which a student is enrolled. The semantics of
enrollment may differ slightly by state. Non-enrollment relationships between a student and an education
organization may be described using the StudentEducationOrganizationAssociation.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentSchoolAssociationsAll",
          "type": "array",
          "items": {
            "$ref": "studentSchoolAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentSchoolAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentSchoolAssociationsByExample",
          "type": "array",
          "items": {
            "$ref": "studentSchoolAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "entryDate",
              "description": "The month, day, and year on which an individual enters and begins to receive
instructional services in a school.",
              "type": "date-time",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentSchoolAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentSchoolAssociationByKey",
          "type": "studentSchoolAssociation",
          "parameters": [
            {
              "paramType": "query",
              "name": "entryDate",
              "description": "The month, day, and year on which an individual enters and begins to receive
instructional services in a school.",
              "type": "date-time",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentSchoolAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postStudentSchoolAssociations",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "studentSchoolAssociation",
              "description": "The JSON representation of the \"studentSchoolAssociation\" resource to be
created or updated.",
              "type": "studentSchoolAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/studentSchoolAssociations/{id}",
      "description": "This association represents the School in which a student is enrolled. The semantics of
enrollment may differ slightly by state. Non-enrollment relationships between a student and an education
organization may be described using the StudentEducationOrganizationAssociation.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentSchoolAssociationsById",
          "type": "studentSchoolAssociation",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentSchoolAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putStudentSchoolAssociation",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "studentSchoolAssociation",
              "description": "The JSON representation of the \"studentSchoolAssociation\" resource to be
updated.",
              "type": "studentSchoolAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteStudentSchoolAssociationById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "studentSchoolAssociation": {
      "id": "studentSchoolAssociation",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "graduationPlanReference": {
          "type": "graduationPlanReference",
          "required": true,
          "description": "A reference to the related GraduationPlan resource."
        },
        "schoolReference": {
          "type": "schoolReference",
          "required": true,
          "description": "A reference to the related School resource."
        },
        "schoolYearTypeReference": {
          "type": "schoolYearTypeReference",
          "required": true,
          "description": "A reference to the related SchoolYearType resource."
        },
        "classOfSchoolYearTypeReference": {
          "type": "schoolYearTypeReference",
          "required": true,
          "description": "A reference to the related SchoolYearType resource."
        },
        "studentReference": {
          "type": "studentReference",
          "required": true,
          "description": "A reference to the related Student resource."
        },
        "employedWhileEnrolled": {
          "type": "boolean",
          "required": false,
          "description": "An individual who is a paid employee or works in his or her own business, profession,
or farm and at the same time is enrolled in secondary, postsecondary, or adult education."
        },
        "entryDate": {
          "type": "date-time",
          "required": true,
          "description": "The month, day, and year on which an individual enters and begins to receive
instructional services in a school."
        },
        "entryGradeLevelDescriptor": {
          "type": "string",
          "required": true,
          "description": "The grade level or primary instructional level at which a student enters and receives
services in a school or an educational institution during a given academic session."
        },
        "entryGradeLevelReasonType": {
          "type": "string",
          "required": false,
          "description": "The primary reason as to why a staff member determined that a student should be
promoted or not (or be demoted) at the end of a given school term."
        },
        "entryTypeDescriptor": {
          "type": "string",
          "required": true,
          "description": "The process by which a student enters a school during a given academic session."
        },
        "exitWithdrawDate": {
          "type": "date-time",
          "required": true,
          "description": "The month, day, and year of the first day after the date of an individual's last
attendance at a school (if known), the day on which an individual graduated, or the date on which it becomes
known officially that an individual left school."
        },
        "exitWithdrawTypeDescriptor": {
          "type": "string",
          "required": true,
          "description": "The circumstances under which the student exited from membership in an educational
institution."
        },
        "primarySchool": {
          "type": "boolean",
          "required": false,
          "description": "Indicates if a given enrollment record should be considered the primary record for a
student. If omitted, the default is true."
        },
        "repeatGradeIndicator": {
          "type": "boolean",
          "required": true,
          "description": "An indicator of whether the student is enrolling to repeat a grade level, either by
failure or an agreement to hold the student back."
        },
        "residencyStatusDescriptor": {
          "type": "string",
          "required": true,
          "description": "An indication of the location of a persons legal residence relative to (within or
outside of) the boundaries of the public school attended and its administrative unit."
        },
        "schoolChoiceTransfer": {
          "type": "boolean",
          "required": true,
          "description": "An indication of whether students transferred in or out of the school did so during
the school year under the provisions for public school choice in accordance with Title I, Part A, Section 1116."
        },
        "educationPlans": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of studentSchoolAssociationEducationPlans.  The type of
EducationPlan(s) the student is following, if appropriate. For example:          Special education IEP
Vocational.",
          "items": {
            "$ref": "studentSchoolAssociationEducationPlan"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "graduationPlanReference": {
      "id": "graduationPlanReference",
      "properties": {
        "typeDescriptor": {
          "type": "string",
          "required": true,
          "description": "The type of academic plan the student is following for graduation: for example, 
Minimum, Recommended, Distinguished, or Standard."
        },
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA). 
Also known as the State LEA ID."
        },
        "graduationSchoolYear": {
          "type": "integer",
          "required": true,
          "description": "The school year the student is expected to graduate."
        }
      }
    },
    "schoolReference": {
      "id": "schoolReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        }
      }
    },
    "schoolYearTypeReference": {
      "id": "schoolYearTypeReference",
      "properties": {
        "schoolYear": {
          "type": "integer",
          "required": true,
          "description": "Key for School Year"
        }
      }
    },
    "studentReference": {
      "id": "studentReference",
      "properties": {
        "studentUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a student."
        }
      }
    },
    "studentSchoolAssociationEducationPlan": {
      "id": "studentSchoolAssociationEducationPlan",
      "properties": {
        "educationPlanType": {
          "type": "string",
          "required": true,
          "description": "The type of EducationPlan(s) the student is following, if appropriate. For 
example:          Special education IEP          Vocational."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/studentSchoolAttendanceEvents

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/studentSchoolAttendanceEvents",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/studentSchoolAttendanceEvents",
      "description": "This event entity represents the recording of whether a student is in attendance for a
school day.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentSchoolAttendanceEventsAll",
          "type": "array",
          "items": {
            "$ref": "studentSchoolAttendanceEvent"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentSchoolAttendanceEvent"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentSchoolAttendanceEventsByExample",
          "type": "array",
          "items": {
            "$ref": "studentSchoolAttendanceEvent"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "attendanceEventCategoryDescriptor",
              "description": "A code describing the attendance event, for example:          Present
Unexcused absence          Excused absence          Tardy.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "eventDate",
              "description": "Date for this attendance event.",
              "type": "date-time",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolYear",
              "description": "The identifier for the school year.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "termDescriptor",
              "description": "The term for the Session during the school year.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentSchoolAttendanceEvent"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentSchoolAttendanceEventByKey",
          "type": "studentSchoolAttendanceEvent",
          "parameters": [
            {
              "paramType": "query",
              "name": "attendanceEventCategoryDescriptor",
              "description": "A code describing the attendance event, for example:          Present
Unexcused absence          Excused absence          Tardy.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "eventDate",
              "description": "Date for this attendance event.",
              "type": "date-time",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolYear",
              "description": "The identifier for the school year.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "termDescriptor",
              "description": "The term for the Session during the school year.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentSchoolAttendanceEvent"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postStudentSchoolAttendanceEvents",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "studentSchoolAttendanceEvent",
              "description": "The JSON representation of the \"studentSchoolAttendanceEvent\" resource to be
created or updated.",
              "type": "studentSchoolAttendanceEvent",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/studentSchoolAttendanceEvents/{id}",
      "description": "This event entity represents the recording of whether a student is in attendance for a
school day.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentSchoolAttendanceEventsById",
          "type": "studentSchoolAttendanceEvent",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentSchoolAttendanceEvent"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putStudentSchoolAttendanceEvent",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "studentSchoolAttendanceEvent",
              "description": "The JSON representation of the \"studentSchoolAttendanceEvent\" resource to be
updated.",
              "type": "studentSchoolAttendanceEvent",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteStudentSchoolAttendanceEventById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "studentSchoolAttendanceEvent": {
      "id": "studentSchoolAttendanceEvent",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "schoolReference": {
          "type": "schoolReference",
          "required": true,
          "description": "A reference to the related School resource."
        },
        "sessionReference": {
          "type": "sessionReference",
          "required": true,
          "description": "A reference to the related Session resource."
        },
        "studentReference": {
          "type": "studentReference",
          "required": true,
          "description": "A reference to the related Student resource."
        },
        "attendanceEventCategoryDescriptor": {
          "type": "string",
          "required": true,
          "description": "A code describing the attendance event, for example:          Present
Unexcused absence          Excused absence          Tardy."
        },
        "attendanceEventReason": {
          "type": "string",
          "required": false,
          "description": "The reported reason for a student's absence."
        },
        "educationalEnvironmentType": {
          "type": "string",
          "required": false,
          "description": "The setting in which a child receives education and related services. This attribute
is only used if it differs from the EducationalEnvironment of the Section. This is only used in the
AttendanceEvent if different from the associated Section."
        },
        "eventDate": {
          "type": "date-time",
          "required": true,
          "description": "Date for this attendance event."
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "schoolReference": {
      "id": "schoolReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        }
      }
    },
    "sessionReference": {
      "id": "sessionReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        },
        "schoolYear": {
          "type": "integer",
          "required": true,
          "description": "The identifier for the school year."
        },
        "termDescriptor": {
          "type": "string",
          "required": true,
          "description": "The term for the Session during the school year."
        }
      }
    },
    "studentReference": {
      "id": "studentReference",
      "properties": {
        "studentUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a student."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/studentSectionAssociations

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/studentSectionAssociations",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/studentSectionAssociations",
      "description": "This association indicates the course sections to which a student is assigned.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentSectionAssociationsAll",
          "type": "array",
          "items": {
            "$ref": "studentSectionAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentSectionAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentSectionAssociationsByExample",
          "type": "array",
          "items": {
            "$ref": "studentSectionAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "beginDate",
              "description": "Month, day, and year of the Student's entry or assignment to the Section.",
              "type": "date-time",
              "required": false
            },
            {
              "paramType": "query",
              "name": "classPeriodName",
              "description": "An indication of the portion of a typical daily session in which students receive
instruction in a specified subject (e.g., morning, sixth period, block period, or AB schedules).",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "classroomIdentificationCode",
              "description": "A unique number or alphanumeric code assigned to a room by a school, school
system, state, or other agency or entity.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "localCourseCode",
              "description": "The local code assigned by the School that identifies the course offering
provided for the instruction of students.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolYear",
              "description": "The identifier for the school year.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "sequenceOfCourse",
              "description": "When a section is part of a sequence of parts for a course, the number of the
sequence. If the course has only one part, the value of this section attribute should be 1.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "termDescriptor",
              "description": "The term for the Session during the school year.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "uniqueSectionCode",
              "description": "A unique identifier for the Section that is defined by the classroom, the
subjects taught, and the instructors who are assigned.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentSectionAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentSectionAssociationByKey",
          "type": "studentSectionAssociation",
          "parameters": [
            {
              "paramType": "query",
              "name": "beginDate",
              "description": "Month, day, and year of the Student's entry or assignment to the Section.",
              "type": "date-time",
              "required": true
            },
            {
              "paramType": "query",
              "name": "classPeriodName",
              "description": "An indication of the portion of a typical daily session in which students receive
instruction in a specified subject (e.g., morning, sixth period, block period, or AB schedules).",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "classroomIdentificationCode",
              "description": "A unique number or alphanumeric code assigned to a room by a school, school
system, state, or other agency or entity.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "localCourseCode",
              "description": "The local code assigned by the School that identifies the course offering
provided for the instruction of students.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolYear",
              "description": "The identifier for the school year.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "sequenceOfCourse",
              "description": "When a section is part of a sequence of parts for a course, the number of the
sequence. If the course has only one part, the value of this section attribute should be 1.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "termDescriptor",
              "description": "The term for the Session during the school year.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "uniqueSectionCode",
              "description": "A unique identifier for the Section that is defined by the classroom, the
subjects taught, and the instructors who are assigned.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentSectionAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postStudentSectionAssociations",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "studentSectionAssociation",
              "description": "The JSON representation of the \"studentSectionAssociation\" resource to be
created or updated.",
              "type": "studentSectionAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/studentSectionAssociations/{id}",
      "description": "This association indicates the course sections to which a student is assigned.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentSectionAssociationsById",
          "type": "studentSectionAssociation",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentSectionAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putStudentSectionAssociation",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "studentSectionAssociation",
              "description": "The JSON representation of the \"studentSectionAssociation\" resource to be
updated.",
              "type": "studentSectionAssociation",
              "required": true
          }
        ],
        "consumes": [
          "application/json"
        ],
        "summary": "Updates or creates a resource based on the resource identifier.",
        "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
        "responseMessages": [
          {
            "code": 201,
            "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
          },
          {
            "code": 202,
            "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
          },
          {
            "code": 204,
            "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
          },
          {
            "code": 400,
            "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
          },
          {
            "code": 401,
            "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
          },
          {
            "code": 403,
            "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
          },
          {
            "code": 404,
            "message": "The resource could not be found."
          },
          {
            "code": 409,
            "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
          },
          {
            "code": 412,
            "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
          },
          {
            "code": 500,
            "message": "An unhandled error occurred on the server. See the response body for details.",
            "responseModel": "webServiceError"
          }
        ]
      },
      {
        "method": "DELETE",
        "nickname": "deleteStudentSectionAssociationById",
        "type": "void",
        "parameters": [
          {
            "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "studentSectionAssociation": {
      "id": "studentSectionAssociation",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "sectionReference": {
          "type": "sectionReference",
          "required": true,
          "description": "A reference to the related Section resource."
        },
        "studentReference": {
          "type": "studentReference",
          "required": true,
          "description": "A reference to the related Student resource."
        },
        "beginDate": {
          "type": "date-time",
          "required": true,
          "description": "Month, day, and year of the Student's entry or assignment to the Section."
        },
        "endDate": {
          "type": "date-time",
          "required": true,
          "description": "Month, day, and year of the withdrawal or exit of the Student from the Section."
        },
        "homeroomIndicator": {
          "type": "boolean",
          "required": true,
          "description": "Indicates the Section is the student's homeroom. Homeroom period may the convention
for taking daily attendance."
        },
        "repeatIdentifierType": {
          "type": "string",
          "required": false,
          "description": "An indication as to whether a student has previously taken a given course.
Repeated, counted in grade point average          Repeated, not counted in grade point average          Not
repeated          Other."
        },
        "teacherStudentDataLinkExclusion": {
          "type": "boolean",
          "required": false,
          "description": "Indicates that the student-section combination is excluded from calculation of value-
added or growth attribution calculations used for a particular teacher evaluation."
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "sectionReference": {
      "id": "sectionReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        },
        "classPeriodName": {
          "type": "string",
          "required": true,
          "description": "An indication of the portion of a typical daily session in which students receive
instruction in a specified subject (e.g., morning, sixth period, block period, or AB schedules)."
        },
        "classroomIdentificationCode": {
          "type": "string",
          "required": true,
          "description": "A unique number or alphanumeric code assigned to a room by a school, school system,
state, or other agency or entity."
```

```
        },
        "localCourseCode": {
          "type": "string",
          "required": true,
          "description": "The local code assigned by the School that identifies the course offering provided
for the instruction of students."
        },
        "termDescriptor": {
          "type": "string",
          "required": true,
          "description": "The term for the Session during the school year."
        },
        "schoolYear": {
          "type": "integer",
          "required": true,
          "description": "The identifier for the school year."
        },
        "uniqueSectionCode": {
          "type": "string",
          "required": true,
          "description": "A unique identifier for the Section that is defined by the classroom, the subjects
taught, and the instructors who are assigned."
        },
        "sequenceOfCourse": {
          "type": "integer",
          "required": true,
          "description": "When a section is part of a sequence of parts for a course, the number of the
sequence. If the course has only one part, the value of this section attribute should be 1."
        }
      }
    },
    "studentReference": {
      "id": "studentReference",
      "properties": {
        "studentUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a student."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/studentSectionAttendanceEvents

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/studentSectionAttendanceEvents",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/studentSectionAttendanceEvents",
      "description": "This event entity represents the recording of whether a student is in attendance for a
section.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentSectionAttendanceEventsAll",
          "type": "array",
          "items": {
            "$ref": "studentSectionAttendanceEvent"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentSectionAttendanceEvent"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentSectionAttendanceEventsByExample",
          "type": "array",
          "items": {
            "$ref": "studentSectionAttendanceEvent"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "attendanceEventCategoryDescriptor",
              "description": "A code describing the attendance event, for example:          Present
Unexcused absence          Excused absence          Tardy.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "classPeriodName",
              "description": "An indication of the portion of a typical daily session in which students receive
instruction in a specified subject (e.g., morning, sixth period, block period, or AB schedules).",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "classroomIdentificationCode",
              "description": "A unique number or alphanumeric code assigned to a room by a school, school
system, state, or other agency or entity.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "eventDate",
              "description": "Date for this attendance event.",
              "type": "date-time",
              "required": false
            },
            {
              "paramType": "query",
              "name": "localCourseCode",
              "description": "The local code assigned by the School that identifies the course offering
provided for the instruction of students.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "schoolYear",
              "description": "The identifier for the school year.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "sequenceOfCourse",
              "description": "When a section is part of a sequence of parts for a course, the number of the
sequence. If the course has only one part, the value of this section attribute should be 1.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "termDescriptor",
              "description": "The term for the Session during the school year.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "uniqueSectionCode",
              "description": "A unique identifier for the Section that is defined by the classroom, the
subjects taught, and the instructors who are assigned.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentSectionAttendanceEvent"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
```

```
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentSectionAttendanceEventByKey",
          "type": "studentSectionAttendanceEvent",
          "parameters": [
            {
              "paramType": "query",
              "name": "attendanceEventCategoryDescriptor",
              "description": "A code describing the attendance event, for example:          Present
Unexcused absence          Excused absence          Tardy.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "classPeriodName",
              "description": "An indication of the portion of a typical daily session in which students receive
instruction in a specified subject (e.g., morning, sixth period, block period, or AB schedules).",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "classroomIdentificationCode",
              "description": "A unique number or alphanumeric code assigned to a room by a school, school
system, state, or other agency or entity.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "eventDate",
              "description": "Date for this attendance event.",
              "type": "date-time",
              "required": true
            },
            {
              "paramType": "query",
              "name": "localCourseCode",
              "description": "The local code assigned by the School that identifies the course offering
provided for the instruction of students.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolId",
              "description": "The identifier assigned to a school by the State Education Agency (SEA).",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "schoolYear",
              "description": "The identifier for the school year.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "sequenceOfCourse",
              "description": "When a section is part of a sequence of parts for a course, the number of the
sequence. If the course has only one part, the value of this section attribute should be 1.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "termDescriptor",
              "description": "The term for the Session during the school year.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "uniqueSectionCode",
              "description": "A unique identifier for the Section that is defined by the classroom, the
subjects taught, and the instructors who are assigned.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentSectionAttendanceEvent"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postStudentSectionAttendanceEvents",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "studentSectionAttendanceEvent",
              "description": "The JSON representation of the \"studentSectionAttendanceEvent\" resource to be
created or updated.",
              "type": "studentSectionAttendanceEvent",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
```

```
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/studentSectionAttendanceEvents/{id}",
      "description": "This event entity represents the recording of whether a student is in attendance for a
section.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentSectionAttendanceEventsById",
          "type": "studentSectionAttendanceEvent",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentSectionAttendanceEvent"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
```

```
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putStudentSectionAttendanceEvent",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "studentSectionAttendanceEvent",
              "description": "The JSON representation of the \"studentSectionAttendanceEvent\" resource to be
updated.",
              "type": "studentSectionAttendanceEvent",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
```

```
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteStudentSectionAttendanceEventById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "studentSectionAttendanceEvent": {
      "id": "studentSectionAttendanceEvent",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "sectionReference": {
          "type": "sectionReference",
          "required": true,
          "description": "A reference to the related Section resource."
        },
        "studentReference": {
          "type": "studentReference",
          "required": true,
          "description": "A reference to the related Student resource."
        },
        "attendanceEventCategoryDescriptor": {
          "type": "string",
          "required": true,
          "description": "A code describing the attendance event, for example:          Present
```

```
Unexcused absence          Excused absence          Tardy."
        },
        "attendanceEventReason": {
          "type": "string",
          "required": false,
          "description": "The reported reason for a student's absence."
        },
        "educationalEnvironmentType": {
          "type": "string",
          "required": false,
          "description": "The setting in which a child receives education and related services. This attribute
is only used if it differs from the EducationalEnvironment of the Section. This is only used in the
AttendanceEvent if different from the associated Section."
        },
        "eventDate": {
          "type": "date-time",
          "required": true,
          "description": "Date for this attendance event."
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "sectionReference": {
      "id": "sectionReference",
      "properties": {
        "schoolId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to a school by the State Education Agency (SEA)."
        },
        "classPeriodName": {
          "type": "string",
          "required": true,
          "description": "An indication of the portion of a typical daily session in which students receive
instruction in a specified subject (e.g., morning, sixth period, block period, or AB schedules)."
        },
        "classroomIdentificationCode": {
          "type": "string",
          "required": true,
          "description": "A unique number or alphanumeric code assigned to a room by a school, school system,
state, or other agency or entity."
        },
        "localCourseCode": {
          "type": "string",
          "required": true,
          "description": "The local code assigned by the School that identifies the course offering provided
for the instruction of students."
        },
        "termDescriptor": {
          "type": "string",
          "required": true,
          "description": "The term for the Session during the school year."
        },
        "schoolYear": {
          "type": "integer",
          "required": true,
          "description": "The identifier for the school year."
        },
        "uniqueSectionCode": {
          "type": "string",
          "required": true,
          "description": "A unique identifier for the Section that is defined by the classroom, the subjects
taught, and the instructors who are assigned."
        },
        "sequenceOfCourse": {
          "type": "integer",
          "required": true,
          "description": "When a section is part of a sequence of parts for a course, the number of the
sequence. If the course has only one part, the value of this section attribute should be 1."
        }
      }
    },
    "studentReference": {
      "id": "studentReference",
      "properties": {
        "studentUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a student."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/studentSpecialEducationProgramAssociations

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/studentSpecialEducationProgramAssociations",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/studentSpecialEducationProgramAssociations",
      "description": "This association represents the special education program(s) that a student participates
in or receives services from. The association is an extension of the StudentProgramAssociation particular for
special education programs.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentSpecialEducationProgramAssociationsAll",
          "type": "array",
          "items": {
            "$ref": "studentSpecialEducationProgramAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentSpecialEducationProgramAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentSpecialEducationProgramAssociationsByExample",
          "type": "array",
          "items": {
            "$ref": "studentSpecialEducationProgramAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "beginDate",
              "description": "The month, day, and year on which the Student first received services.",
              "type": "date-time",
              "required": false
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "programEducationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "programName",
              "description": "The formal name of the Program of instruction, training, services, or benefits
available through federal, state, or local agencies.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "programType",
              "description": "The type of program.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentSpecialEducationProgramAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentSpecialEducationProgramAssociationByKey",
          "type": "studentSpecialEducationProgramAssociation",
          "parameters": [
            {
              "paramType": "query",
              "name": "beginDate",
              "description": "The month, day, and year on which the Student first received services.",
              "type": "date-time",
              "required": true
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "programEducationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "programName",
              "description": "The formal name of the Program of instruction, training, services, or benefits
available through federal, state, or local agencies.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "programType",
              "description": "The type of program.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentSpecialEducationProgramAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postStudentSpecialEducationProgramAssociations",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "studentSpecialEducationProgramAssociation",
              "description": "The JSON representation of the \"studentSpecialEducationProgramAssociation\"
resource to be created or updated.",
              "type": "studentSpecialEducationProgramAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/studentSpecialEducationProgramAssociations/{id}",
      "description": "This association represents the special education program(s) that a student participates
in or receives services from. The association is an extension of the StudentProgramAssociation particular for
special education programs.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentSpecialEducationProgramAssociationsById",
          "type": "studentSpecialEducationProgramAssociation",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentSpecialEducationProgramAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putStudentSpecialEducationProgramAssociation",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "studentSpecialEducationProgramAssociation",
              "description": "The JSON representation of the \"studentSpecialEducationProgramAssociation\"
resource to be updated.",
              "type": "studentSpecialEducationProgramAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
```

```
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteStudentSpecialEducationProgramAssociationById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "studentSpecialEducationProgramAssociation": {
      "id": "studentSpecialEducationProgramAssociation",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "educationOrganizationReference": {
          "type": "educationOrganizationReference",
          "required": true,
          "description": "A reference to the related EducationOrganization resource."
        },
        "programReference": {
          "type": "programReference",
          "required": true,
          "description": "A reference to the related Program resource."
        },
        "studentReference": {
          "type": "studentReference",
          "required": true,
          "description": "A reference to the related Student resource."
        },
        "beginDate": {
          "type": "date-time",
          "required": true,
          "description": "The month, day, and year on which the Student first received services."
        },
        "endDate": {
          "type": "date-time",
          "required": true,
          "description": "The month, day, and year on which the Student exited the Program or stopped receiving
services."
        },
        "ideaEligibility": {
          "type": "boolean",
          "required": true,
          "description": "Indicator of the eligibility of the student to receive special education services
according to the Individuals with Disabilities Education Act (IDEA)."
        },
        "iepBeginDate": {
          "type": "date-time",
          "required": false,
          "description": "The effective date of the most recent IEP."
        },
        "iepEndDate": {
          "type": "date-time",
          "required": false,
          "description": "The end date of the most recent IEP."
        },
        "iepReviewDate": {
          "type": "date-time",
          "required": false,
          "description": "The date of the last IEP review."
        },
        "lastEvaluationDate": {
          "type": "date-time",
          "required": false,
          "description": "The date of the last special education evaluation."
        },
        "medicallyFragile": {
          "type": "boolean",
          "required": false,
          "description": "Indicates whether the Student receiving special education and related services
is:          1) in the age range of birth to 22 years, and          2) has a serious, ongoing illness or a
chronic condition that has lasted or is anticipated to last at least 12 or more months or has required at least
one month of hospitalization, and that requires daily, ongoing medical treatments and monitoring by
appropriately trained personnel which may include parents or other family members, and          3) requires the
routine use of medical device or of assistive technology to compensate for the loss of usefulness of a body
function needed to participate in activities of daily living, and          4) lives with ongoing threat to his
or her continued well-being.          Aligns with federal requirements."
        },
        "multiplyDisabled": {
          "type": "boolean",
          "required": false,
          "description": "Indicates whether the Student receiving special education and related services has
been designated as multiply disabled by the admission, review, and dismissal committee as aligned with federal
requirements."
        },
        "reasonExitedDescriptor": {
          "type": "string",
          "required": false,
          "description": "The reason the child left the Program within a school or district."
        },
        "schoolHoursPerWeek": {
          "type": "number",
          "required": false,
          "description": "Indicate the total number of hours of instructional time per week for the school that
the student attends."
        },
        "servedOutsideOfRegularSession": {
          "type": "boolean",
          "required": false,
          "description": "Indicates whether the Student received services during the summer session or between
sessions."
        },
        "specialEducationHoursPerWeek": {
          "type": "number",
          "required": false,
          "description": "The number of hours per week for special education instruction and therapy."
        },
        "specialEducationSettingDescriptor": {
          "type": "string",
          "required": true,
          "description": "The major instructional setting (more than 50 percent of a student's special
education program)."
        },
        "serviceProviders": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of
studentSpecialEducationProgramAssociationServiceProviders.  The Staff providing special education services to
the Student.",
          "items": {
            "$ref": "studentSpecialEducationProgramAssociationServiceProvider"
          }
        },
        "services": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of studentProgramAssociationServices.  Indicates the Service
(s) being provided to the Student by the Program.",
          "items": {
            "$ref": "studentProgramAssociationService"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "educationOrganizationReference": {
      "id": "educationOrganizationReference",
      "properties": {
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        }
      }
    },
    "programReference": {
      "id": "programReference",
      "properties": {
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        },
        "type": {
          "type": "string",
          "required": true,
          "description": "The type of program."
        },
        "name": {
          "type": "string",
          "required": true,
          "description": "The formal name of the Program of instruction, training, services, or benefits
available through federal, state, or local agencies."
        }
      }
    },
    "studentReference": {
      "id": "studentReference",
      "properties": {
        "studentUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a student."
        }
      }
    },
    "studentSpecialEducationProgramAssociationServiceProvider": {
      "id": "studentSpecialEducationProgramAssociationServiceProvider",
      "properties": {
        "staffReference": {
          "type": "staffReference",
          "required": true,
          "description": "A reference to the related Staff resource."
        },
        "primaryProvider": {
          "type": "boolean",
          "required": false,
          "description": "Primary ServiceProvider."
        }
      }
    },
    "studentProgramAssociationService": {
      "id": "studentProgramAssociationService",
      "properties": {
        "serviceDescriptor": {
          "type": "string",
          "required": true,
          "description": "Indicates the Service being provided to the student by the Program."
        },
        "primaryIndicator": {
          "type": "boolean",
          "required": false,
          "description": "True if service is a primary service."
        },
        "serviceBeginDate": {
          "type": "date-time",
          "required": false,
          "description": "First date the Student was in this option for the current school year."
        },
        "serviceEndDate": {
          "type": "date-time",
          "required": false,
          "description": "Last date the Student was in this option for the current school year."
        }
      }
    },
    "staffReference": {
      "id": "staffReference",
      "properties": {
        "staffUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a staff."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```

/studentTitleIPartAProgramAssociations

```
{
  "apiVersion": "0.95",
  "swaggerVersion": "1.2",
  "basePath": "http://localhost:54746/api/v2.0/2017",
  "resourcePath": "/studentTitleIPartAProgramAssociations",
  "produces": [
    "application/json"
  ],
  "apis": [
    {
      "path": "/studentTitleIPartAProgramAssociations",
      "description": "This association represents the Title I Part A program(s) that a student participates in
or from which the Student receives services. The association is an extension of the StudentProgramAssociation
particular for Title I Part A programs.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentTitleIPartAProgramAssociationsAll",
          "type": "array",
          "items": {
            "$ref": "studentTitleIPartAProgramAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources based with paging capabilities (using the \"Get All\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get All\" pattern. In this
version of the API there is support for paging.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The matching resource(s) were successfully retrieved.  If no instances are found will
return an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentTitleIPartAProgramAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentTitleIPartAProgramAssociationsByExample",
          "type": "array",
          "items": {
            "$ref": "studentTitleIPartAProgramAssociation"
          },
          "parameters": [
            {
              "paramType": "query",
              "name": "offset",
              "description": "Indicates how many items should be skipped before returning results.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "limit",
              "description": "Indicates the maximum number of items that should be returned in the results
(defaults to 25).",
              "type": "integer",
              "required": false,
              "minimum": 1,
              "maximum": 250
            },
            {
              "paramType": "query",
              "name": "beginDate",
              "description": "The month, day, and year on which the Student first received services.",
              "type": "date-time",
              "required": false
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "programEducationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": false
            },
            {
              "paramType": "query",
              "name": "programName",
              "description": "The formal name of the Program of instruction, training, services, or benefits
available through federal, state, or local agencies.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "programType",
              "description": "The type of program.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves resources matching values of an example resource (using the \"Get By Example\"
pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Example\" search
pattern.  The values of any properties of the resource that are specified will be used to return all matching
results (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource(s) were successfully retrieved.  If no instances are found will return
an empty collection.",
              "responseModel": "array",
              "items": {
                "$ref": "studentTitleIPartAProgramAssociation"
              }
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors.  This will typically be an issue with the query parameters or their values."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "GET",
          "nickname": "getStudentTitleIPartAProgramAssociationByKey",
          "type": "studentTitleIPartAProgramAssociation",
          "parameters": [
            {
              "paramType": "query",
              "name": "beginDate",
              "description": "The month, day, and year on which the Student first received services.",
              "type": "date-time",
              "required": true
            },
            {
              "paramType": "query",
              "name": "educationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "programEducationOrganizationId",
              "description": "The identifier assigned to an education agency by the State Education Agency
(SEA).  Also known as the State LEA ID.",
              "type": "integer",
              "required": true
            },
            {
              "paramType": "query",
              "name": "programName",
              "description": "The formal name of the Program of instruction, training, services, or benefits
available through federal, state, or local agencies.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "programType",
              "description": "The type of program.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "query",
              "name": "studentUniqueId",
              "description": "A unique alphanumeric code assigned to a student.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the values of the resource's natural key (using the \"
Get By Key\" pattern).",
          "notes": "This GET operation provides access to resources using the \"Get by Key\" search pattern.
The values of the natural key of the resource must be fully specified, and the service will return the matching
result (if it exists).",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentTitleIPartAProgramAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "POST",
          "nickname": "postStudentTitleIPartAProgramAssociations",
          "type": "void",
          "parameters": [
            {
              "paramType": "body",
              "name": "studentTitleIPartAProgramAssociation",
              "description": "The JSON representation of the \"studentTitleIPartAProgramAssociation\" resource
to be created or updated.",
              "type": "studentTitleIPartAProgramAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Creates or updates resources based on the natural key values of the supplied resource.",
          "notes": "The POST operation can be used to create or update resources. In database terms, this is
often referred to as an \"upsert\" operation (insert + update).  Clients should NOT include the resource \"id\"
in the JSON body because it will result in an error (you must use a PUT operation to update a resource by \"
id\"). The web service will identify whether the resource already exists based on the natural key values
provided, and update or create the resource appropriately.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not
yet completed due to current system load. Processing may still fail due to violation of referential integrity
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    },
    {
      "path": "/studentTitleIPartAProgramAssociations/{id}",
      "description": "This association represents the Title I Part A program(s) that a student participates in
or from which the Student receives services. The association is an extension of the StudentProgramAssociation
particular for Title I Part A programs.",
      "operations": [
        {
          "method": "GET",
          "nickname": "getStudentTitleIPartAProgramAssociationsById",
          "type": "studentTitleIPartAProgramAssociation",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be retrieved.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-None-Match",
              "description": "The previously returned ETag header value, used here to prevent the unnecessary
data transfer of an unchanged resource.",
              "type": "string",
              "required": false
            }
          ],
          "produces": [
            "application/json"
          ],
          "summary": "Retrieves a specific resource using the resource's identifier (using the \"Get By Id\"
pattern).",
          "notes": "This GET operation retrieves a resource by the specified resource identifier.",
          "responseMessages": [
            {
              "code": 200,
              "message": "The resource was successfully retrieved.",
              "responseModel": "studentTitleIPartAProgramAssociation"
            },
            {
              "code": 304,
              "message": "The resource's current server-side ETag value matched the If-None-Match header value
supplied with the request indicating the resource has not been modified."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "PUT",
          "nickname": "putStudentTitleIPartAProgramAssociation",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be updated.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the PUT from updating a resource modified 
by another consumer.",
              "type": "string",
              "required": false
            },
            {
              "paramType": "body",
              "name": "studentTitleIPartAProgramAssociation",
              "description": "The JSON representation of the \"studentTitleIPartAProgramAssociation\" resource 
to be updated.",
              "type": "studentTitleIPartAProgramAssociation",
              "required": true
            }
          ],
          "consumes": [
            "application/json"
          ],
          "summary": "Updates or creates a resource based on the resource identifier.",
          "notes": "The PUT operation is used to update or create a resource by identifier.  If the resource 
doesn't exist, the resource will be created using that identifier.  Additionally, natural key values cannot be 
changed using this operation, and will not be modified in the database.  If the resource \"id\" is provided in 
the JSON body, it will be ignored as well.",
          "responseMessages": [
            {
              "code": 201,
              "message": "The resource was created.  An ETag value is available in the ETag header, and the 
location of the resource is available in the Location header of the response."
            },
            {
              "code": 202,
              "message": "The resource has been validated and accepted by the service, but processing has not 
yet completed due to current system load. Processing may still fail due to violation of referential integrity 
requirements."
            },
            {
              "code": 204,
              "message": "The resource was updated.  An updated ETag value is available in the ETag header of 
the response."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body 
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was 
either not provided or is invalid.  The operation may succeed once authenication has been successfully 
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.  
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid 
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match 
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        },
        {
          "method": "DELETE",
          "nickname": "deleteStudentTitleIPartAProgramAssociationById",
          "type": "void",
          "parameters": [
            {
              "paramType": "path",
              "name": "id",
              "description": "A resource identifier specifying the resource to be deleted.",
              "type": "string",
              "required": true
            },
            {
              "paramType": "header",
              "name": "If-Match",
              "description": "The ETag header value used to prevent the DELETE from removing a resource 
modified by another consumer.",
              "type": "string",
              "required": false,
              "allowMultiple": false
            }
          ],
          "summary": "Deletes an existing resource using the resource identifier.",
          "notes": "The DELETE operation is used to delete an existing resource by identifier.  If the resource 
doesn't exist, an error will result (the resource will not be found).",
          "responseMessages": [
            {
              "code": 202,
              "message": "The request has accepted by the service, but processing has not yet completed due to 
current system load. Processing may still fail due to violation of referential integrity requirements."
            },
            {
              "code": 204,
              "message": "The resource was successfully deleted."
            },
            {
              "code": 400,
              "message": "Bad Request.  The request was invalid and cannot be completed.  See the response body 
for specific validation errors."
            },
            {
              "code": 401,
              "message": "Unauthorized.  The request requires authentication.  The OAuth bearer token was
either not provided or is invalid.  The operation may succeed once authenication has been successfully
completed."
            },
            {
              "code": 403,
              "message": "Forbidden.  The request cannot be completed in the current authorization context.
Contact your administrator if you believe this operation should be allowed."
            },
            {
              "code": 404,
              "message": "The resource could not be found."
            },
            {
              "code": 409,
              "message": "Conflict.  The request cannot be completed because it would result in an invalid
state.  See the response body for details."
            },
            {
              "code": 412,
              "message": "The resource's current server-side ETag value does not match the supplied If-Match
header value in the request.  This indicates the resource has been modified by another consumer."
            },
            {
              "code": 500,
              "message": "An unhandled error occurred on the server. See the response body for details.",
              "responseModel": "webServiceError"
            }
          ]
        }
      ]
    }
  ],
  "models": {
    "studentTitleIPartAProgramAssociation": {
      "id": "studentTitleIPartAProgramAssociation",
      "properties": {
        "id": {
          "type": "string",
          "required": true,
          "description": "The unique identifier of the resource."
        },
        "educationOrganizationReference": {
          "type": "educationOrganizationReference",
          "required": true,
          "description": "A reference to the related EducationOrganization resource."
        },
        "programReference": {
          "type": "programReference",
          "required": true,
          "description": "A reference to the related Program resource."
        },
        "studentReference": {
          "type": "studentReference",
          "required": true,
          "description": "A reference to the related Student resource."
        },
        "beginDate": {
          "type": "date-time",
          "required": true,
          "description": "The month, day, and year on which the Student first received services."
        },
        "endDate": {
          "type": "date-time",
          "required": true,
          "description": "The month, day, and year on which the Student exited the Program or stopped receiving
services."
        },
        "reasonExitedDescriptor": {
          "type": "string",
          "required": false,
          "description": "The reason the child left the Program within a school or district."
        },
        "servedOutsideOfRegularSession": {
          "type": "boolean",
          "required": false,
          "description": "Indicates whether the Student received services during the summer session or between
sessions."
        },
        "titleIPartAParticipantType": {
          "type": "string",
          "required": true,
          "description": "An indication of the type of Title I program, if any, in which the student is
participating and by which the student is served:          Public Targeted Assistance Program           Public
Schoolwide Program          Private School Students Participating          Local Neglected Program."
        },
        "services": {
          "type": "array",
          "required": false,
          "description": "An unordered collection of studentProgramAssociationServices.  Indicates the Service
(s) being provided to the Student by the Program.",
          "items": {
            "$ref": "studentProgramAssociationService"
          }
        },
        "_etag": {
          "type": "string",
          "required": false,
          "description": "A unique system-generated value that identifies the version of the resource."
        }
      }
    },
    "educationOrganizationReference": {
      "id": "educationOrganizationReference",
      "properties": {
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        }
      }
    },
    "programReference": {
      "id": "programReference",
      "properties": {
        "educationOrganizationId": {
          "type": "integer",
          "required": true,
          "description": "The identifier assigned to an education agency by the State Education Agency (SEA).
Also known as the State LEA ID."
        },
        "type": {
          "type": "string",
          "required": true,
          "description": "The type of program."
        },
        "name": {
          "type": "string",
          "required": true,
          "description": "The formal name of the Program of instruction, training, services, or benefits
available through federal, state, or local agencies."
        }
      }
    },
    "studentReference": {
      "id": "studentReference",
      "properties": {
        "studentUniqueId": {
          "type": "string",
          "required": true,
          "description": "A unique alphanumeric code assigned to a student."
        }
      }
    },
    "studentProgramAssociationService": {
      "id": "studentProgramAssociationService",
      "properties": {
        "serviceDescriptor": {
          "type": "string",
          "required": true,
          "description": "Indicates the Service being provided to the student by the Program."
        },
        "primaryIndicator": {
          "type": "boolean",
          "required": false,
          "description": "True if service is a primary service."
        },
        "serviceBeginDate": {
          "type": "date-time",
          "required": false,
          "description": "First date the Student was in this option for the current school year."
        },
        "serviceEndDate": {
          "type": "date-time",
          "required": false,
          "description": "Last date the Student was in this option for the current school year."
        }
      }
    },
    "webServiceError": {
      "id": "webServiceError",
      "properties": {
        "message": {
          "type": "string",
          "required": false,
          "description": "The \"user-friendly\" error message."
        },
        "exceptionMessage": {
          "type": "string",
          "required": false,
          "description": "The system-generated exception message."
        },
        "exceptionType": {
          "type": "string",
          "required": false,
          "description": "The type of the exception."
        },
        "stackTrace": {
          "type": "string",
          "required": false,
          "description": "The server-side stack trace (only available in DEBUG builds)."
        }
      }
    }
  }
}
```