# Product Integrations

## Help Guide

September 2023

Anthology Inc.

5201 Congress Avenue
Boca Raton, FL 33487
Main: +1.561.923.2500
Support: +1.800.483.9106
www.anthology.com

# Revision History

| Rev. | Date | Description |
|---|---|---|
| 01 | Sep. 2023 | Initial release of document for Anthology Product Integrations.<br>See What's New. |

# Contents

# Product Integrations

Anthology products can be integrated using various technologies. The integration strategies have evolved as the products matured.

For Anthology products deployed in the Azure cloud, the following integration technologies are in use:

- **Azure API Management**. APIs are published securely for internal and external developers to use when connecting systems.
- **Azure Logic Apps**. Workflows are created to connect services in the cloud and on-premises.
- **Azure Service Bus**. Topics are published for subscription to send messages between on-premises and cloud-based applications and services.
- **Azure Event Grid**. Connect supported Azure and third-party services while simplifying event-based app development.
- **Azure Functions**. Simplify complex orchestration problems with an event-driven serverless compute platform.
- **Azure Data Factory**. Visually integrate data sources to accelerate data transformation and support enterprise workflows.

# What's New

Initial Release: September 2023

- Azure Service Bus & APIs
- Product Integrations:
  - Anthology Reach & Anthology Student
  - Anthology Student & Custom Payment Gateway Provider
- Resources

# Azure Service Bus & APIs

The main objective of Anthology's integration architecture is to separate release-specific code for Anthology products and client-specific extension code from the integration functions so that integrations continue to function when product code or extension code is updated.

The integration architecture can be described based on two models: asynchronous and synchronous.

## Asynchronous Integration Using Azure Service Bus

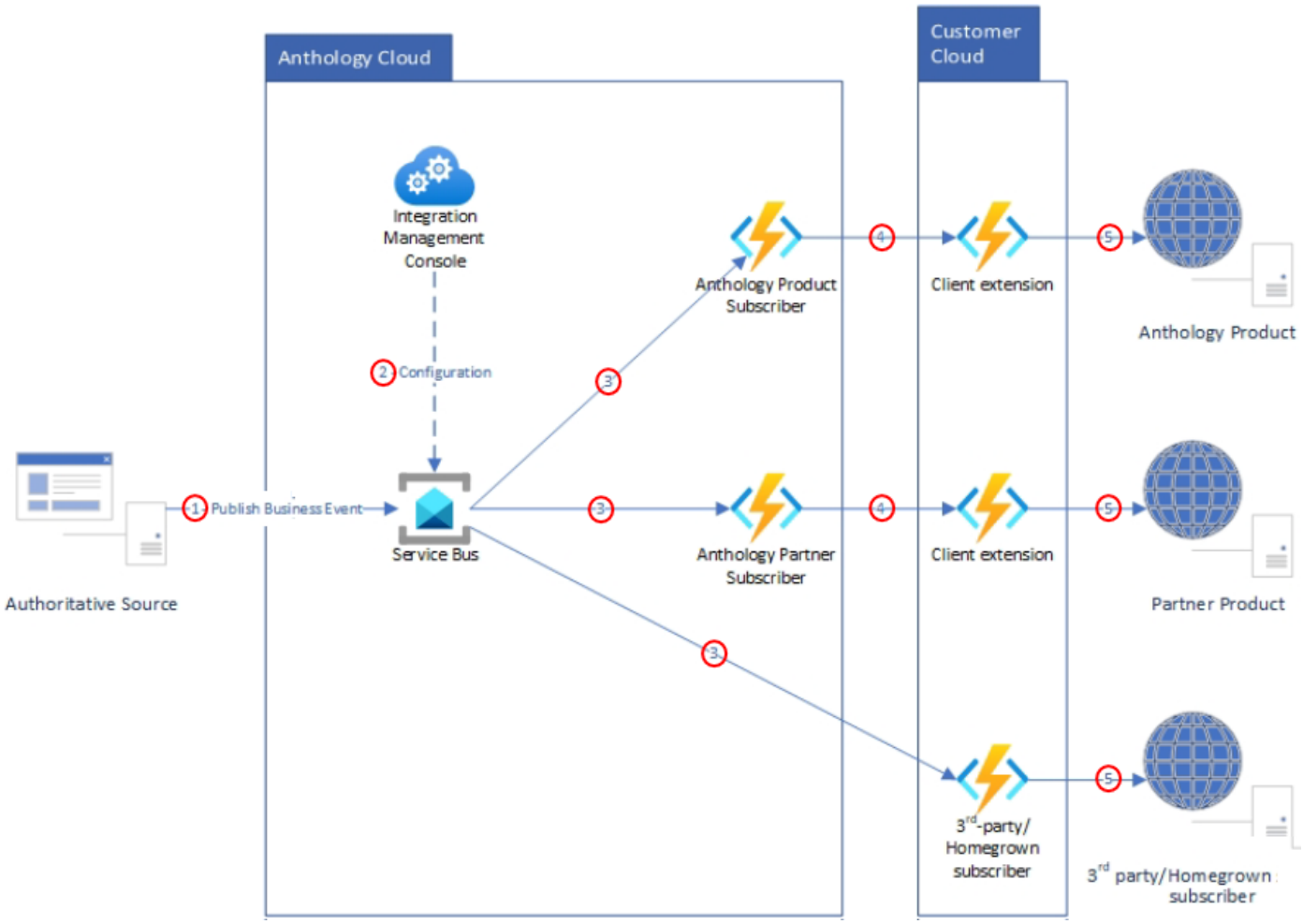
The asynchronous model is a "publish-subscribe" model wherein an Anthology source product publishes business events to the Azure Service Bus and any system attached to the customer's cloud can subscribe to the events. This is basically a one-way message flow, i.e., the source system does not expect a response from the target systems.

Anthology's roadmap includes expanding its product integration capabilities using the Azure Service Bus.

What is Azure Service Bus?

Azure Service Bus is a fully managed enterprise message broker with message queues and publish-subscribe topics (in a namespace). Service Bus is used to decouple applications and services from each other, providing the following benefits:

- Load-balancing work across competing workers
- Safely routing and transferring data and control across service and application boundaries
- Coordinating transactional work that requires a high-degree of reliability

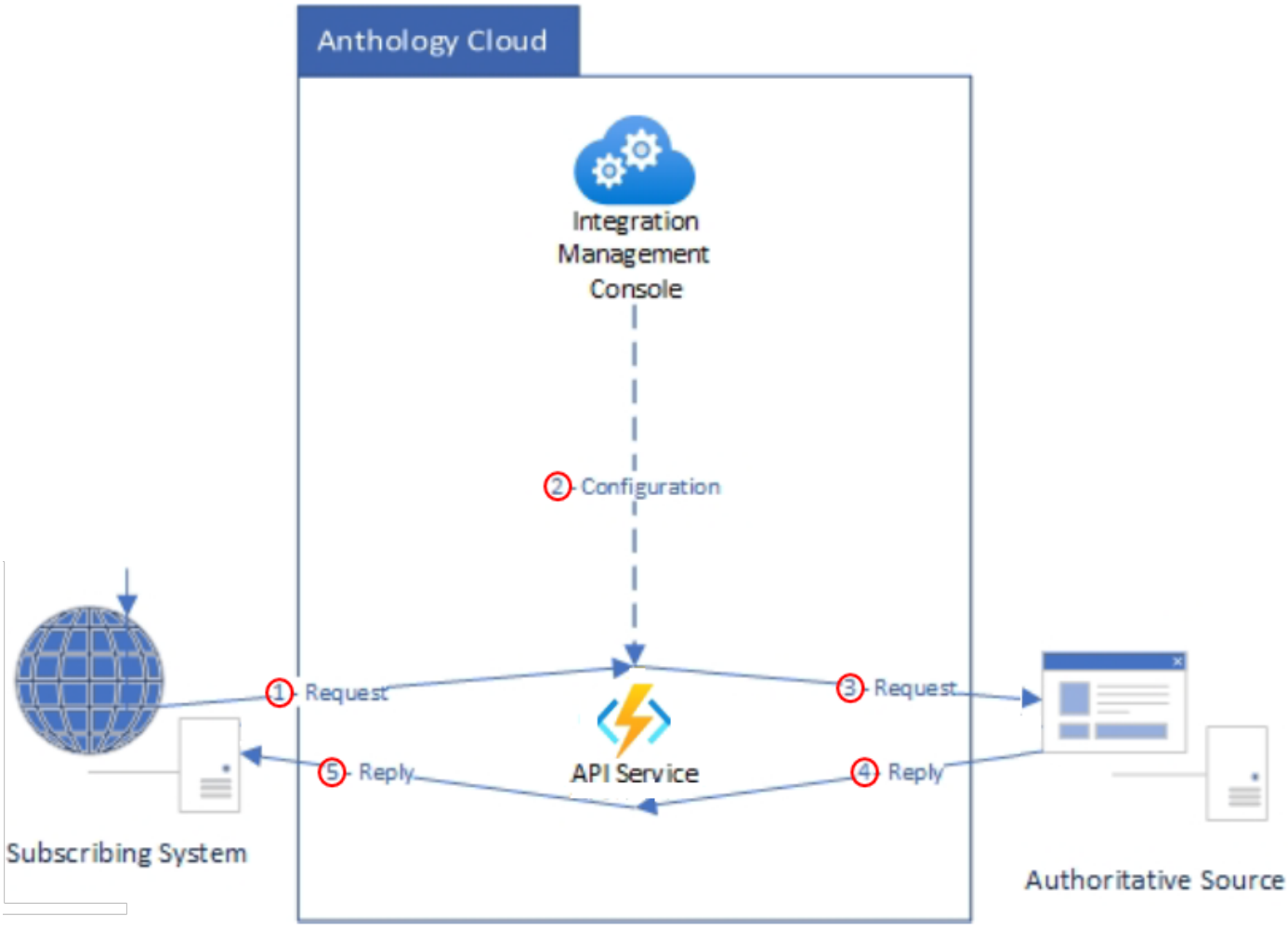



1. The authoritative source publishes a business event to the Service Bus.
2. The Service Bus uses configuration information from the Integration Management Console to validate subscribers.
3. Anthology product or partner functions retrieve messages from the Service Bus and process business logic.
4. Anthology product functions pass messages to client extension functions for client-specific business logic.
5. Messages are published to the subscribing system.

## Synchronous Integration Using APIs

The synchronous model introduces an API Service that handles incoming and outgoing messages between the source and target systems. The API Service responds to requests from the subscribing systems. Based on the configuration in the Integration Management Console, the API Service identifies the authoritative source and determines which subscribing system is to receive the reply.

Anthology APIs are published securely for internal and external developers.




1. The subscribing system sends API requests to the API Service.
2. The API Service uses configuration information from Integration Management Console to find the authoritative source.
3. The API Service sends API requests to the authoritative source.
4. Responses are passed back to the API Service.
5. The API Service receives responses and sends them to the subscribing system.

# Product Integrations

Anthology Reach & Anthology Student

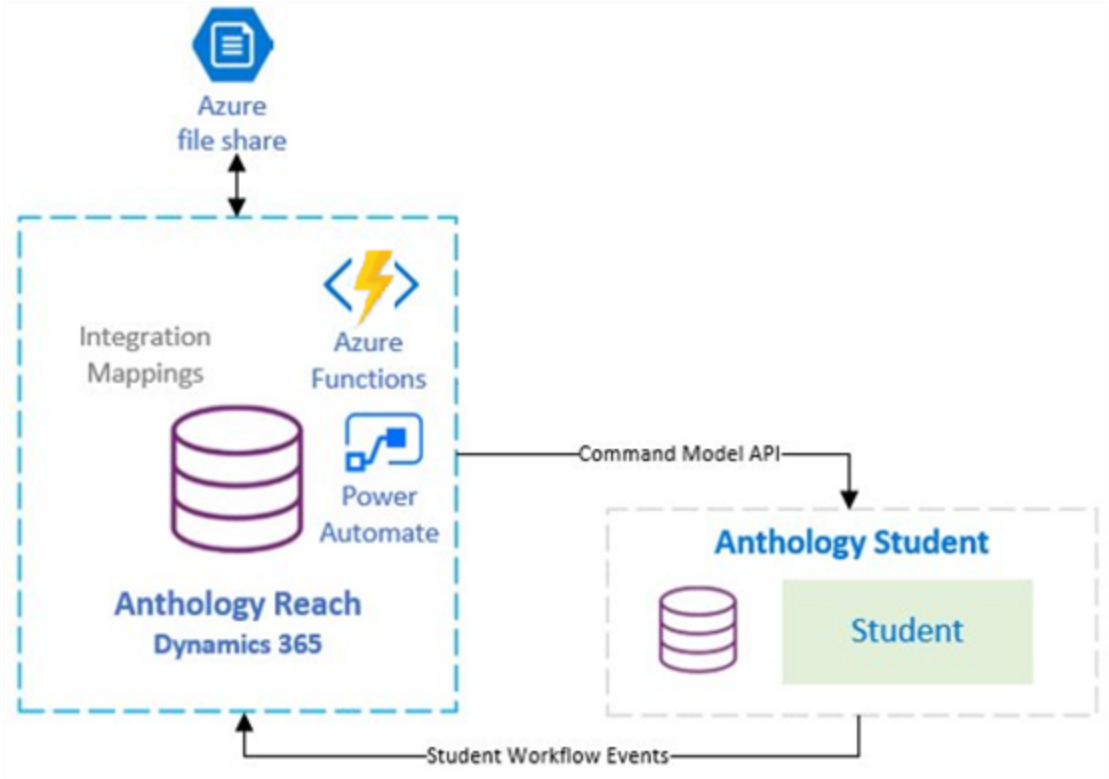
Anthology Student and Custom Payment Gateway Provider

# Anthology Reach & Anthology Student

## Integration Modalities

The integration between Anthology Reach and Anthology Student can be accomplished using the following three modalities:

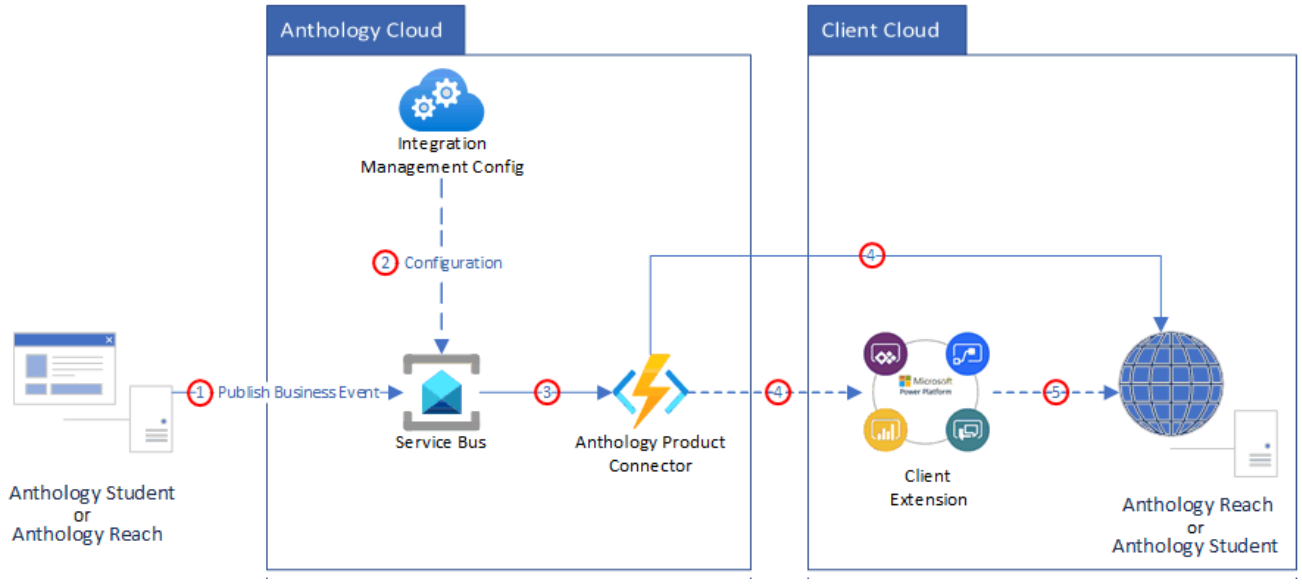
1. **Flow-based Data Integration** — Data integration via Microsoft Power Automate flows and Anthology Student events and workflows.
    - Anthology Student events triggers workflows in response to record changes in the Anthology Student database.
    - Power Automate flows interact with the Command Model APIs of Anthology Student.




    Flow-based data integration is currently available for numerous entities in Anthology Reach and Anthology Student and can be deployed as needed by clients.

    For more information, see Flow-based Integration.

2. **Service Bus Integration** — Data integration via subscribed business events using the Azure Service Bus functionality.




1. The source system (Anthology Student or Anthology Reach) publishes business events that clients can subscribe to.
2. The Integration Management Console enables configuration of the Azure Service Bus as well as failure management (dead letter queues) and logging.
3. The Azure Service Bus invokes the Anthology Product Connector which provides the Azure Functions and/or logic apps required for the integration.
4. The Anthology Product Connector transfers data from the Anthology cloud to the client cloud.
5. If client extensions exist, the Anthology Product Connector interfaces with the client extensions. The client extensions update the target system (Anthology Student or Anthology Reach).

Initially, Service Bus integrations will be available only for a few business events. More business events will be added over time so that Service Bus integrations can eventually replace flow-based integrations.

For more information, see Service Bus Integration.

3. **Hybrid Integration** — Integration using both the legacy flow-based data integration and the new Service Bus integration. Depending on the client's data integration requirements, flow-based and Service Bus integration methods can be deployed simultaneously.

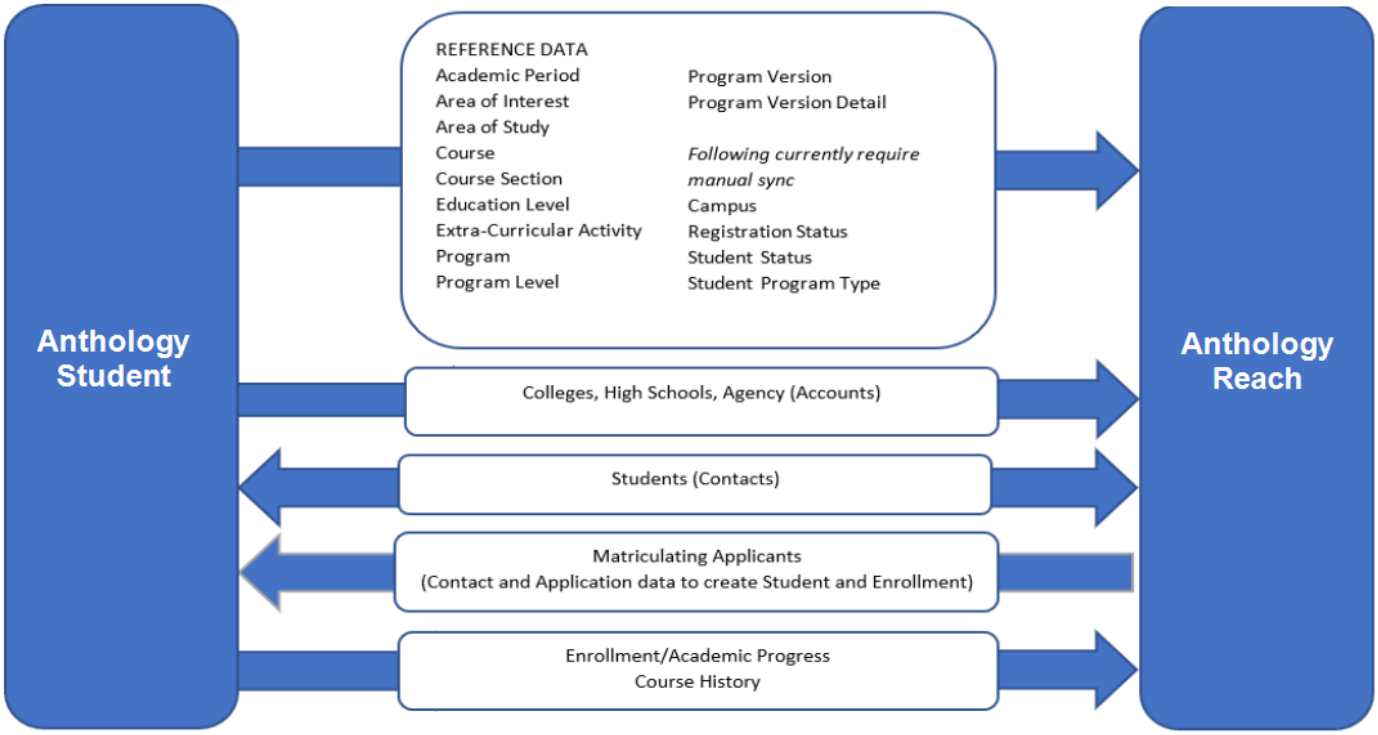
   For more information, see Deployment.

## Flow-based Integration

The flow-based integration between Anthology Student and Anthology Reach uses:

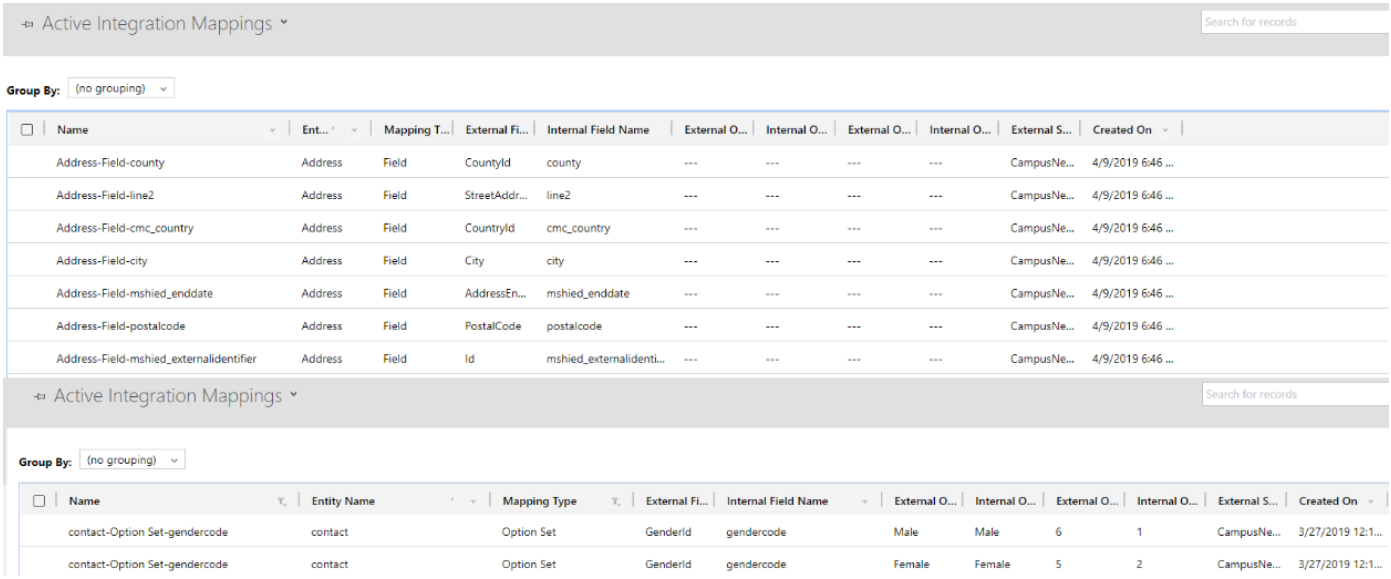
- Anthology Workflow events to integrate data from Anthology Student to Anthology Reach
- Microsoft Flows on the Anthology Reach system and command model APIs of Anthology Student to integrate data from Anthology Reach to Anthology Student

The data flow between Anthology Student and Anthology Reach is bidirectional for fully integrated entities such as Student and Contact. The data transfer is unidirectional for Reference data such as Programs, Academic Periods, and Course History. Reference data can be used as a look-up field within other entities. Some Reference data requires manual synchronization between the systems.



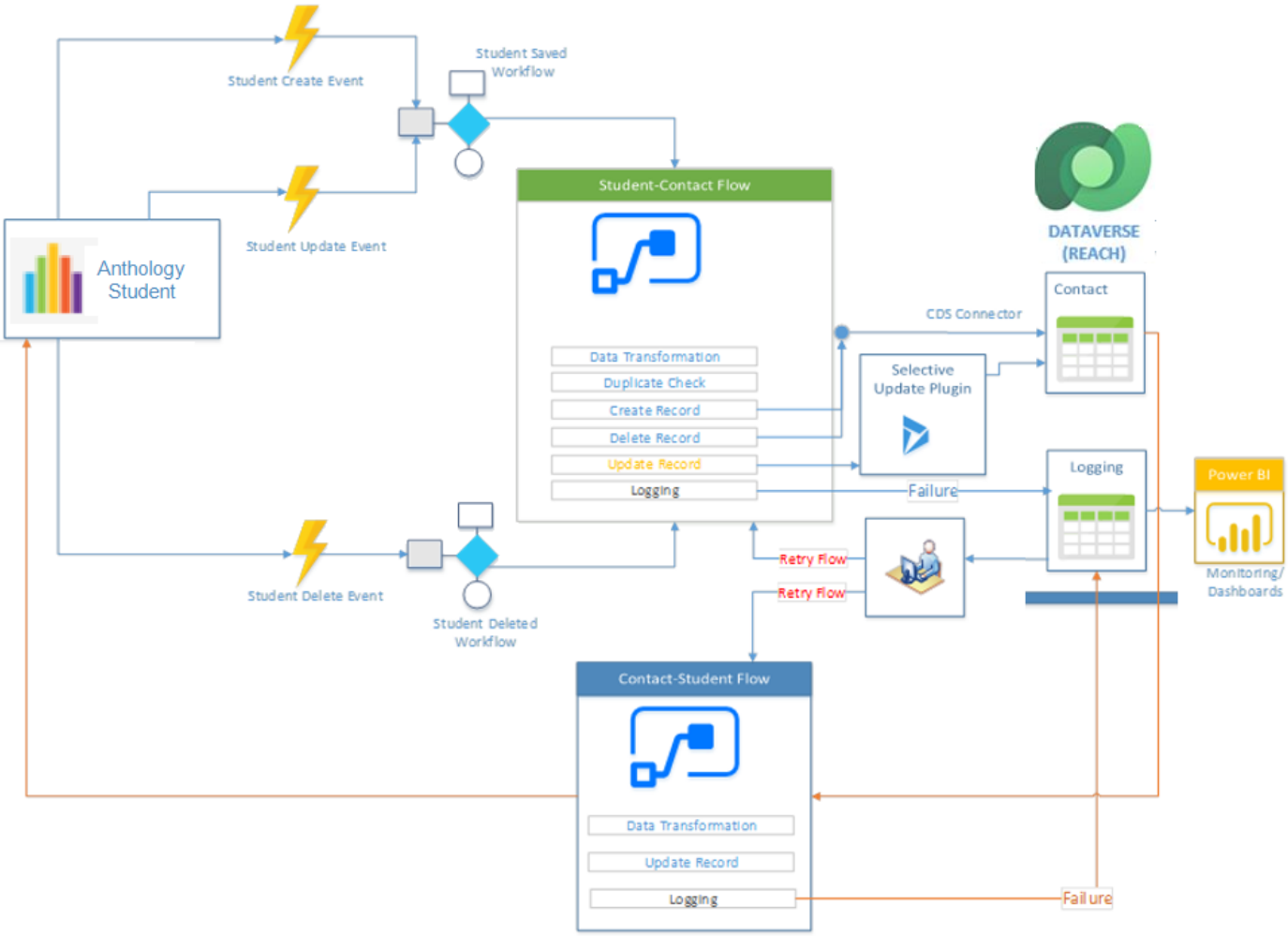

For information on the entities integrated between Anthology Student and Anthology Reach, see Integrated Entities.

Entity Integration Mappings are used to create Option Set mappings for the Option set fields which are integrated. See Import Integration Mapping Templates.




## Data Integration Process Flow

The following diagram illustrates the process flow for integrating data between Anthology Student and Anthology Reach.




The data integration between Anthology Student and Anthology Reach process flow is as follows:

1. Workflows for an Anthology Student entity are created based on saved and deleted events.
2. Workflows post the entity data to the corresponding Power Automate flow by sending an HTTP request to the flow.
3. The Power Automate flow, which is configured to trigger on the HTTP request, captures the Anthology Student entity data, executes the required data transformations of Lookup and Option Set mappings, and pushes the data into Anthology Reach via Common Data Service (CDS) connectors.
4. The flow identifies if an integration record already exists in Anthology Reach based on the external identifier field which maps to the ID record in Anthology Student.
5. After resolving the record, a Create or Update command is issued.
6. The Integration Log entity in the system logs the status of the integration message.

### Anthology Reach Integration Flows

The following table lists current Power Automate flows for Anthology Reach and associated Anthology Student workflows.

| Anthology Reach Flow | Anthology Student Workflow |
|---|---|
| CNS-CNE AgencyBranch-Account Integration (Engage) | CNS Integration AgencyBranch Saved<br>CNS Integration AgencyBranch Deleted |
| CNS-CNE College-Account Integration (Engage) | CNS Integration College Saved<br>CNS Integration College Deleted |
| CNS-CNE Concentration-AreaOfStudy Integration (Engage) | CNS Integration Concentration Saved<br>CNS Integration Concentration Deleted |
| CNS-CNE ClassSection-CourseSection Integration (Engage) | CNS Integration ClassSection Saved<br>CNS Integration ClassSection Deleted |
| CNS-CNE Course Integration (Engage) | CNS Integration Course Saved<br>CNS Integration Course Deleted |
| CNS-CNE ExtraCurricularActivities Integration (Engage) | CNS Integration ExtraCurricularActivities Saved<br>CNS Integration ExtraCurricularActivities Deleted |
| CNS-CNE HighSchool-Account Integration (Engage) | CNS Integration HighSchool Saved<br>CNS Integration HighSchool Deleted |
| CNS-CNE PreviousEducation-EducationLevel Integration (Engage) | CNS Integration PreviousEducation Saved<br>CNS Integration PreviousEducation Deleted |
| CNS-CNE Program Integration (Engage) | CNS Integration Program Saved<br>CNS Integration Program Deleted |
| CNS-CNE ProgramGroup-AreaOfInterest Integration (Engage) | CNS Integration ProgramGroup Saved<br>CNS Integration ProgramGroup Deleted |
| CNS-CNE ProgramLevel-Degree Integration (Engage) | CNS Integration ProgramLevel Saved<br>CNS Integration ProgramLevel Deleted |
| CNS-CNE ProgramVersionDetail-StartDate Integration (Engage) | CNS Integration ProgramVersionDetail Saved<br>CNS Integration ProgramVersionDetail Deleted |
| CNS-CNE ProgramVersion Integration (Engage) | CNS Integration ProgramVersion Saved<br>CNS Integration ProgramVersion Deleted |

| Anthology Reach Flow | Anthology Student Workflow |
|---|---|
| CNS-CNE Shift Integration (Engage) | CNS Integration Shift Saved<br>CNS Integration Shift Deleted |
| CNS-CNE Student-Contact Integration (Engage) | CNS Integration Student Saved<br>CNS Integration Student Deleted<br>CNS Integration Student(Person) Saved |
| CNS-CNE StudentGroup - Group Integration (Engage) | CNS Integration Student Group Saved<br>CNS Integration Student Group Deleted |
| CNS-CNE Student GroupMembership - GroupMembership Integration (Engage) | CNS Integration Student Group Membership - Add Students To Group<br>CNS Integration Student Group Membership - Add Student To Group<br>CNS Integration Student Group Membership - Remove Student From Group<br>CNS Integration Student Group Membership - Remove Students From Group |
| CNS-CNE StudentPreviousEducation-Previous Education Integration (Engage) | CNS Integration StudentPreviousEducation Saved<br>CNS Integration StudentPreviousEducation Deleted |
| CNS-CNE StudentRelationshipAddress-Address Integration (Engage) | CNS Integration StudentRelationshipAddress Saved<br>CNS Integration StudentRelationshipAddress Deleted |
| CNS-CNE Term-AcademicPeriod Integration (Engage) | CNS Integration Term Saved<br>CNS Integration Term Deleted |
| CNS-CNE Student Enrollment Period - Enrollment Integration (Engage) | CNS Integration StudentEnrollmentPeriod Deleted<br>CNS Integration StudentEnrollmentPeriod Saved<br>CNS Integration StudentEnrollmentPeriod EnrollmentStatusChange |

Integration flows are added in the periodic Anthology Reach releases and can be customized as needed. For more details, see Flows and Procedures.

### Integration Prerequisites

The flow-based integration betweenAnthology Student and Anthology Reach requires the following configurations and mappings:

- Configurations for the Contact Entity
- Configurations for the Course Section Entity

- Picklist Mappings

**Configurations for the Contact Entity**

Update the Field Mappings for the Source Category and Source Method Fields

1. In Anthology Reach, navigate to **Settings** > **Integrations** > **Integration Mappings**.
2. Locate the **contact-Field-_cmc_sourcemethodid_value** field and view the fields on the **General** tab.
3. In the **Parameters** field, add the source method name that is used for integrating the Contact data.

   *Example*: If the external source method name is *Email*, update the value in the**Parameters** fields as follows:

   Email,cmc_sourcemethod,cmc_sourcemethodname
4. Locate the **contact-Field-_cmc_sourcecategoryid_value** field and view the fields on the **General** tab.
5. In the **Parameters** field, add the source category name that is used for integrating Contact data.

   *Example*: If the external source category name is *Application*, update the value in the **Parameters** fields as follows:

   Application,cmc_sourcecategory,cmc_sourcecategoryname

Update the Data Mapping for the EthnicitiesList Field

1. In Anthology Reach, navigate to **Settings** > **Integrations** > **Integration Mappings**.
2. Choose the Mapping Type as Data mapping to filter the records and locate **contact-Option Set-Race-ethnicitiesList** and view the fields.
3. Ensure that the value of the **Dataset Name** field is *RaceType.*

**Configurations for the Course Section Entity**

Update the Field Mapping for the Staff Id Field

1. In Anthology Reach, navigate to **Settings** > **Integrations** > **Integration Mappings**.
2. Locate the **mshied_coursesection-Field-cmc_staffid** field and view the fields on the **General** tab.
3. In the **Parameters** field, add the external identifier of the user that is configured in the **Integration Default User** field on the **Default Configuration**page. For details see, theDefault Configurationstopic inAnthology Reach help.

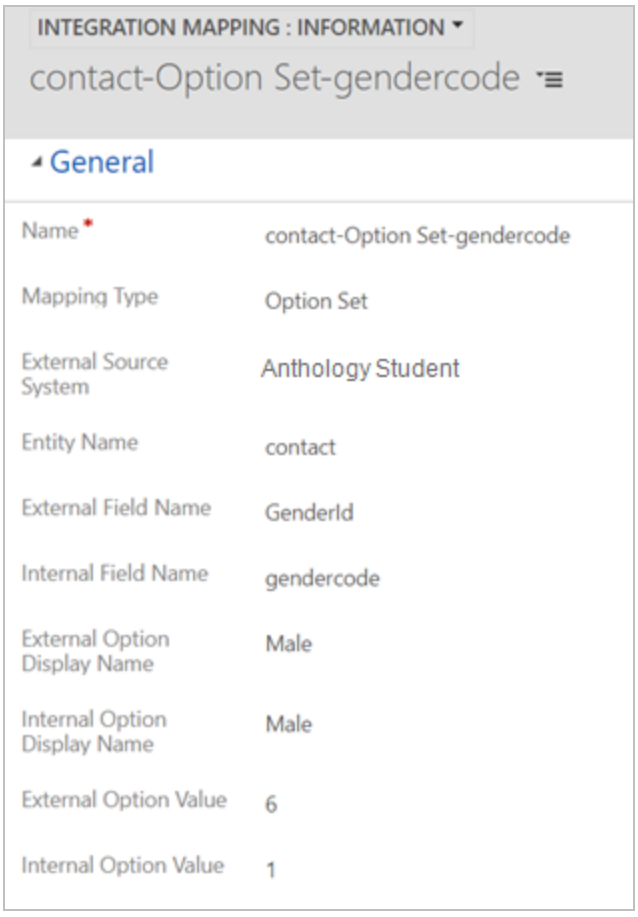
   *Example*: If the external identifier for the user is *2*, update the value in the **Parameters** field as follows:

   {InstructorId},,systemuser,cmc_externalidentifier,2

**Picklist Mappings**

The picklist or Option Sets need to be mapped when the record is integrated in the flow. To enable this functionality, Option Set mappings are maintained in an MS Dynamics entity.

- **Name**: Specify the mapping name.
- **Mapping Type**: Select **OptionSet**.
- **External Source System**: Select the value Anthology Student.
- **Entity Name**: Schema name of the entity in Anthology Reach for which the mapping is required, for example, contact.
- **External Field Name**: Field name as appearing in the request sent to the flow, for example, GenderId.
- **Internal Field Name**: Field schema name in Anthology Reach, for example, gendercode.
- **External Option Display Name**: Display name of the option in Anthology Student, for example, Male.
- **Internal Option Display Name**: Display name of the option in Anthology Reach, for example, Male.
- **External Option Value**: Value of the option in Anthology Student, for example, 6.
- **Internal Option Value**: Value of the option in Anthology Reach, for example, 1.




## Add a New Picklist Value

The following steps need to be performed when a new picklist value is added inAnthology Student:

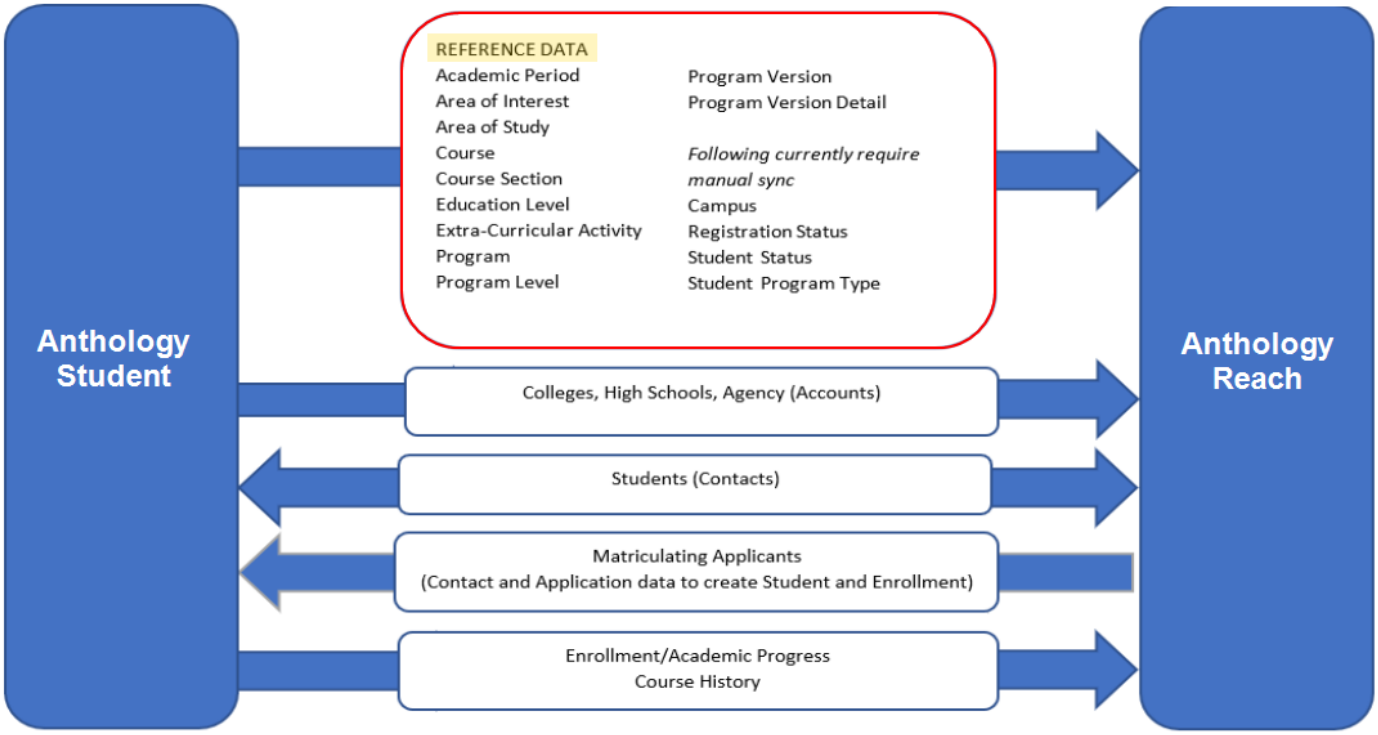
1. Create the corresponding picklist value inAnthology Reachby editing the field in an appropriate solution. Note the picklist numerical value generated for this value.
2. Add an Integration Mapping record with appropriate values for the new mapping based on the value retrieved in the earlier steps.

## Integrated Entities

Records of entities integrated between Anthology Student and Anthology Reach are categorized as operational data and reference data. Reference data is used to qualify operational data.

### Reference Data

Anthology Student is the system of record ("source of truth") for reference data items. Records of the reference data entities are created in Anthology Reach as a result of unidirectional synchronization from Anthology Student.




Examples of reference data entities include:

- Academic Period
- Area of Interest
- Area of Study
- Course
- Course Section
- Education Level
- Extra-curricular Activity
- Program
- Program Level

- Program Version
- Program Version Details
- Registration Status
- Shifts
- Student Status

**Notes:**

- Records of the following entities must be imported manually into Anthology Reach:
  - Campus
  - School Status
  - User
  - Registration Status
  - Student Program Type
  - Grade Level
- Option set mapping must be performed manually for the following fields:

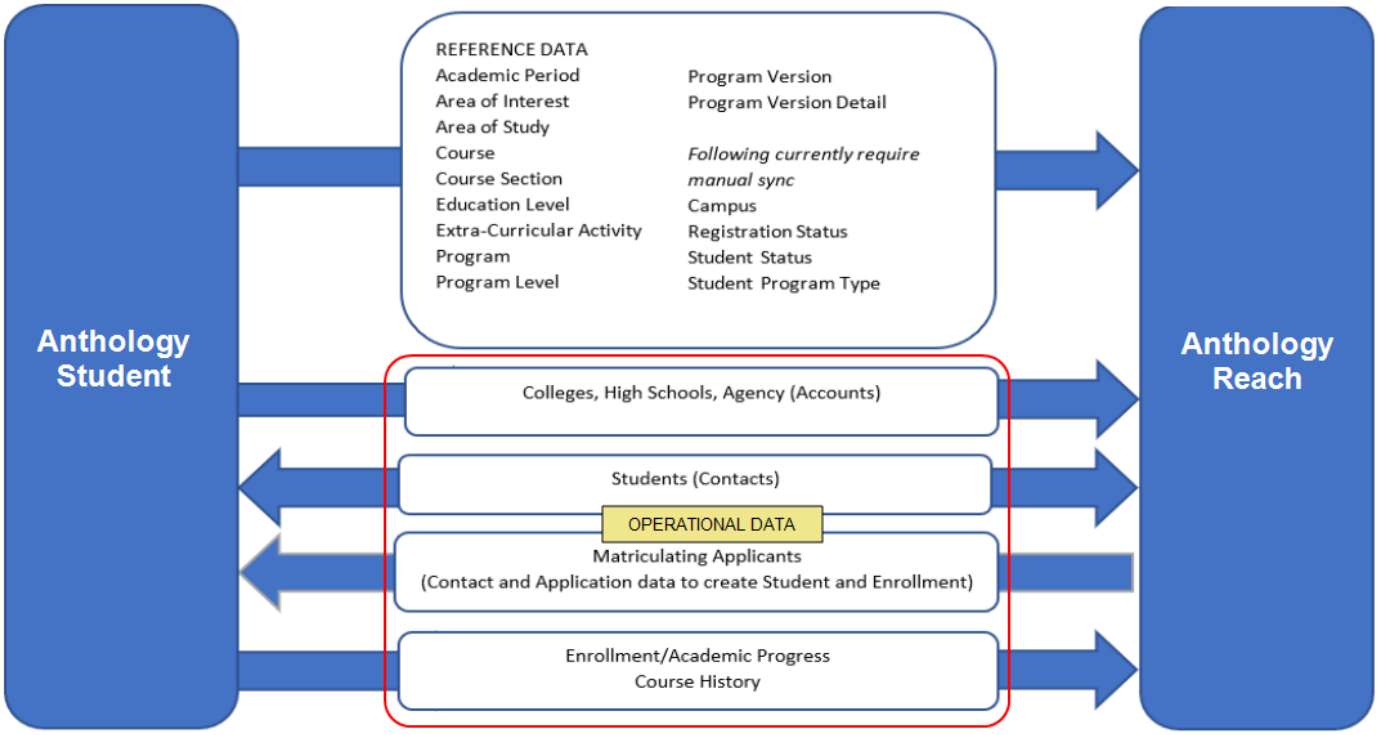
| Entity | Fields |
|---|---|
| Account | • Country<br>• State |
| Contact | • Last Permanent Residence<br>• Country<br>• Nationality<br>• Race<br>• County<br>• State<br>• Country |
| Address | • Address Type<br>• Country<br>• State<br>• County |
| Academic Period Details | • Attendance Type |
| Address | • Mail Type |

**Operational Data**

Operational data or transactional data is maintained in both Anthology Reach and Anthology Student.

Depending on the entities, operational data can be integrated:

- Uni-directionally from Anthology Student to Anthology Reach
- Uni-directionally from Anthology Reach to Anthology Student
- Bi-directionally between Anthology Reach and Anthology Student




Examples of operational data entities include:

- Account
- Address
- Contact
- Enrollment
- Previous Education
- Student Group

Records of the following operational entities are created in Anthology Reach as a result of uni-directional integration from Anthology Student.

- Academic Progress
- Account (High School, College, Agency Branch)
- Address
- Document status
- Hold Groups
- Previous Education
- Student
- Student Courses
- Student Group Memberships
- Student Status History

Special handling applies to several entities:

- **Address** - When an application record is synced from Anthology Reach to Anthology Student, address records associated with the contact in the application record are synced from Anthology Reach to Anthology Student. When address records, that were synced as a result of application sync, are updated in Anthology Reach, the address updates are integrated from Anthology Reach to Anthology Student.
- **Application** - On syncing from Anthology Reach to Anthology Student, equivalent Enrollment records are created in Anthology Student.
- **Contact** - This entity is synced bi-directionally.
- **Enrollment** - Integration occurs from Anthology Student to Anthology Reach. Further updates to instances of the Enrollment entity occur only in Anthology Student. Such updates are then synced with equivalent enrollment records in Anthology Reach.

  **Note**: Enrollment records in Anthology Reach are displayed in read-only mode.
- **Previous Education** - This entity is updated bi-directionally.
- **Test Scores** are updated via triggers (instead of flows) from Anthology Reach to Anthology Student.

### Flows and Procedures

The following flows integrate data from Anthology Reach entities with Anthology Student entities.

| Anthology Reach Flows | Description |
|---|---|
| CNE-CNS Application-Enrollment Integration (Engage) | Creates equivalent Enrollment records in Anthology Student from records of the Application entity in Anthology Reach.<br><br>The CMC Application Integration Trigger updates the Integration Flag field in the Application entity. Updates to this field trigger the CNE-CNS Application-Enrollment Integration (Engage) flow.<br><br>Institutions can modify conditions in the flow to control the integration of applications from Anthology Reach to Anthology Student. |
| CNE-CNS Contact-Student Integration (Engage) | Updates records of the Student entity in Anthology Student from records of the Contact entity in Anthology Reach. |
| CNE-CNS Previous Education - Student Previous Education Integration - Create/Update (Engage) | Creates or updates equivalent Previous Education records in Anthology Student. |

**Note**: Shipped Anthology Reach flows are maintained within a Microsoft Dynamics 365 solution. Before modifying a flow, we recommend that you disable the default flow, and then create a new flow for the entity. System upgrades will overwrite flows, which would cause changes to be lost if default flows are modified.

For additional information about integration flows and procedures details, refer to the topics listed below.

#### Add Fields to Field Mapping Templates

You can configure new fields for data integration between Anthology Reach (CNE) and Anthology Student (CNS).

Refer to the following procedures to add fields to integrate data:

- From Anthology Student to Anthology Reach
- From Anthology Reach to Anthology Student

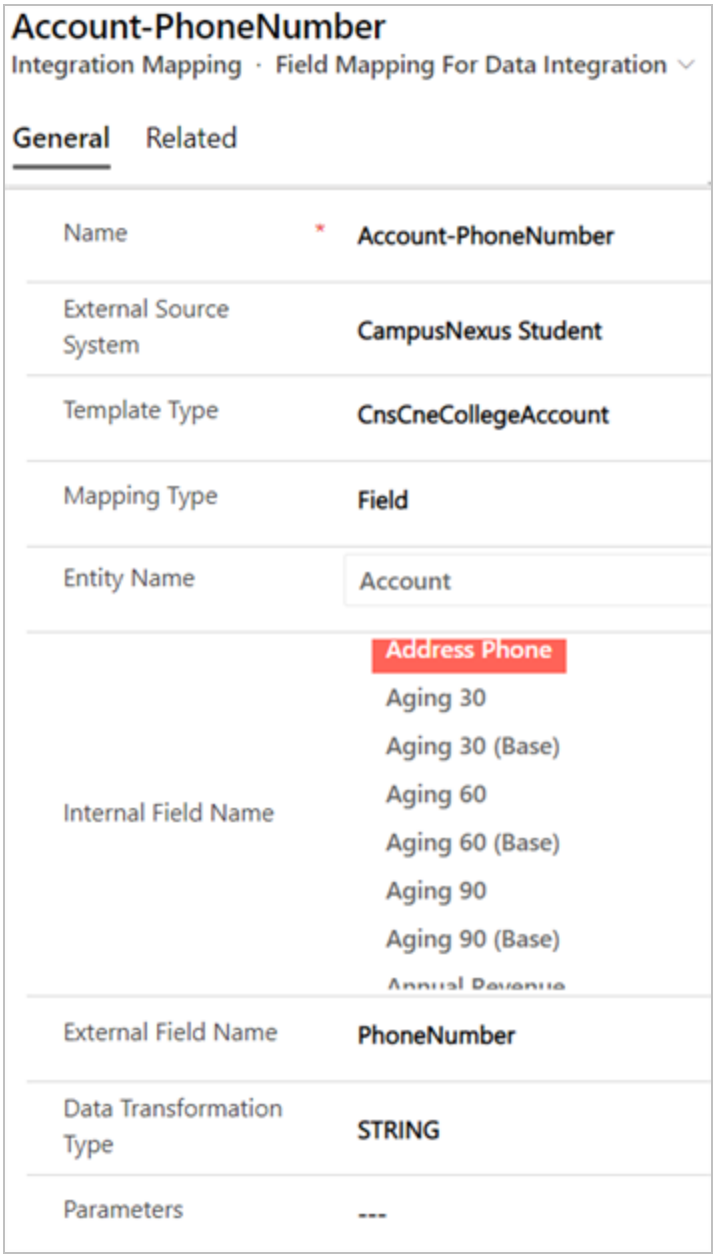
## From Anthology Student to Anthology Reach

Data integration from Anthology Student to Anthology Reach uses field mapping templates that have the following naming convention:

- CnsCne{Student Entity Name}{Anthology Reach Entity Name}

  *Example*: CnsCneCollegeAccount

The Field Mapping Templates are used to determine what data will be integrated when integrating an Anthology Student entity. The following image shows a Field Mapping Template:




The following table provides information on the fields in the Field Mapping Template.

| Field Name | Description |
|---|---|
| Name | The value can be anything, It is suggested to have the {Anthology Reach Entity Name} and the {Anthology Reach Field Name}. |
| External Source System | Set this to Anthology Student. |
| TemplateType | Use this format CnsCne{Anthology StudentEntity Name}{Anthology Reach Entity Name}, for example, CnsCneCollegeAccount. |

| Field Name | Description |
|---|---|
| Mapping Type | Field |
| Entity Name | Set the Anthology Reach entity name to which the field must be integrated. |
| Internal Field Name | Set the Anthology Reachfield name which must be integrated. |
| External Field Name | Set the CNS Field Name as it appears in the CNS entity payload. |
| Data Transformation Type | Select a data transformation type the field. It can have one of the following values:<br>• STRING<br>• NUMBER<br>• DATETIME<br>• LOOKUP<br>• OPTIONSET<br>• MULTIOPTIONSET<br>For more details, see Data Transformation Parameters. |
| Parameters | Specify the parameters to be used for the Data Transformation. Details are provided in the Data Transformations table. |

## Data Transformation Parameters

In the field mapping record, based on the value selected in the **Data Transformation Type**, configure the **Parameters** field.

In the **Parameters** field:

- The parameters are separated by a comma.
- The first parameter is the External Field Name which is enclosed in brackets {} (e.g., {PhoneNumber}). Alternatively, a static value can be specified, without the brackets.

The following table provides information on the parameters that can be configured for each **Data Transformation Type**.

| Data Transformation Type | Format | Description | | |
|---|---|---|---|---|
| | | Parameter Name | Description | Required |
| STRING | <External Field Name>,<Prefix> | External Field Name | Specify the external field name or a static value | Required |
| | | Prefix | This value will be used to prefix the value to the External Field Value | Optional |
| NUMBER | <External Field Name> | External Field Name | Specify the external field name or a static value | Required |
| DATETIME | <External Field Name>,<Date Format>,<Time Zone>,<Time Off-set> | External Field Name | Specify the external field name or a static value | Required |
| | | Date Format | The date format in which the external value is represented. E.g. yyyy-MM-dd | Optional |
| | | Time Zone | The Time Zone of the external date value. If not specifieed, it is assumed to be in UTC | Optional |
| | | Time Offset | If the time is offset by a fixed number of hours, the value specifies the offset to the UTC time | Optional |

| Data Transformation Type | Format | Description | | |
|---|---|---|---|---|
| | | Parameter Name | Description | Required |
| LOOKUP | {External Field Name},<Dataset>, <Schema name of the entity>, <Schema name of the field based on which association with the parent entity occurs>, <Optional default value> | External Field Name | Specify the external field name or a static value. | Required |
| | | Dataset | Name of the Dataset which has the mappings between the external value and the internal value. | Optional |
| | | Schema name of the entity | Schema name of the entity | Required |
| | | Schema name of the field based on which association with the parent entity occurs | Schema name of the field based on which association with the parent entity occurs. | Required |
| | | Optional default value | This is a fallback value, when the external value cannot be mapped to an internal value. | Optional |
| OPTIONSET | {External Field Name},<Dataset> | External Field Name | Specify the external field name or a static value. | Required |
| | | Dataset | Name of the Dataset which has the mappings between the external value and the internal value. | Optional |
| MULTIOPTIONSET | {External Field Name},<Dataset> | External Field Name | Specify the external field name or a static value. | Required |
| | | Dataset | Name of the Dataset which has the mappings between the external value and the internal value. | Optional |

## From Anthology Reach to Anthology Student

Data integration from Anthology Reach to Anthology Student uses field mapping templates that have the following naming convention:

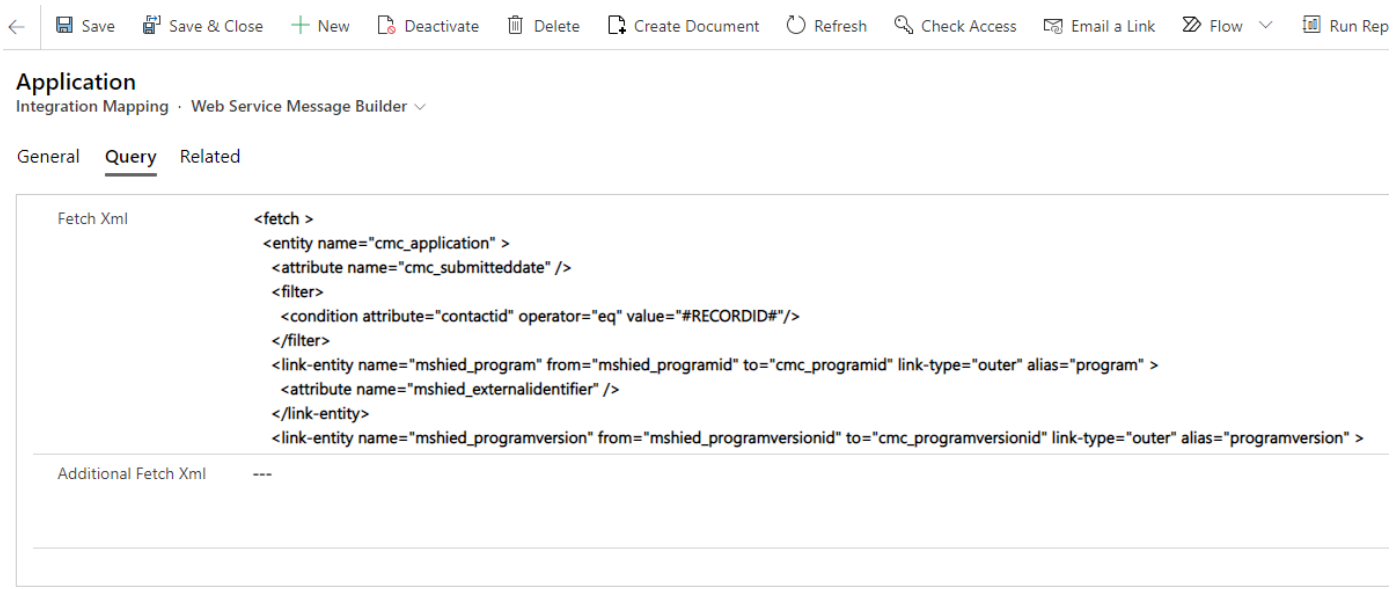
CneCns{Anthology Reach Entity Name}{Student Entity Name}

*Example*: CneCnsContactStudent

The integration field mapping templates in the Web Service Message Builder form define the fields and the JSON Payload structure that is used for data integration between Anthology Reach and Anthology Student. The integration mapping records have a parent child relationships.

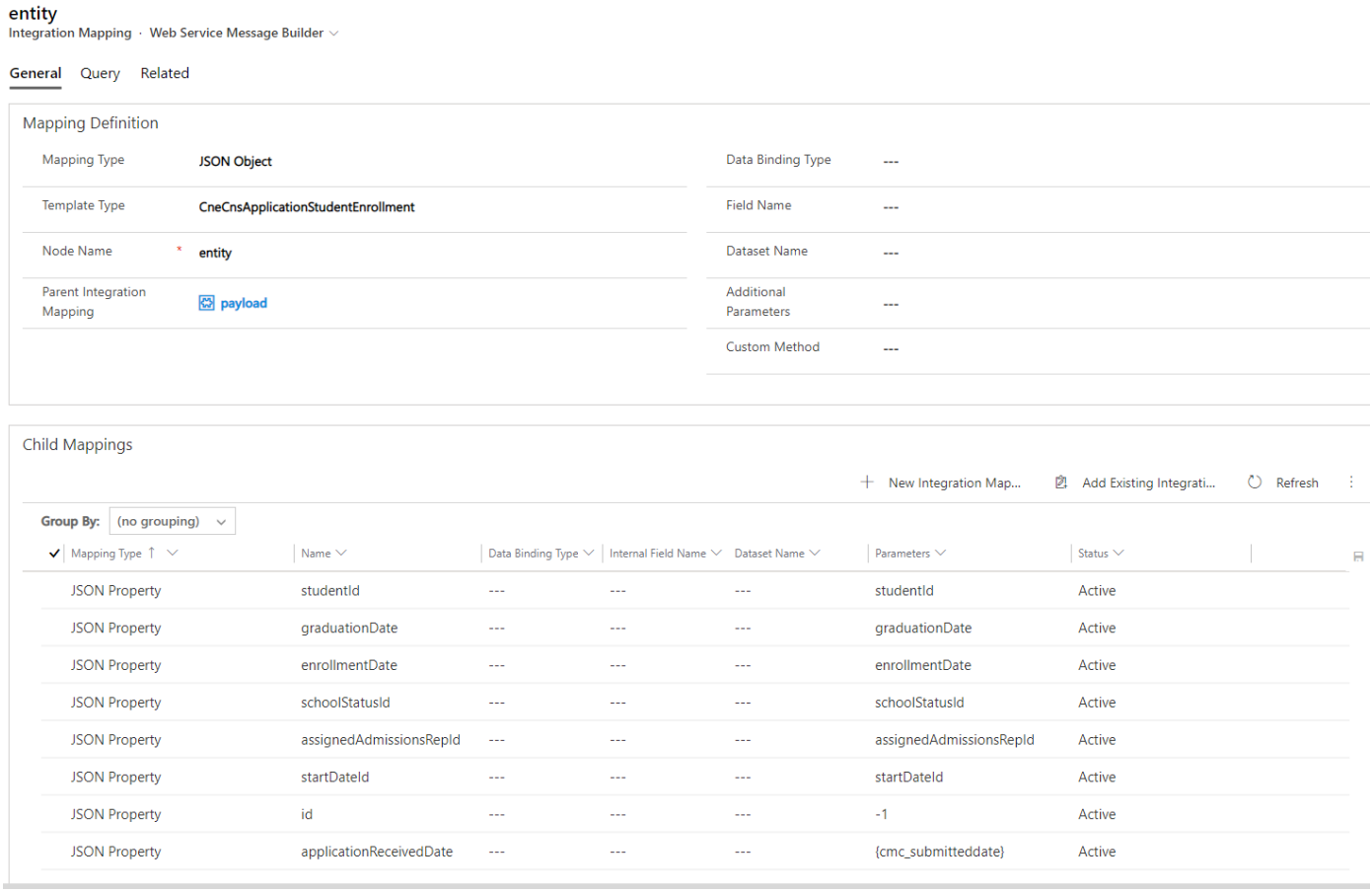
In the root integration mapping record, the **Parent Integration Mapping** field is blank.

In the Integration mapping record, the data for the JSON payload is defined by the FetchXML Query attribute. The **FetchXML Query** attribute is present in the **Query** tab in the **Web Service Message Builder** form. The query has one parameter which is a placeholder for the Record ID against which the data needs to be fetched. The placeholder is represented using the key #RECORDID#.




The JSON members/property values can be nested. To depict this nested payload data, the Parent-Child relationship of Integration Mappings is leveraged.

Following is a sample screenshot of the integration mapping, where the **Child Mappings** grid shows the child Integration Mapping records.




In this screenshot, we have the “entity” JSON property having child properties as listed in the “Child Mappings” grid. The “entity” field is a child element of the parent integration mapping which is defined by the field “Parent Integration Mapping” (“payload” in the screenshot)

Add a new field for data integration

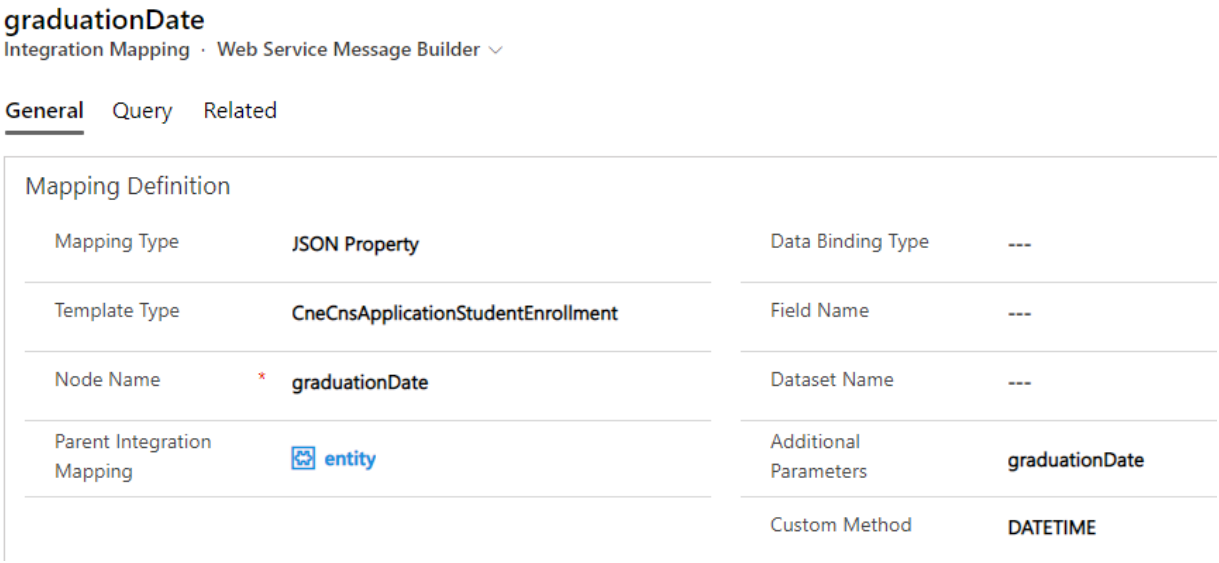
In the **Web Service Message Builder** form, perform the following steps.

1. In the root Integration Mapping record, update the Fetch XML query to fetch the required field from Anthology Reach.
2. In the parent Integration Mapping record, in the Child Mappings section, add a new Integration Mapping record.
3. In the new Integration Mapping form, specify the field values as follows:
    a. **Mapping Type** - Select *JSON Property*.
    b. **Template Type** - Specify the Template name.
    c. **Node Name** - Specify the name of the JSON property.

d. **Parent Integration Mapping** - Is automatically set to the parent Integration Mapping record.

e. **Custom Method** - Specify the data transformation type. For more information, see Data Transformation Parameters.

f. **Additional Parameters** - Specify the transformation parameters. For more information, see Data Transformation Parameters.




## Data Transformation Parameters

In the field mapping record, based on the data transformation type specified in the Custom Method field, the Additional Parameters field must be configured.

This section provides information on the values that can be configured in the Custom Method and Additional Parameters fields of a field mapping record.

The following are the data transformation types that can be specified in the **Custom Method** field:

- STRING
- DATETIME
- NUMBER
- BOOLEAN

The following three parameters are common for the**Custom Method** type:

- **Source Parameter**: This represents the output column of the FetchXML query from the data that is fetched. The alias or the column name needs to be enclosed in {} brackets. For example, {fullname}.

  Alternatively, a static value can be specified for this parameter. When a static value is specified it should not be enclosed within {} brackets. For the DATETIME data transformation type, either a fixed date can be specified or CURRENTDATE can be set to indicate the current date and time.

- **Fallback Value**: This is an optional parameter to indicate a fallback value when there is no value set for the field in Anthology Reach.
- **Dataset Name**: This is an optional parameter when the value in Anthology Reach needs to be mapped to another value. The dataset name is specified from where the data mapping needs to be referenced.

  **Note**: To support the earlier form of data mapping where the data mapping records were located based on the entity name and field name, use the format – "FIELDDATASET-{entityname}-{fieldname}". For example, FIELDDATASET-contact-gendercode.

The STRING data transformation type has the following additional parameters depending on the type of operation:

- **SPLIT - To split the field values based on a delimiter, use the following parameters:**
  - **operationtype**- Specify**SPLIT**to perform the split operation.
  - **position**- The position in the field value from where the content will be split.
  - **splitdelimiter**- The delimiter (a string or regular expression) for splitting the field value.

    **Note**:
    - All character based delimiters can be used for performing the split operation.
    - When the delimiter is:
      - comma - use*$comma*.
      - space - use " ".
- **RIGHT or LEFT - To extract the field values from right or left, use the following parameters:**
  - **operationtype**- Specify*RIGHT* or*LEFT*to extract text from right or left. It can have one of the following values:
    - *RIGHT*- Extracts the field value from right.
    - *LEFT*- Extracts the field value from left.
  - **picklength**- The number of characters that must be extracted from right or left and displayed as the field value.

The STRING data transformation type has the following additional parameters depending on the type of operation:

- **SPLIT - To split the field values based on a delimiter, use the following parameters:**
  - **operationtype**- Specify**SPLIT**to perform the split operation.
  - **position**- The position in the field value from where the content will be split.
  - **splitdelimiter**- The delimiter (a string or regular expression) for splitting the field value.

**Note:**

- All character based delimiters can be used for performing the split operation.
- When the delimiter is:
  - comma - use*$comma*.
  - space - use " ".

- **RIGHT or LEFT - To extract the field values from right or left, use the following parameters:**
  - **operationtype**- Specify*RIGHT* or*LEFT*to extract text from right or left. It can have one of the following values:
    - *RIGHT*- Extracts the field value from right.
    - *LEFT*- Extracts the field value from left.
  - **picklength**- The number of characters that must be extracted from right or left and displayed as the field value.

The NUMBER data transformation type has the following fourth parameter:

- **Rounding Digits**: By default, the numbers are rounded to 0 decimal places. However, if the number should be rounded to a desired decimal place, specify the decimal number in this paramete. For example, if the input value is 12.3416 and the Rounding Digits is 2, then field value is updated as 12.34.

The DATETIME data transformation type has the following fourth parameter:

- **DateFormat**: The default date format is “yyyy-MM-ddTHH:mm:ss”. If a different date format is required it can be specified in this parameter. For information on the date format, see Microsoft documentation.

The following table provides sample values for the **Additional parameters** field for each data transformation type in the **Custom Method** field:

| Data Transformation Type | Syntax | Parameters | Description |
|---|---|---|---|
| STRING | {internalfieldname} | {fullname} | Takes the value from the FetchXML query from the column name fullname. |
| STRING | {internalfieldname}, fallbackvalue, datasetname, operationtype, position, splitdelimiter | {mshied_race_},,RaceMap, Split,1,$comma | |

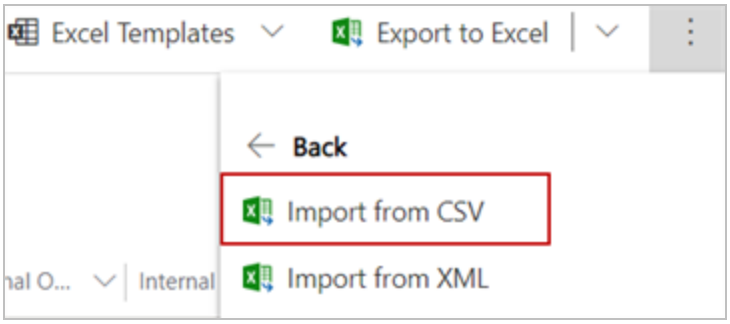
| Data Transformation Type | Syntax | Parameters | Description |
|---|---|---|---|
| STRING | {internalfieldname}, fallbackvalue, datasetname, operationtype, picklength | {nickname},fallbackvalue, nickname,LEFT,4 | |
| STRING | {internalfieldname}, fallbackvalue, datasetname, operationtype, picklength | {nickname},fallbackvalue, nickname,RIGHT,4 | |
| DATETIME | | {birthdate},,,yyyy-MM-dd | Takes the value from birthdate column with no values for the Fallback value and dataset name are not specified. The date format is specified to output the value as “1977-11-14”. |
| NUMBER | | {gender},-1, FIELDDATASET-contact-gendercode | Takes the value from **gender** column, with a Fallback Value as -1. The Dataset parameter is specified to support the old format using the entity name “contact” and field name as “gendercode”. |
| NUMBER | | {color},2,Colormap | Takes the value from the **color** column, with a Fallback value as 2, and the value of the Dataset parameter is Colormap, that is used to map the value in Anthology Reach to the required value. |
| NUMBER | | -1 | This sends a static number value of -1. |
| BOOLEAN | | {ferpa},false | Takes the value from column “ferpa” Fallback value is false. |
| DATETIME | | CURRENTDATE,,yyyy/MM/dd hh:mm:ss | Value is picked from the current date time of integration. No fallback value is set. Date format is provided to give output as 2021/02/15 04:44:11. |

### Import Integration Mapping Templates

The following table provides the excel files that can be downloaded and used to create and update the Student to Reach integration mappings in the Reach environment as needed.

| CSV File Name | Purpose |
|---|---|
| AnthologyStudent-Reach-IntegrationMappings.csv<br>[Last updated: May 11, 2023] | Use this file for new implementation of Anthology Reach. This file contains the full set of all latest integration mappings. |
| AnthologyStudent-Reach-IntegrationMappingsChangeLog.xls<br>[Last updated: May 11, 2023] | Use this file to review and update the existing Student integration mappings in Anthology Reach. This file identifies new or changed mappings. |

1. In the Anthology Reach environment, navigate to **Settings** > **Integration** > **Integration Mappings**.
2. Click the **Import from CSV** option.




3. On the **Import from CSV** page, click **Choose File** to select the required file, and click **Next**.
4. Click **Review Mapping** and verify that fields are mapped.
5. For the Option Set fields, select the Option Set fields and values before importing, as these fields will not be mapped.
6. Click **Finish Import**.

### Integrate External Document Status Records

The following flows are enabled to integrate Document records from Anthology Student to records of the External Document Status entity in Anthology Reach:

- **CNS-CNE Documents - External Document Status_Pull Based (Engage)** – By default, this flow is set to run every 15 minutes, and "pulls" Document records in batches of 100.
- **CNS-CNE Documents - External Document Status_Create/Update (Engage)** – This flow is called from the above flow and creates and updates External Document Status records in Anthology Reach.

- **CNS-CNE Documents - External Document Status_Delete (Engage)** – When a Document record is deleted in Anthology Student, this flow is triggered to also delete the equivalent External Document Status record in Anthology Reach.

In Anthology Reach, External Document Status records are:

- Created when Document records are created in Anthology Student
- Updated when corresponding Document records are updated in Anthology Student. For instance, the status of the records is updated from **Required** to **Requested - Not Received**, **Requested - Required**, **Approved By Advisor**, and so on.

Updates to External Document Status records will take place after Document records are initially imported manually or using an integration solution. This enables syncing to occur later in Anthology Reach as Document records are created or updated in Anthology Student.

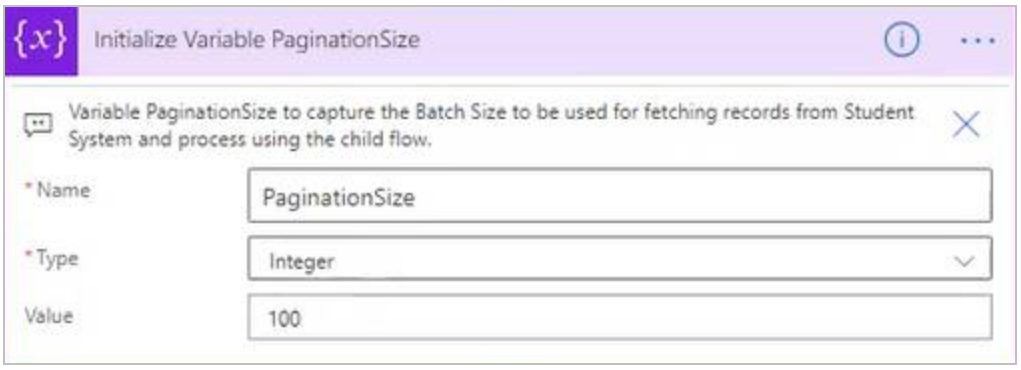
# Create or Update External Document Status Records

1. Import Document records from Anthology Student to Anthology Reach.
2. On the Default Configuration page, set a value in the **External Document SIS Last Integration Time** field.

Updates to External Document Status records will occur at 15-minute intervals in Anthology Reach.

**Notes:**

- Mappings of fields of the Document entity in Anthology Student to the External Document Status record in Anthology Reach are available in integration mapping records. Administrators can add records or edit the current records to manage the integration in their scenario.
- In the **CNS-CNE Documents - External Document Status_Pull Based (Engage)** flow, records will be "pulled" in batches of 100 until the record count in the source system is less than 10, after which records will not be "pulled" in the current cycle. "Pulling" will resume on the next run of the flow after 15 minutes to integrate the remaining records and any new records created during the interval.
- The default batch size of 100 records can be changed in the following step in the **CNS-CNE Documents - External Document Status_Pull Based (Engage)** flow:



- Integration errors will be logged in the **Additional Details** field in the integration log record that's created for every run of the **CNS-CNE Documents - External Document Status_Pull Based (Engage)** flow.
- Integrated External Student Status records will be displayed in reverse chronological order (default) in:
    - The contact (Student Progress > Enrollments > External Document Status tab).
    - The Enrollments grid (Recruitment > Enrollments > External Document Status tab).

**Integrate Groups**

The following flows integrate changes in group memberships from Anthology Student to Anthology Reach:

- **CNS-CNE Student GroupMembership - GroupMembership Integration_Pull Based (Engage)**

  This flow "pulls" information from changed groups at an hourly frequency. By default, the concurrency of this flow is set to 1, indicating that only a single instance of this flow must run. This will prevent scenarios where the same record is processed by a second instance of the flow while it's being processed in the first instance.
- **CNS-CNE Student GroupMembership - GroupMembership Integration_Create/Update (Engage)**

  To create and / or update group records in Anthology Reach, information is passed into this flow:
    - From the above flow in a "pull" scenario.
    - From the next flow in a trigger-based update of manual groups.
- **CNS-CNE Student GroupMembership - GroupMembership Integration_Trigger Based (Engage)**

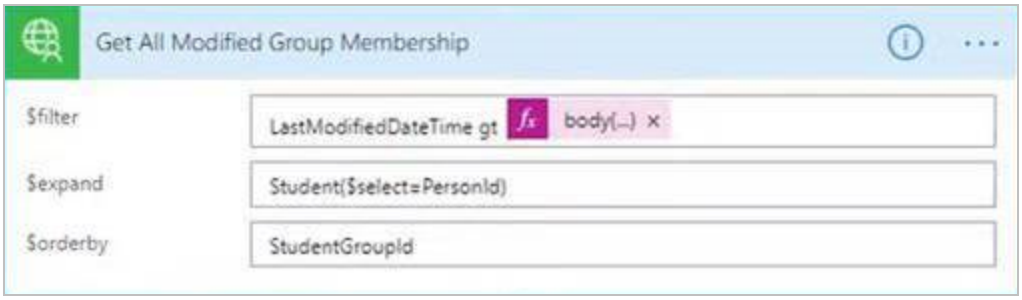
# Update Groups in Anthology Reach

1. Import group membership information from Anthology Student to Anthology Reach.
2. On the Default Configuration page, set a value in the field **Group Membership SIS Last Integration Time**.
3. Enable the following flows:
    - **CNS-CNE Student GroupMembership - GroupMembership Integration_Pull Based (Engage)**
    - **CNS-CNE Student GroupMembership - GroupMembership Integration_Create/Update (Engage)**

Updates to group memberships (dynamic, static and manual) will occur automatically at hourly intervals in Anthology Reach.

**Notes**:

- By default filter criteria are set in the **CNS-CNE Student GroupMembership - GroupMembership Integration_Pull Based (Engage)** flow to update all groups (Dynamic, Manual and Static) from Anthology Student to Anthology Reach. Administrators can update criteria in their implementation of this flow.




For example, criteria can be set to integrate dynamic group updates only.

- To perform a one-time integration of manual groups from Anthology Student to Anthology Reach:
  1. Publish the following workflows:
     - IStudentGroupMemberService.AddStudentsToStudentGroupEvent
     - IStudentGroupMemberService.AddStudentToGroupEvent
     - IStudentGroupMemberService.RemoveStudentFromGroupEvent
     - IStudentGroupMemberService.RemoveStudentsFromStudentGroupEvent
  2. Enable the flow **CNS-CNE Student GroupMembership - GroupMembership Integration_Trigger Based (Engage)**.

**Integrate Hold Codes and Hold Assignments**

Integration of Hold Code and Hold Assignment records involves prerequisites and manual steps.

# Prerequisites

1. The following fields must be mapped in Integration Mapping records:

| Anthology Student | Anthology Reach |
|---|---|
| HoldCode | cmc_externalidentifier cmc_code |
| Action | cmc_holdimpact<br>**Note**: This is an Option Set field. Administrators must add values to this field before integrating hold code assignment records. |
| Module | cmc_holdmodule<br>**Note**: This is an Option Set field, values are available for this field in Anthology Reach. |
| Description | cmc_name |

2. Hold Code records identical to those in Anthology Student must be created in Anthology Reach. This is important as inheritance from Anthology Student will be successful only if Hold Code records with identical values are available in both systems.

# Hold Code Integration in Anthology Reach

The following flow in Anthology Reach is mapped to workflows in Anthology Student:

| Anthology Reach Flow | Anthology Student Workflows |
|---|---|
| CNS-CNE Hold Student Group - Hold Integration (Engage)<br>This flow tracks the addition or removal of groups in a hold code. | • CNS Integration Hold Student Group Saved<br>• CNS Integration Hold Student Group Deleted |

The workflows in Anthology Student will monitor changes to hold code records which will be transmitted to Anthology Reach.

In Anthology Reach, hold code integration, i.e., hold assignment record creation will occur after group and group membership information of the contact is integrated from Anthology Student. Hold assignment records can also be created in Anthology Reach in a scenario where the Anthology Student user adds student records to a group that is already associated with a hold code.

# Hold Assignment Record Mapping

In Anthology Reach, hold assignment records are integrated indirectly from Anthology Student via related Group – Group Membership and Group – Hold Codes records.

| Anthology Reach Hold Assignment Record | Anthology Reach Group Membership Record |
|---|---|
| cmc_contactid | GroupMembership.Contact |
| cmc_enddate | GroupMembership.EndDate |
| cmc_externalsourcesystem | GroupMembership.ExternalSourceSystem |
| cmc_hold | lookup to Hold with Group Membership.Group |
| cmc_startedate | GroupMembership.StartDate |
| cmc_name | Auto Name Convention “ContactName-HoldName” |

**Note**:

The CNS-CNE StudentHoldCode - HoldAssignment Integration_Main (Engage) flow is called from the flows:

- CNS-CNE StudentHoldCode - HoldAssignment Integration_Group Membership Trigger (Engage)
- CNS-CNE Hold Student Group - Hold Integration (Engage)

It creates hold assignment records in Anthology Reach based on the association of groups and Hold Codes in Anthology Student.

For instance, in Anthology Student if the student belongs to multiple groups associated with an individual hold code, identical mappings will be created in Anthology Reach in which the student will belong to multiple groups

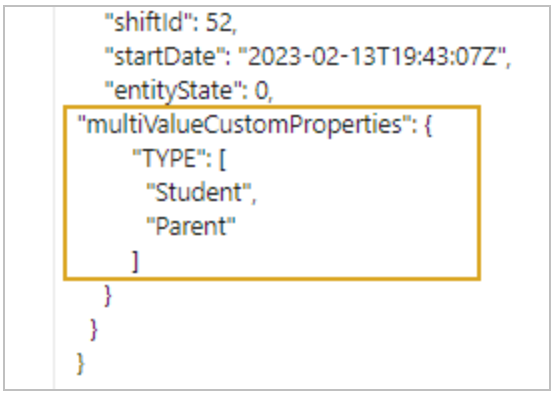
which are associated with a single hold record which will include a related Hold Assignment for the student. This Hold Assignment record will be related to multiple Group Membership records under different groups.

### Integrate Multi-Select List Fields and Option Sets

This section provides information on how to integrate data from Anthology Reach to Anthology Student for a multi-select list field.

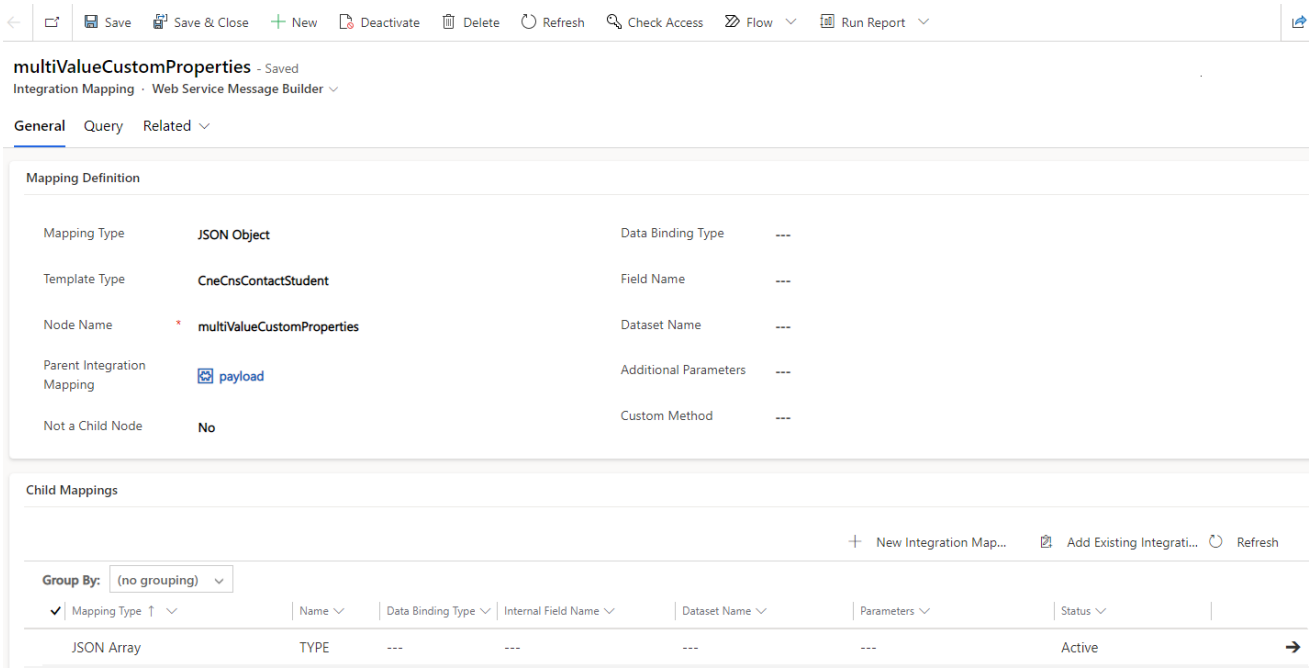
## From Anthology Reach to Anthology Student

The following is an illustration of the JSON payload for a multi-select list field to be sent from Anthology Reach to Anthology Student.

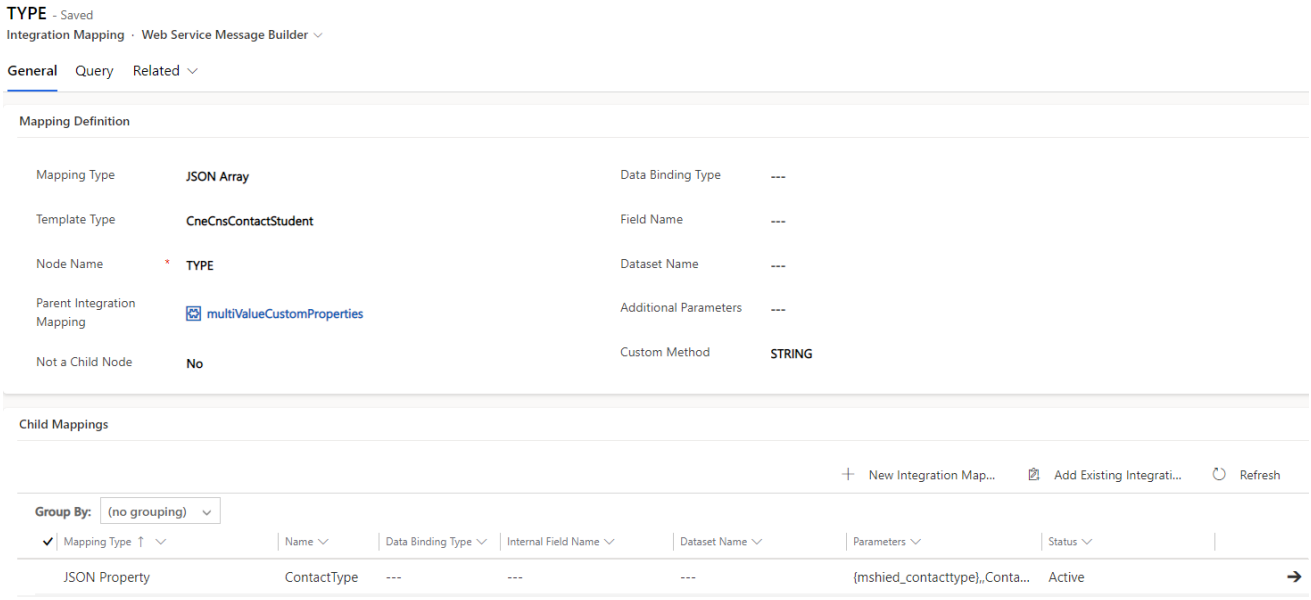



Before sending the data of a multi-select list field from Anthology Reach to Anthology Student, the multi-select list field must be added to the JSON data structure in the Field Mapping Template. To do so:

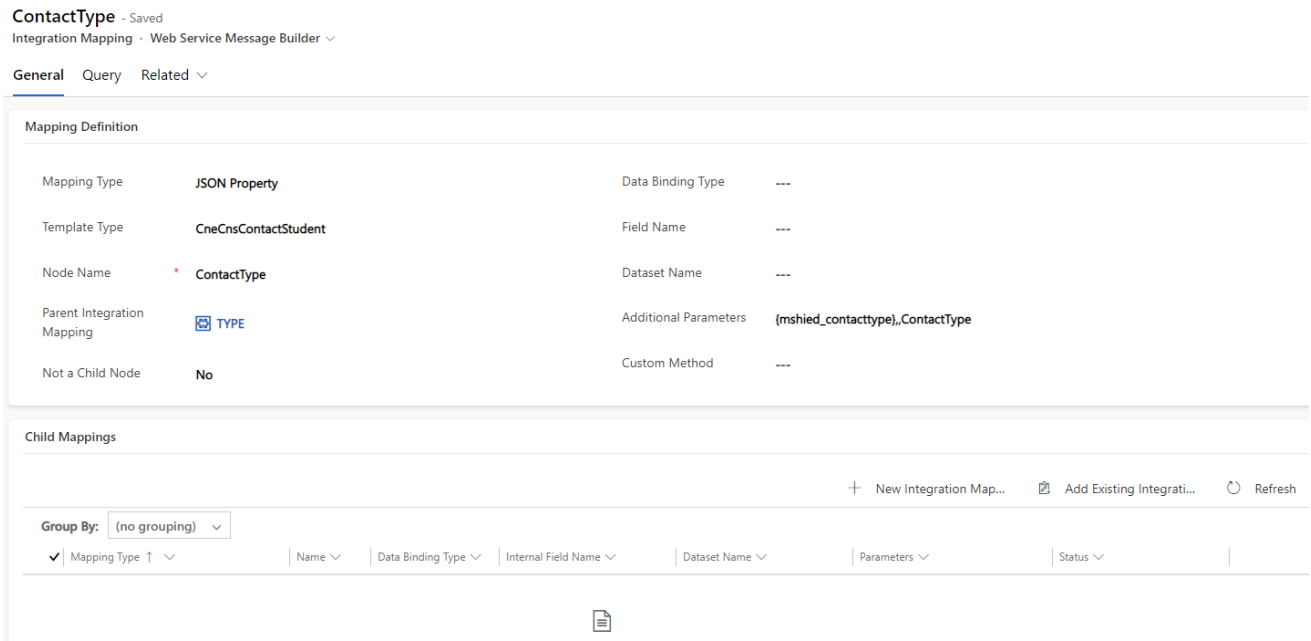
1. In the Integration Mapping Template of the entity, add an integration mapping record with the **Mapping Type** as *JSON Object*.



2. For the payload Integration Mapping record of the entity to which the Multi-Select List Item field must be added, create a child integration mapping record, **MultiValueCustomProperties**, and set the value of the **Mapping Type** as *JSON Array*.




3. In the **MultiValueCustomProperties** Integration Mapping record, for each Multi-Select List Item field create a child mapping record with the **Mapping Type** as *JSON Property* to store the details of the multi-select list property.



# Integrate Multi-Select Option Set Fields

To enable appending or overwriting values of multi-select fields (e.g., Race, Ethnicity) when data is sent from Anthology Student to Anthology Reach, create an integration mapping record with the following field values.

| Integration Mapping Field | Value |
|---|---|
| Name | Any descriptive name (e.g. Contact-ResetMultiOption). |
| Template Type | The template name for which the record must be updated. |
| Mapping Type | Field |
| Internal Field Name | ResetMultiSelect_Field |
| Data Transformation Type | STRING |
| Parameter | Concatenated schema names of the multi optionset fields.<br>*Example*: "mshied_race_#cmcps_custommultioption"<br>where - **mshied_race_** and **cmcps_custommultioption** are schema names of multi optionset fields which are concatenated using the separator #. |

### Integrate Student Status History Records

The following flows enable the integration of student status history records from Anthology Student to Anthology Reach:

- **CNS-CNE Student School Status History – Student Status History_Pull Based (Engage)** – By default, this flow is set to run every 15 minutes, and "pulls" student status history records in batches of 100.
- **CNS-CNE Student School Status History – Student Status History_Create (Engage)** – This flow is called from the above flow and creates student status history records in Anthology Reach.

Student status history records log changes to the value of the **Student Status** field of an enrollment. For example, its value can change in the following sequence: **Inquiry**, **Cleared to Register**, **Application Complete**, **Currently Attending**, and so on.

# Integrate Student Status History Records

1. Import enrollment and student status history records from Anthology Student to Anthology Reach.

   The following steps describe how records created later in Anthology Student will be automatically integrated into Anthology Reach.

2. From the **StudentSchoolStatusHistory** database table, copy the greatest value from the Id parameter into the **Student Status History SIS Last Integration Id** field on the Default Configuration page. This value must be copied from the last student status history record imported in step 1. When flows in the

next step are enabled, the integration of student status history records will begin with the Id that's subsequent to the value set in the field.

3. Student status history records will be integrated at 15-minute intervals on the next run of the following flows:

    - CNS-CNE Student School Status History – Student Status History_Pull Based (Engage)
    - CNS-CNE Student School Status History – Student Status History_Create (Engage)

    When the integration cycle ends, the **Student Status History SIS Last Integration Id** field will be automatically updated with the value of the last integrated record. The next run of the integration will begin with the Id that's subsequent to the value set in the field.

**Notes**:

- In the flow **CNS-CNE Student School Status History – Student Status History_Pull Based (Engage)**, records will be "pulled" in batches of 100 until the record count in the source system is less than 10, after which records will not be "pulled" in the current cycle.

  "Pulling" will resume on the next run of the flow after 15 minutes to integrate the remaining records and any new records created during the interval.

- The integration of specific records will be skipped if errors are encountered. In such a scenario, the value in the field **Student Status History SIS Last Integration Id** will be set to the last record that was successfully integrated.

  *Example*

  When integrating records 1 – 300, errors occurred in records 15, 35 and 168. This will cause the **Id** value of record 14 to be stored in the field **Student Status History SIS Last Integration Id**.

    - **Integration cycle 1**

      Two batches of records from 1- 100 and 101 - 200 are integrated.

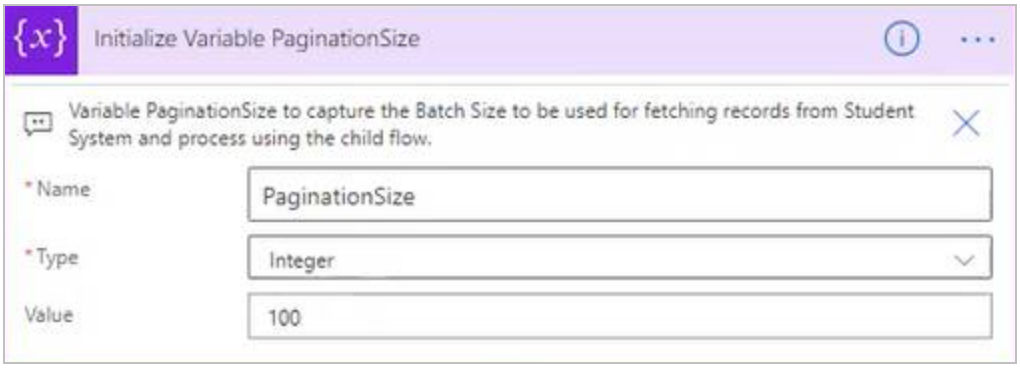
    - **Integration cycle 2**

      Although the integration will begin with record 15 because its subsequent to the Id value of record 14, the system will validate that other records (excluding 15, 35 and 168) are already available, and will skip their integration.

      Thus, the first batch of 100 records will comprise records 15, 35 and 168 and 201-297, which add up to 100 records.

      To complete the integration, records 298-300 will also be integrated in the next cycle.

- The default batch size of 100 records can be changed in the following step in the **CNS-CNE Student School Status History – Student Status History_Pull Based (Engage)** flow:




- The field **Student Status History SIS Last Integration Id** will be updated automatically. Hence its recommended not to manually change its value, except when it must be set before enabling the flows.
- Integrated student status history records will be displayed in reverse chronological order (default) in the contact (Student Progress > Enrollments > Student Status History tab).
- Integration errors will be logged in the **Additional Details** field in the integration log record that's created for every run of the flow **CNS-CNE Student School Status History – Student Status History_Pull Based (Engage)**.

**Integrate Test Scores**

Test score information integrates unidirectionally from Anthology Reach to Anthology Student. By default, integration mapping records for the ACT test are available in Information(StudentTestScore) integration mapping records.

The following steps must be performed to sync new test score types from Anthology Reach to Anthology Student. Before syncing custom test values with Anthology Student, administrators in Anthology Student must ensure that valid values are available in the field **External Test Type Code** for the new test score type.

In the Integration Mapping Record:

1. In the **Internal Field Name**, type the schema name of Test Score field that will be passed from Anthology Reach.
2. In the **External Field Name**, type the schema name of the Anthology Student field that will be updated with the value of the field set in step 1.
3. Specify Test Type information in the following fields:
    - **Test Type** – Set the value of the test type in Anthology Reach that you want to sync with Anthology Student.
    - **External Test Type Code** – Specify the test code that's equivalent to the above test in Anthology Student.
4. In the **Template Type** field, set the value to **StudentIntegrationTestScore**.
5. In the **Mapping Type** field, set the value to **Field**.

The following flows integrate Test Score records from Anthology Reach to Anthology Student:

- **CNE-CNS TestScore Integration Trigger (Engage)**

  This flow is triggered when a test score record is created or updated in Anthology Reach.

- **CNE-CNS TestScore Integration Main (Engage)**

  This flow inherits the Test Score information from the following two flows to integrate the test score details with the Student's available test score record in Anthology Student:

  - **CNE-CNS TestScore Integration Trigger (Engage)** - This flow is called when a TestScore record is created or updated in Anthology Reach.
  - **ApplicationEnrollmentIntegration-WebhookFlow-TestScores** - This flow is called when an application record is synced from Anthology Reach to Anthology Student and the TestScores linked to the Contact record associated with the application need to be synced.

  This flow verifies if the test score record is available in Anthology Student using duplicate check criteria based on the value of the fields **Test Type**, **Test Date**, and **Student**.

  Alternatively, if the test score record is:

  - Not previously available in Anthology Student, a new test score record is created with scores passed in this flow. This flow is also called when a test score that is deleted in Anthology Reach needs to be cleared from Anthology Student.
  - Available in Anthology Student but is blank, i.e., it does not have a **Test Date** or its scores are blank but its **Test Type** is identical to the new record in Anthology Reach, this blank record is updated in Anthology Student with details of the matching record created in Anthology Reach.

**Note**: By default, only scores in which the **Test Source** value is **Official** can be integrated.

OptionSet Values of these **Test Source** types are set as comma-separated values in the **Internal Option Value** field in the integration mapping record **StudentIntegration-TestScore-TestSource-ToIntegrate**. Administrators can customize this record to match their implementation by adding OptionSet values of new **Test Source** types or clearing existing types that are no longer required.

For the Test Score entity, the following table lists fields in Anthology Reach from which information is transmitted to Anthology Student:

| Anthology Reach | | Anthology Student | |
|---|---|---|---|
| Field | Test Type | Field | CNS Test Type Code |
| mshied_actwriting | ACT | Writing Subscore | ACT_2 |
| mshied_actscience | ACT | ACT Science | ACT_6 |
| mshied_actreading | ACT | ACT Reading | ACT_5 |
| mshied_actmath | ACT | ACT Mathematics | ACT_4 |

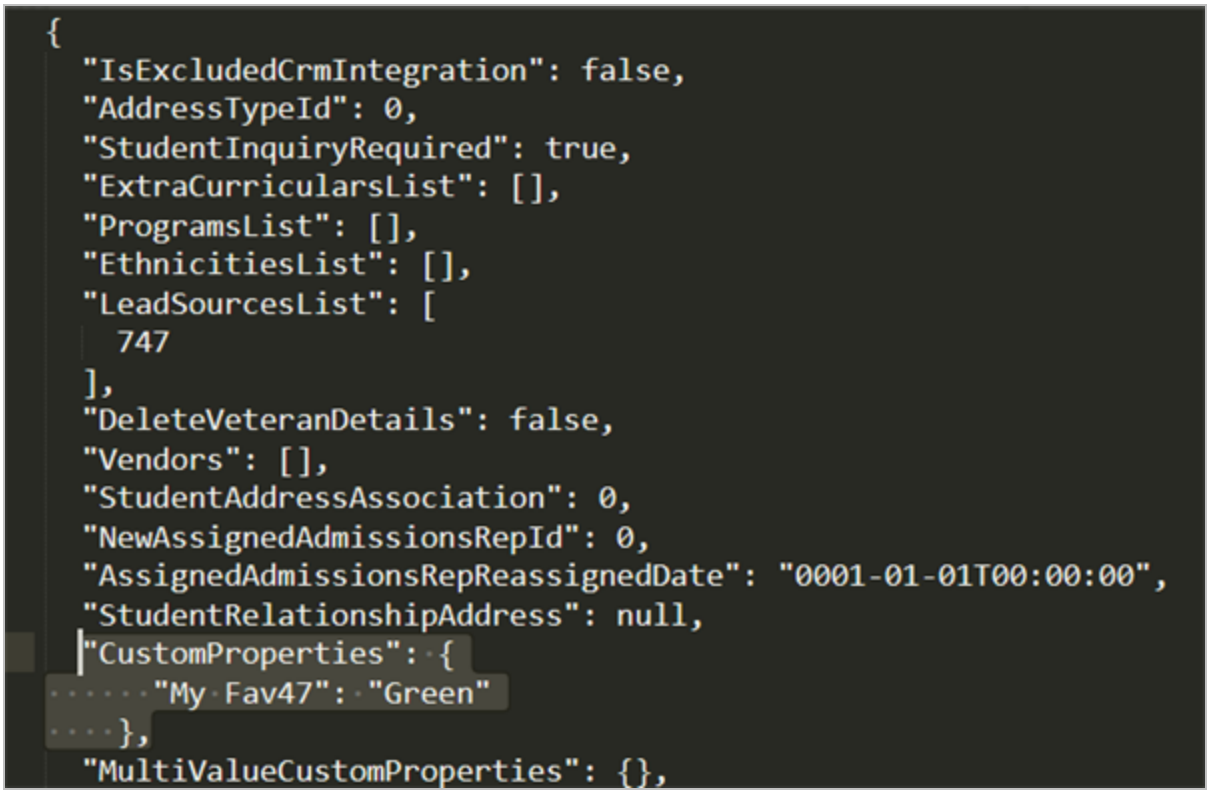
| Anthology Reach | | Anthology Student | |
|---|---|---|---|
| mshied_actenglish | ACT | ACT English | ACT_3 |
| mshied_actcomposite | ACT | ACT Composite | ACT_7 |

**Integrate School-Defined Fields**

In Anthology Student, custom fields for an entity can be created as School-Defined Fields or Extended Properties. This section provides information on how to integrate data for a custom/school-defined field.

Data sent from Anthology Student to Anthology Reach includes the custom fields in the JSON payload.

The following image is a sample JSON payload for an entity with a custom field.




In the above example, My Fav47 is the schema name for the custom field. The custom field is enclosed in the JSON Object "Custom Properties" in the JSON Payload.

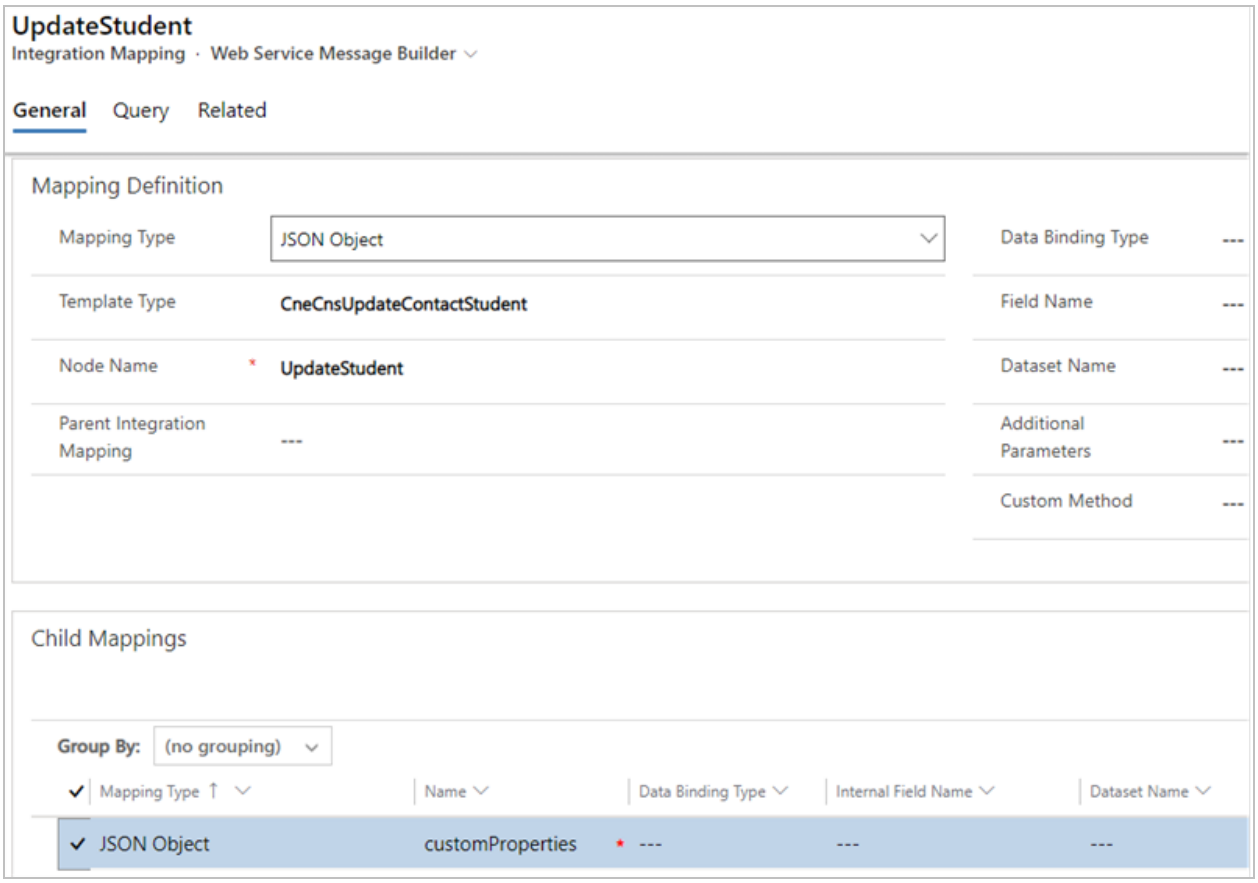
In Anthology Reach, in the field mapping record of the custom field, the following values must be configured:

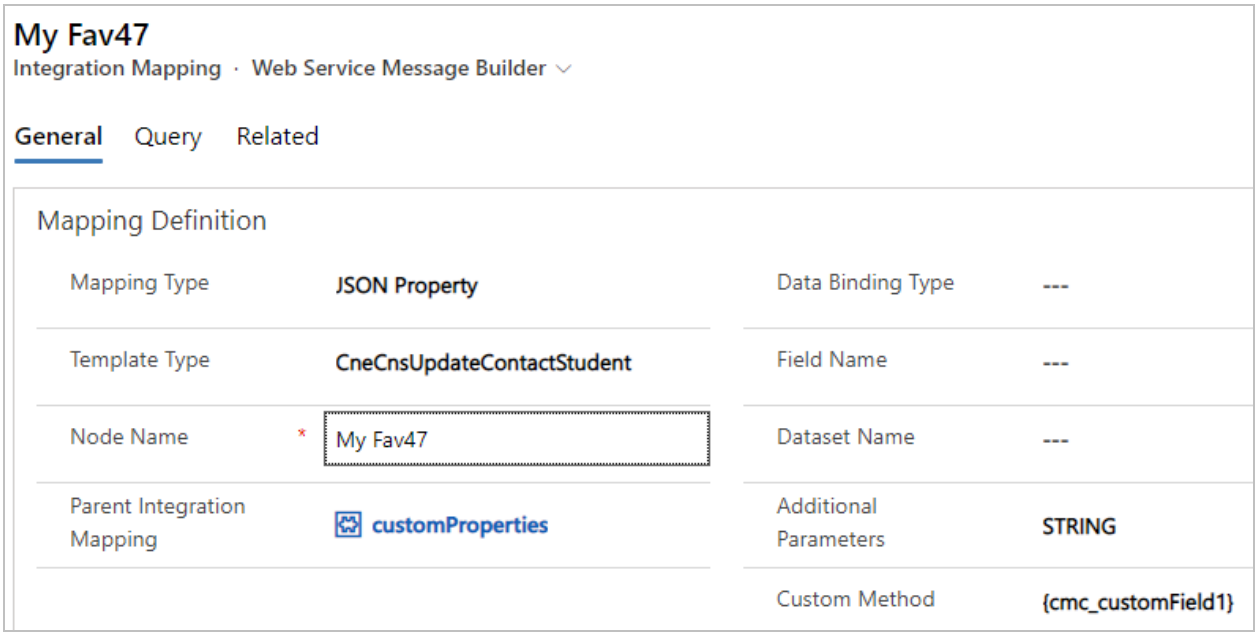
- **External Field Name** - Set it to the JSON Path. For example, CustomProperties.My Fav47
- **Parameters** - CustomProperties.My Fav47.

# From Anthology Reach to Anthology Student

To send the data of a custom field added in an entity in Anthology Reach, the custom field must be added to the JSON data structure in the Field Mapping Template before sending the data to Anthology Student. To do so:

1. In the Integration Mapping Template of the entity, add an integration mapping record with the mapping type as JSON Object.
2. In the Integration Mapping record add a Child Mapping for the custom field with the Mapping Type – "JSON Property".





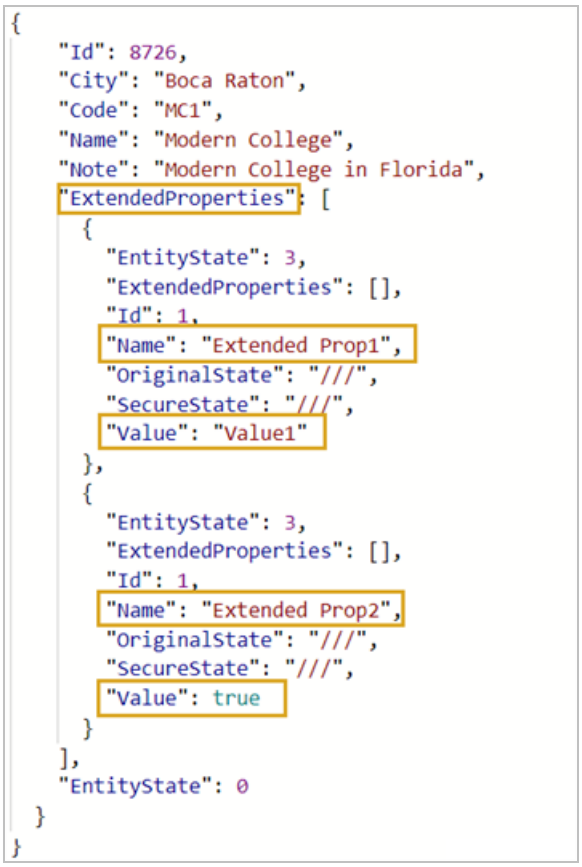

**Integrate Extended Properties**

In Anthology Student, custom fields for an entity can be added as School-Defined Fields or Extended Properties. This section provides information on how to integrate extended properties added for an entity in Anthology Student with custom fields in Anthology Reach.

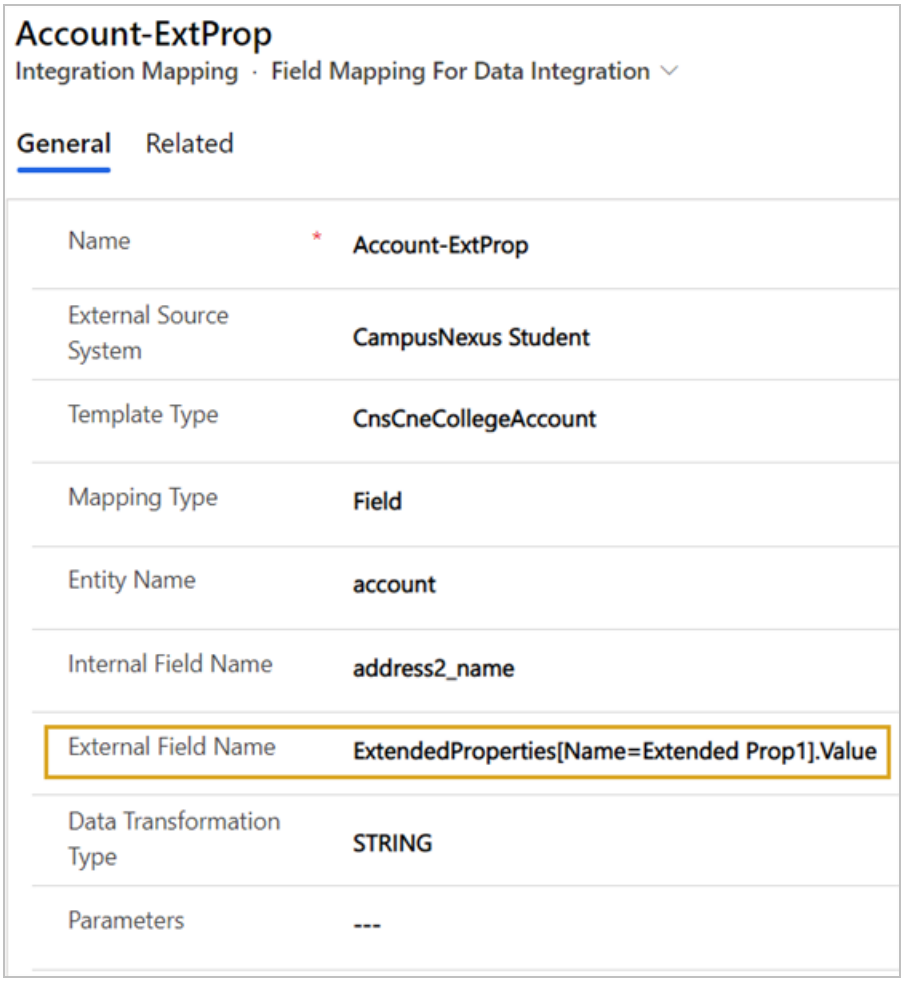
# From Anthology Student to Anthology Reach

Data sent from Anthology Student to Anthology Reach includes the extended properties in the JSON payload.

The following image is a sample JSON payload for an entity with extended properties. The extended properties appear in the payload as a JSON array with the property name as **ExtendedProperties**.




In Anthology Reach, in the field mapping record of the custom field to which the extended property is mapped, update the value of the **External Field Name** as **ExtendedProperties[Name=Extended Prop1].Value**. In the specified value, **ExtendedProp1** is the name of the extended property in the JSON array.



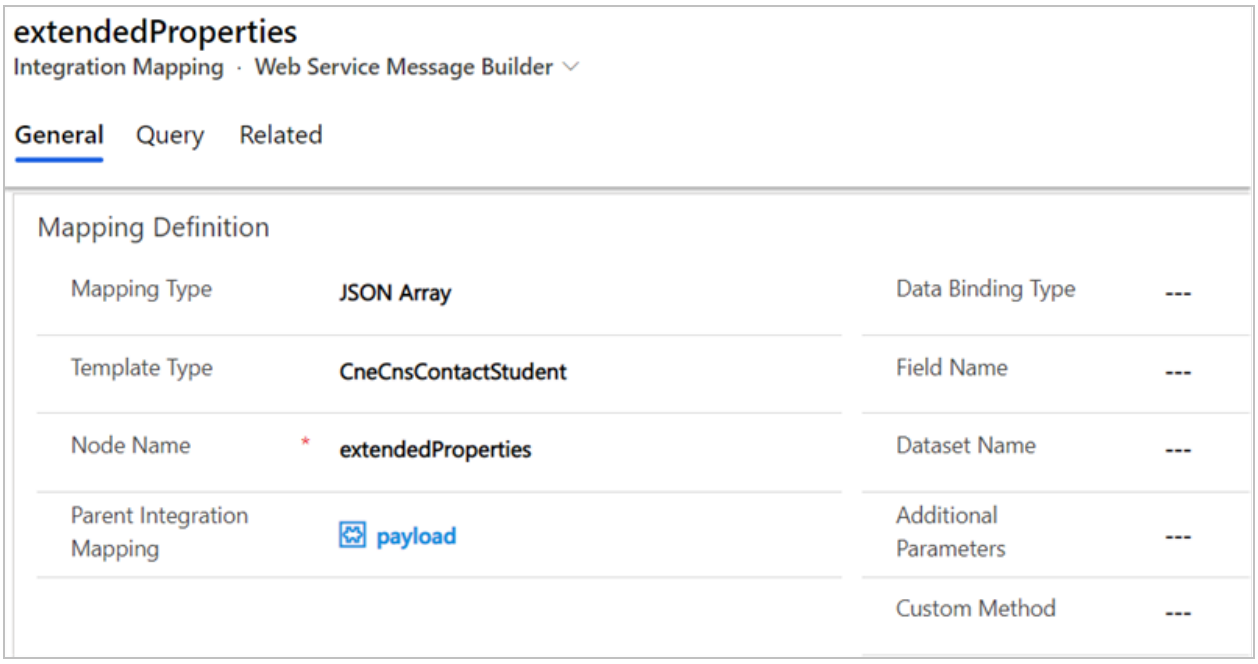

## From Anthology Reach to Anthology Student

The following is an illustration of the JSON payload for extended properties to be sent from Anthology Reach to Anthology Student.

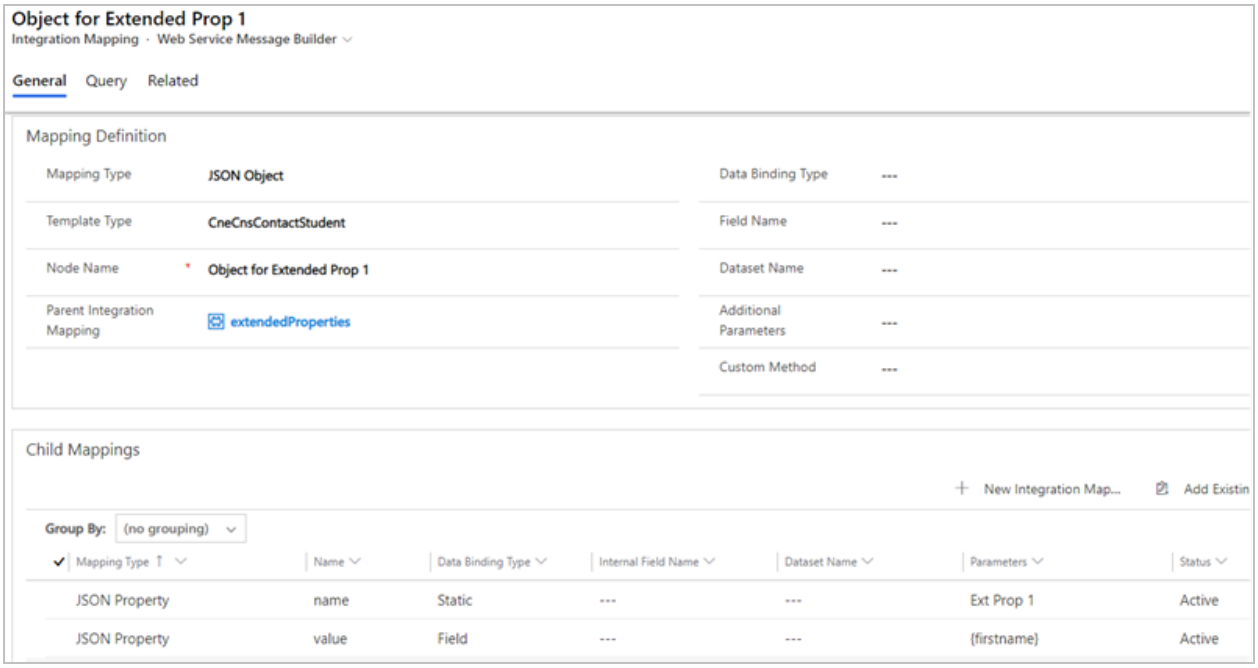
```
{
    "payload": {
      "areaOfStudyTypeId": 3,
      "campusGroupId": 10400,
      "extendedProperties": [
        {
          "name": "Extended Prop1",
          "value": "Value1"
        },
        {
          "name": "Extended Prop2",
          "value": true
        }
      ],
      "id": 56
    }
}
```

Before sending the data of extended properties from Anthology Reach to Anthology Student, the extended properties must be added to the JSON data structure in the Field Mapping Template To do so:

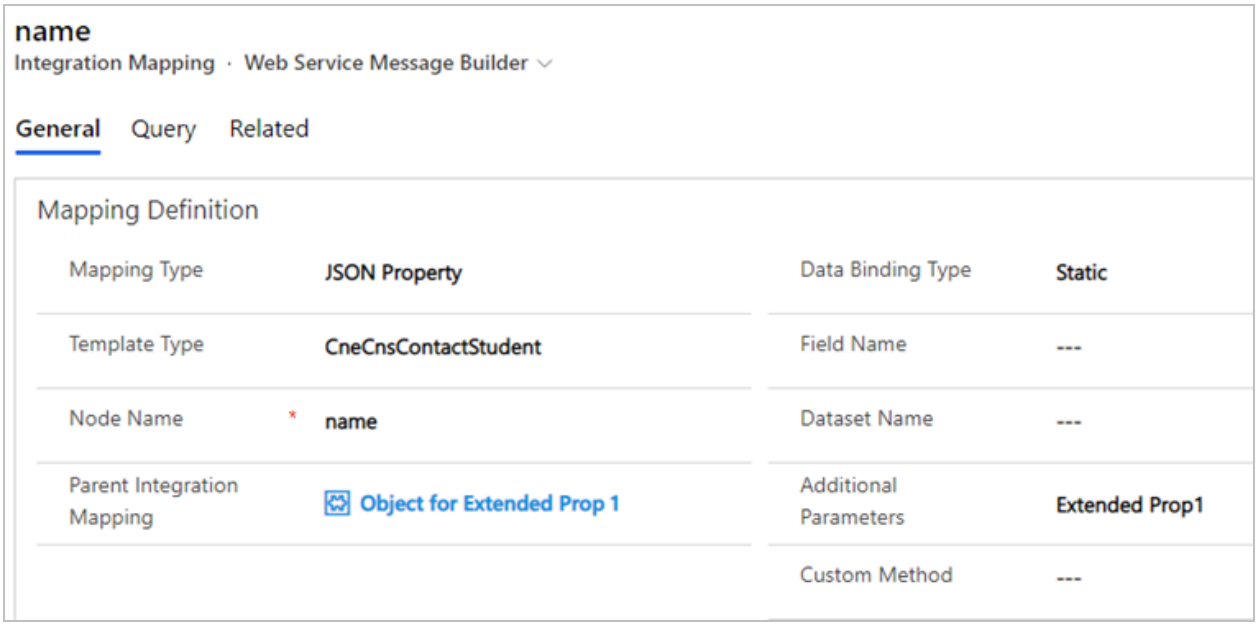
1. For the payload Integration Mapping record of the entity to which extended properties must be added, create a child integration mapping record, **extendedProperties**, and set the value of the **Mapping type** as *JSON Array*.



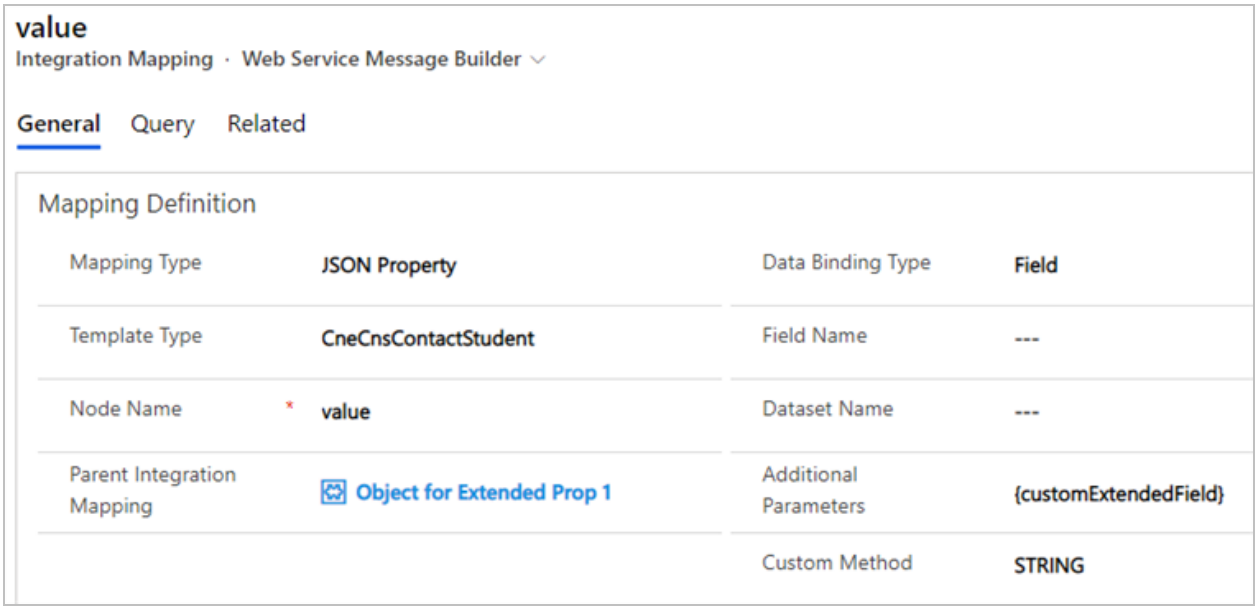

2. In the **extendedProperties** Integration Mapping record, for each extended property, create a child mapping record with **Mapping Type** as *JSON Object* to store the details of the extended property.




3. In the child mapping record of an extended property, create two child Integration Mapping records of type JSON Property, **name** and **value.**
4. For the **name** integration mapping record, set the field values as follows:
    a. **Mapping Type** – JSON Property
    b. **Template Type** – Retain the auto-populated value, which is the Template Type specified in the parent record
    c. **Node Name** – name
    d. **Parent Integration Mapping** – Retain the auto-populated value, which is the parent record
    e. **Data Binding Type** – Static
    f. **Additional Parameters** – Specify the name which should appear in the payload.

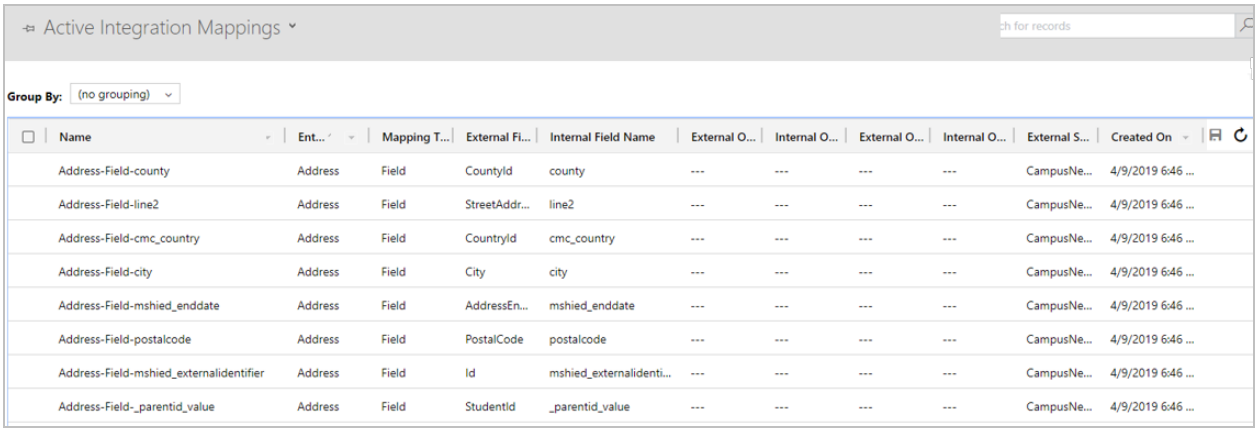


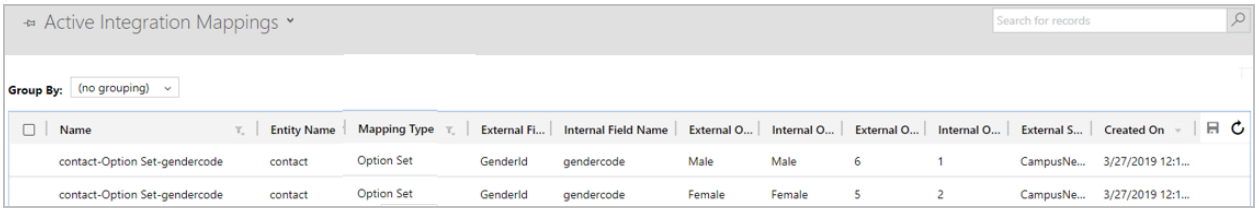

5. For the **value** integration mapping record, set the field values as follows:
    a. **Mapping Type** – JSON Property
    b. **Template Type** – Retain the auto-populated value, the Template Type specified in the parent record
    c. **Node Name** – value
    d. **Parent Integration Mapping** – Retain the auto-populated value, the parent record
    e. **Data Binding Type** – Field
    f. **Custom Method** - Specify the required Data transformation method. For information on Data transformation methods,
    g. **Additional Parameters** – Specify the name which should appear in the payload. Here in this example we have used STRING as the Custom Method for Data transformation and specified the field in the Fetch XML Query which needs to be used to provide the value.




### Map Option Sets

The entity Integration Mapping is used to create the Option Set mappings for the Option set fields which are integrated.




In Anthology Reach > Integration Mapping, the Option Set mapping must be done manually for the following fields:

- Account – Country
- Account – State

- Contact – Last Permanent Residence Country
- Contact – Nationality
- Contact – Race
- Contact – County
- Contact – State
- Contact - Country
- Address – Address Type
- Address – Country
- Address – State
- Address – County
- Academic Period Details – Attendance Type
- Course – Academic Level

Option set mapping provided out of the box with Anthology Reach includes:

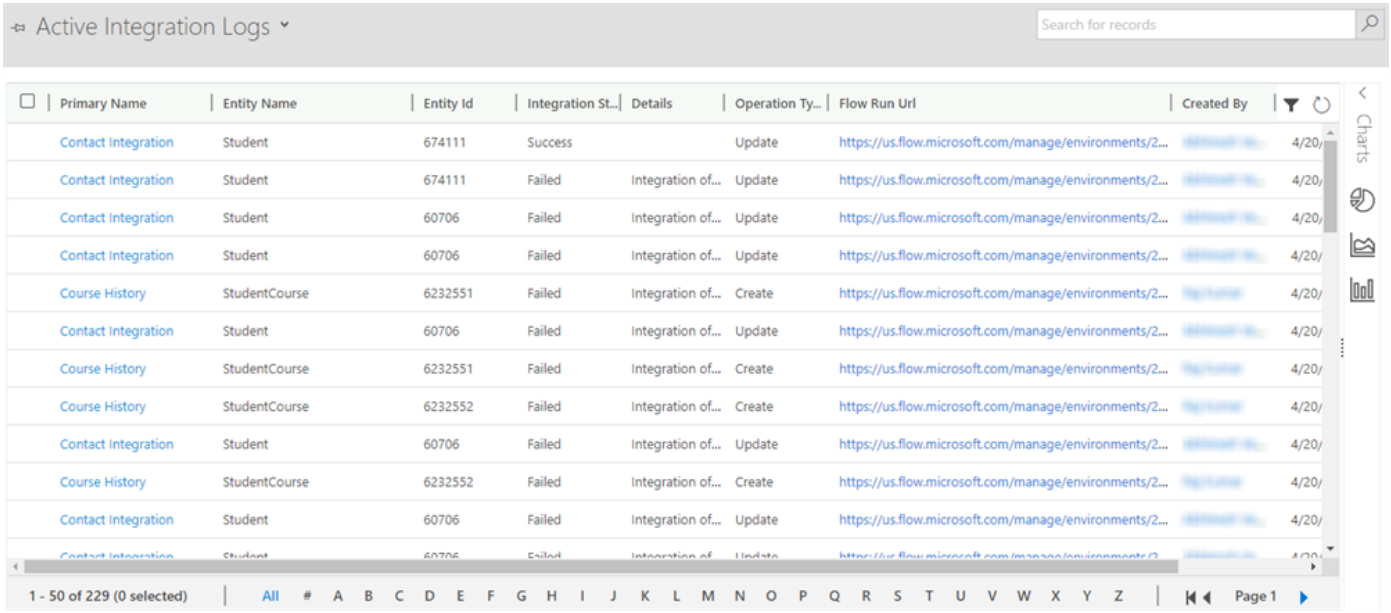
- Account – Account Type
- Contact – Marital Status
- Contact – Gender
- Contact – Contact Type
- Contact – Ethnic Group
- Contact – HIPAA Indicator
- Contact – Suffix
- Address – Mail Type

**Note**: The above mapped options can be extended in Anthology Student, these mappings have to be verified for each customer.

**Review Integration Logs**

For each integration operation, a record of the **Integration Log** entity is created to store the integration details that include:

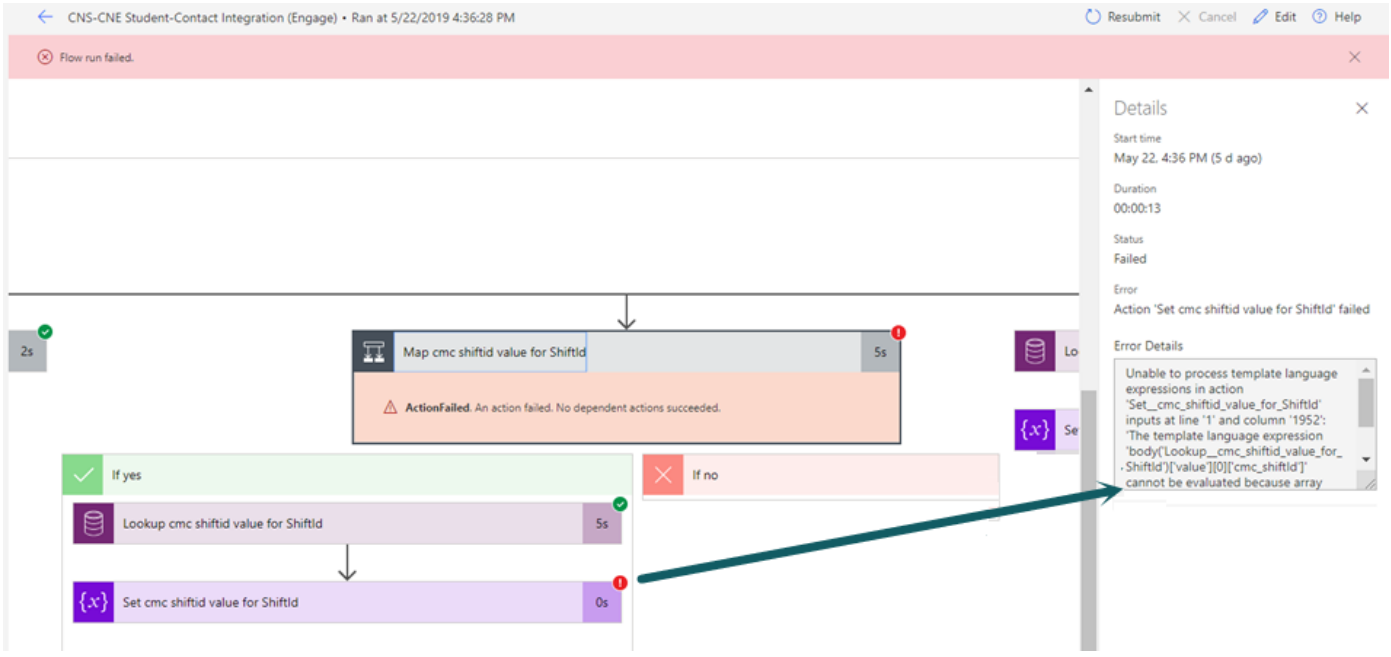
- **Integration Status** - Indicates the status of the integration. It can have one of the following values:
    - Success
    - Failed
- **Entity Name** - Anthology Student entity with its ID in the **Entity Id** field.
- **Primary Name** - Flow name where the flow has failed.
- **Flow Run Url** - URL of the run of the flow that failed. Click the URL to view details of the failure.




# If An Error Occurred in the Flow

In the Integration log record, on clicking the **Flow Run URL** field, the following error message is displayed for the flow:

*"Unable to process template language expressions in action '<**Action Step Name**>' inputs at line '1' and column '1965': 'The template language expression <expression> cannot be evaluated because array index '0' cannot be selected from empty array. Please see https://aka.ms/logicexpressions for usage details.'"*



## Cause

This error can occur if:

- A picklist mapping is not present.
- The record has a lookup value for which data has not been integrated.

## Resolution

- If a picklist mapping is not present, create a picklist mapping.
    a. Create the corresponding picklist value in Anthology Reach by editing the field in an appropriate solution. Note the picklist numerical value generated for this value.
    b. Add an Integration Mapping record with appropriate values for the new mapping based on the value retrieved in the earlier steps.
- If the data for the lookup field has not be integrated, import the required data into Anthology Reach.

  For example, in a Contact record in Anthology Reach, if the **Program** field has a program which is not integrated, the Program must be imported into Anthology Reach.

#### Use Application Insights to Monitor Flows

Administrators can view and monitor the integration details of Anthology Student and Anthology Reach integration flows using Azure Application Insights. Some of the details of the Anthology Student integration flows that can be monitored using the Azure Application Insights include:

- Details of the flow and its run details
- Flow execution statuses
- Flow execution times for Integration related activities

# View the Application Insights for Student Integration Flows

1. Login to https://portal.azure.com.
2. Navigate to the required environment and click **Monitor** > **Logs**.
3. On the right side, in the Query editor, add the query snippet for the flow.

   Use the following query snippets for each type of flow:

   ### Query Snippet for the Anthology Student to Anthology Reach (CNS-CNE) Flows

```
let IntegrationStatusMap = dynamic(
    {
    "175490000": "Success",
```

```
    "175490001": "Failed",
    "175490002": "Pending",
    "175490003": "Success (with warning)"
    });
union traces
| where message startswith "Integration Log added for"
| extend
    CorrelationId = tostring(customDimensions['prop__CorrelationId']),
    EntityName = tostring(customDimensions['prop__EntityName']),
    EntityId = tostring(customDimensions['prop__EntityId']),
    IntegrationStatus = IntegrationStatusMap[tostring(customDimensions['prop__Integ-
rationStatus'])],
    OperationType = tostring(customDimensions['prop__OperationType']),
    IntegrationCategory = tostring(customDimensions['prop__IntegrationCategory']),
    IntegrationSubCategory = tostring(customDimensions['prop__IntegrationSubCategory']),
    StartDateTime = todatetime(customDimensions['prop__StartDateTime']),
    EndDateTime = todatetime(customDimensions['prop__EndDateTime'])
| project
    CorrelationId,
    EntityName,
    EntityId,
    IntegrationStatus,
    IntegrationCategory,
    IntegrationSubCategory,
    OperationType,
    StartDateTime,
    EndDateTime,
    timeTaken = datetime_diff('millisecond', EndDateTime, StartDateTime)
| where IntegrationCategory == 'Reach-Student Integration'
| summarize
    RunCount = count(),
    SuccessCount = countif(IntegrationStatus == 'Success'),
    FailedCount = countif(IntegrationStatus == 'Failed'),
    WarningCount = countif(IntegrationStatus == 'Success (with warning)'),
    PendingCount = countif(IntegrationStatus == ''),
    SuccessRate = (countif(IntegrationStatus in('Success', 'Success (with warning)')) * 100)
/ count(),
    AvgRunTime = avg(timeTaken)
    by EntityName
```

**Query Snippet for the Anthology Reach to Anthology Student (CNE-CNS) Flows**

```
let IntegrationStatusMap = dynamic(
    {
    "175490000": "Success",
    "175490001": "Failed",
    "175490002": "Pending",
    "175490003": "Success (with warning)"
    });
union traces
| where message startswith "FLOWEND - Executed Flow with FlowKeyName "
| extend
    CorrelationId = tostring(customDimensions['prop__CorrelationId']),
    FlowKey = tostring(customDimensions['prop__FlowKey']),
    startTime = todatetime(customDimensions['prop__startTime']),
    endTime = todatetime(customDimensions['prop__endTime']),
    status = tostring(customDimensions['prop__status'])
| project
```

```
    CorrelationId,
    FlowKey,
    startTime,
    endTime,
    status,
    datetime_diff('millisecond', endTime, startTime)
| join kind = leftouter (
    union traces
    | where message startswith "Integration Log added for"
    | extend
        CorrelationId = tostring(customDimensions['prop__CorrelationId']),
        EntityName = tostring(customDimensions['prop__EntityName']),
        EntityId = tostring(customDimensions['prop__EntityId']),
        IntegrationStatus = IntegrationStatusMap[tostring(customDimensions['prop__Integ-
rationStatus'])],
        OperationType = tostring(customDimensions['prop__OperationType']),
        IntegrationCategory = tostring(customDimensions['prop__IntegrationCategory']),
        IntegrationSubCategory = tostring(customDimensions['prop__IntegrationSubCategory']),
        StartDateTime = todatetime(customDimensions['prop__StartDateTime']),
        EndDateTime = todatetime(customDimensions['prop__EndDateTime'])
    | project
        CorrelationId,
        EntityName,
        EntityId,
        IntegrationStatus,
        IntegrationCategory,
        IntegrationSubCategory,
        OperationType,
        StartDateTime,
        EndDateTime,
        timeTaken = datetime_diff('millisecond', EndDateTime, StartDateTime)
    )
    on CorrelationId
| summarize
    RunCount = count(),
    SuccessCount = countif(IntegrationStatus == 'Success'),
    FailedCount = countif(IntegrationStatus == 'Failed'),
    WarningCount = countif(IntegrationStatus == 'Success (with warning)'),
    PendingCount = countif(IntegrationStatus == ''),
    SuccessRate = (countif(IntegrationStatus in('Success','Success (with warning)')) /count
())*100,
    AvgRunTime = avg(timeTaken)
    by FlowKey
```

**Query Snippet for the Pull Based Student Integration Flows**

```
let IntegrationStatusMap = dynamic(
    {
    "175490000": "Success",
    "175490001": "Failed",
    "175490002": "Pending",
    "175490003": "Success (with warning)"
    });
union traces
| where message startswith "Integration Log added for"
| extend
    CorrelationId = tostring(customDimensions['prop__CorrelationId']),
    EntityName = tostring(customDimensions['prop__EntityName']),
```

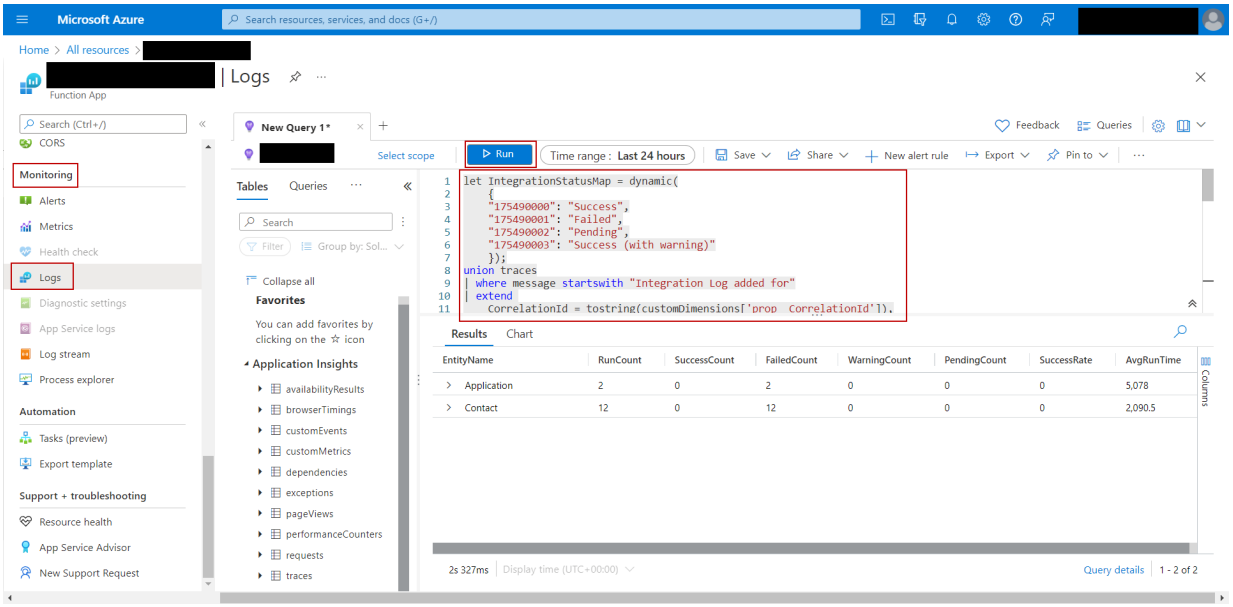
```
    EntityId = tostring(customDimensions['prop__EntityId']),
    IntegrationStatus = IntegrationStatusMap[tostring(customDimensions['prop__Integ-
rationStatus'])],
    OperationType = tostring(customDimensions['prop__OperationType']),
    IntegrationCategory = tostring(customDimensions['prop__IntegrationCategory']),
    IntegrationSubCategory = tostring(customDimensions['prop__IntegrationSubCategory']),
    StartDateTime = todatetime(customDimensions['prop__StartDateTime']),
    EndDateTime = todatetime(customDimensions['prop__EndDateTime'])
| project
    CorrelationId,
    EntityName,
    EntityId,
    IntegrationStatus,
    IntegrationCategory,
    IntegrationSubCategory,
    OperationType,
    StartDateTime,
    EndDateTime,
    timeTaken = datetime_diff('millisecond', EndDateTime, StartDateTime)
| where IntegrationCategory == 'Student-Reach Pull Integration'
| summarize
    RunCount = count(),
    SuccessCount = countif(IntegrationStatus == 'Success'),
    FailedCount = countif(IntegrationStatus == 'Failed'),
    WarningCount = countif(IntegrationStatus == 'Success (with warning)'),
    PendingCount = countif(IntegrationStatus == ''),
    SuccessRate = (countif(IntegrationStatus in('Success', 'Success (with warning)')) * 100)
/ count(),
    AvgRunTime = avg(timeTaken)
    by EntityName
```

4. Click **Run**.

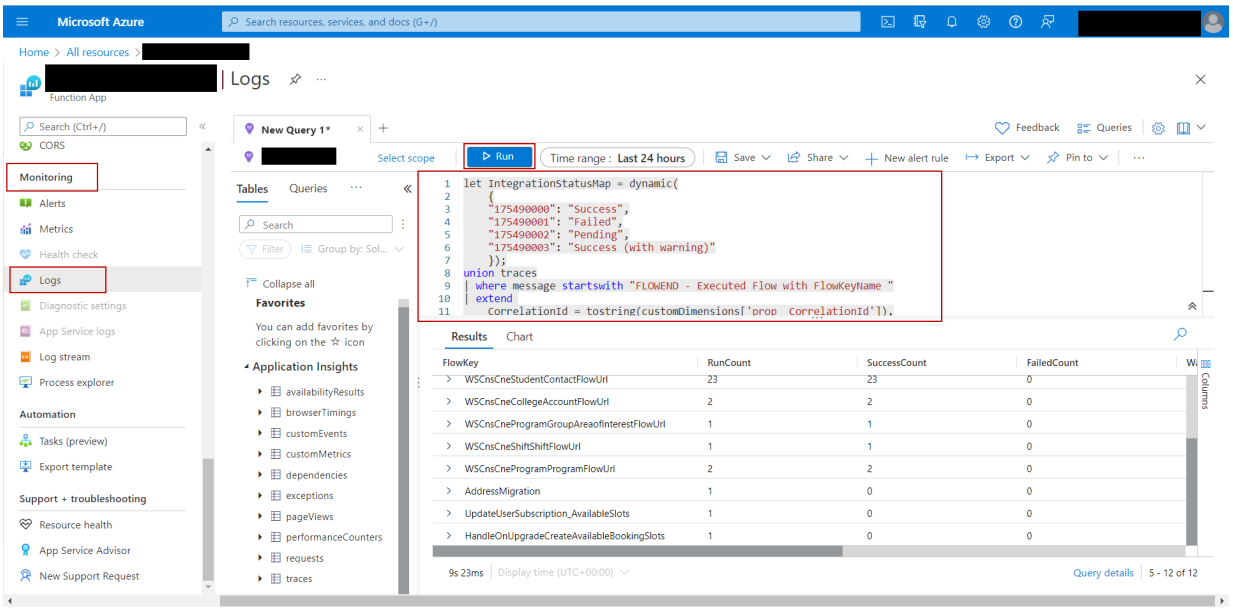
   The Application Insight details for the flows will be generated and displayed.

   The following images illustrate the Application Insights details generated for each type of integration flow:

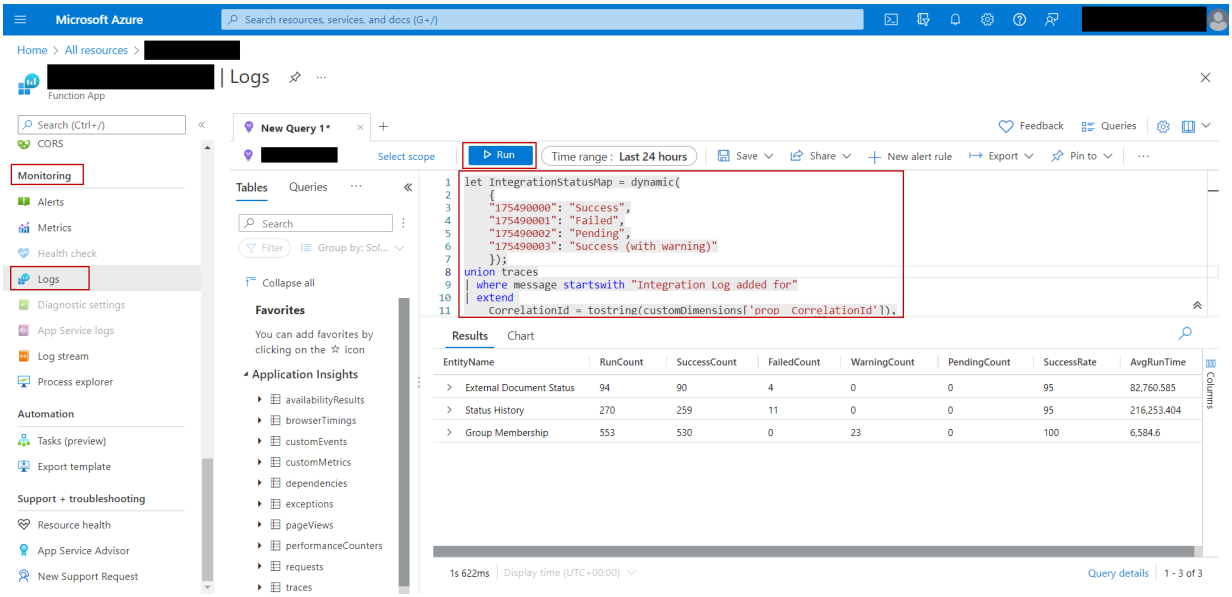
- **Anthology Student to Anthology Reach (CNS-CNE) Flows**




- Anthology Reach **to Anthology Student (CNE-CNS) Flows**




- **Pull Based Flows Student Integration Flows**




## Service Bus Integration

The Anthology Student - Anthology Reach integration using the Azure Service Bus integration architecture:

- Supports bi-directional integration between both systems
- Supports synchronous and asynchronous integration
- Supports a non-invasive introduction of new features or integrations
- Reduces the dependency on Anthology Student releases

### Architecture

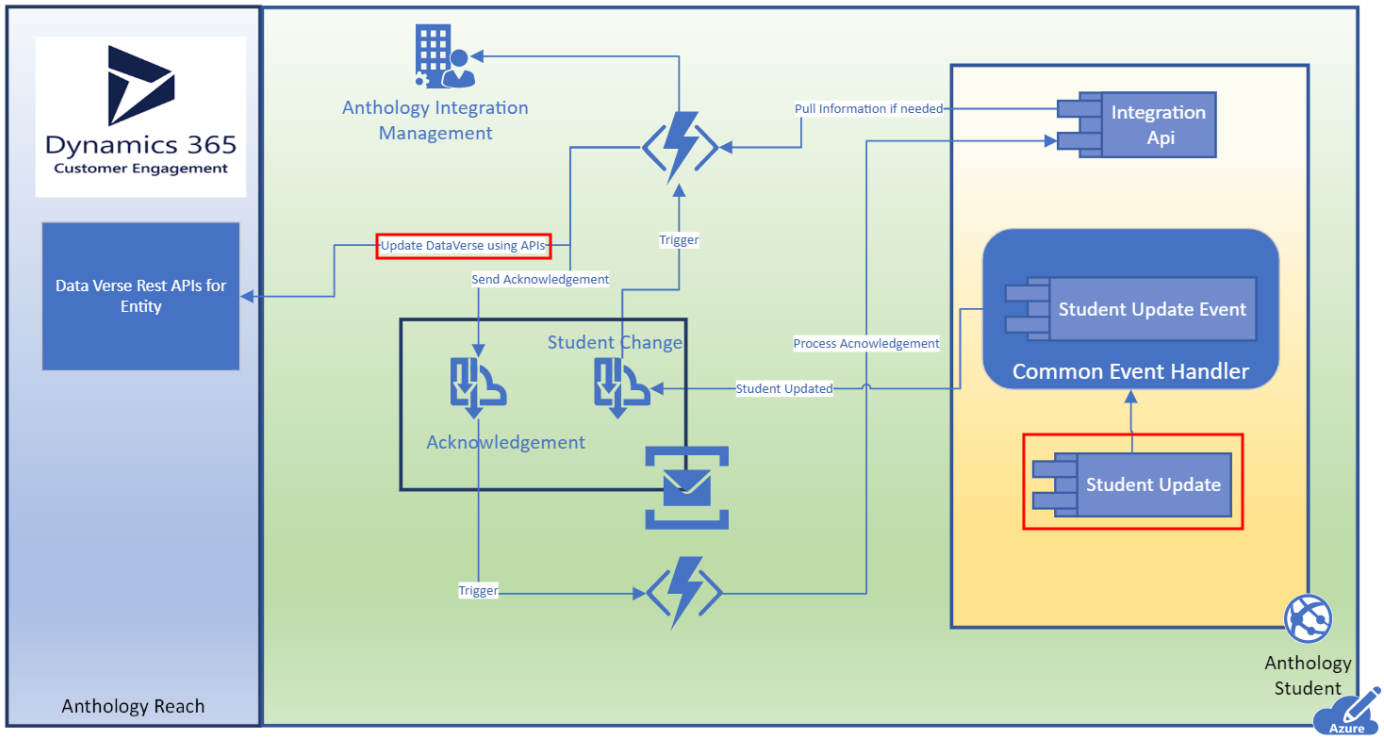
Asynchronous integration from Anthology Student to Anthology Reach:

- Uses the common event handler (CE) to push data to the Azure Service Bus for a subscribed topic or business event.
- Triggers a logic app or Azure Function which calls the Dataverse API to push the data back to Anthology Reach when a message is published to the Service Bus
- Sends an acknowledgement message to the Service Bus which then triggers an Azure Function to update the data in Anthology Student
- Can send an acknowledgement or response message if the data processing at the target involves a workflow

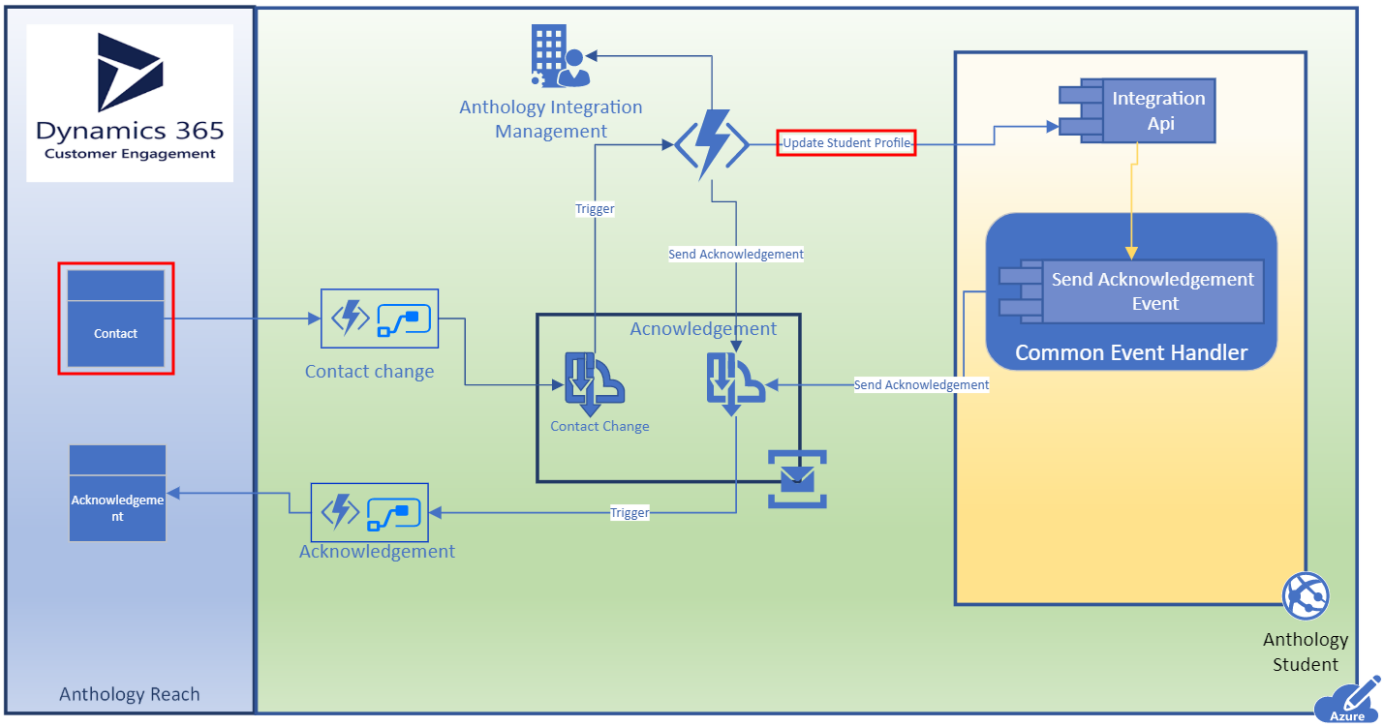
Synchronous integration from Anthology Student to Anthology Reach:

- Can use an HTTP triggered Azure Function to pull data from Anthology Reach on request of data in Anthology Student
- Can use an event handler to call an Azure Function from the event in Anthology Student

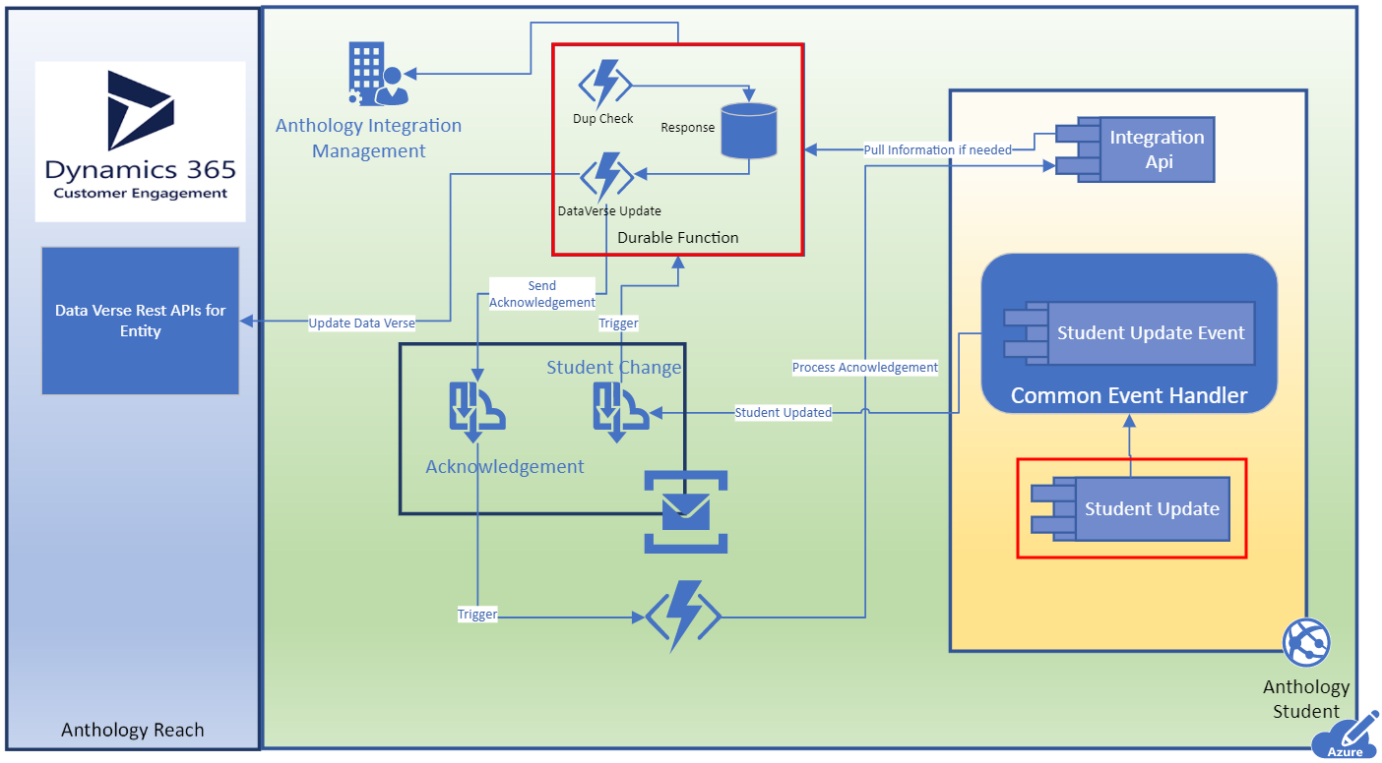
The following image shows the **basic data flow** from Anthology Student to Anthology Reach with updates propagated to the Dataverse API.



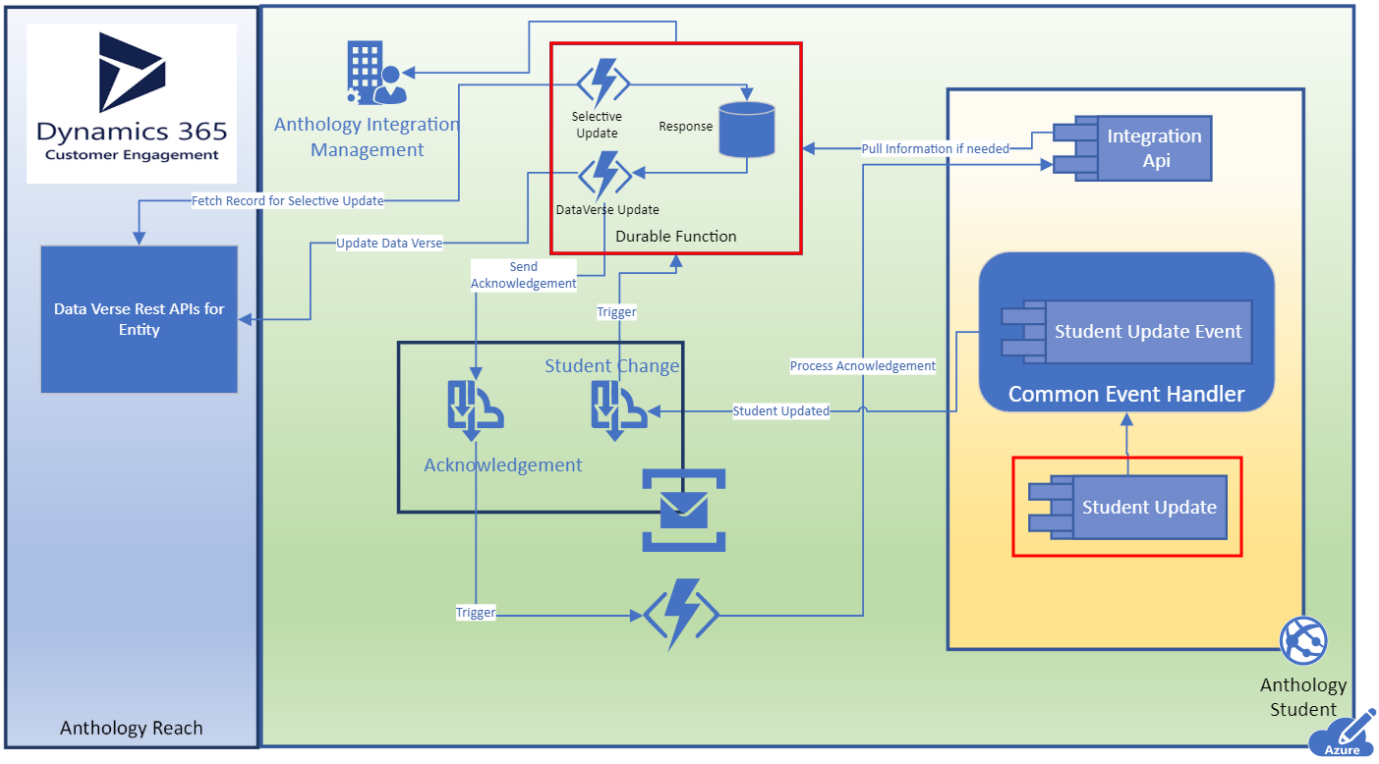

The following image shows the data flow from Anthology Reach to Anthology Student when a **Contact** is updated in Anthology Reach and the updates are propagated to the Student Profile in Anthology Student.



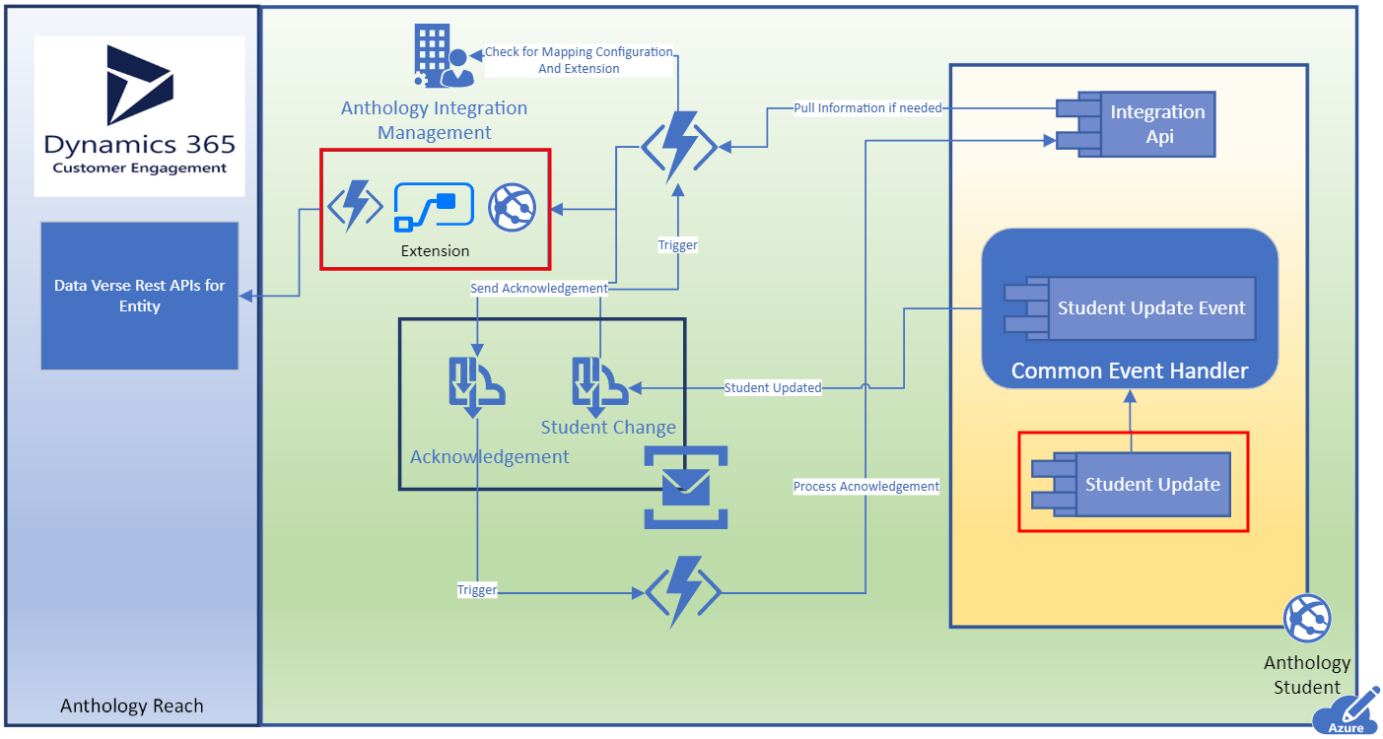
Anthology Student and Anthology Reach use the standard **duplicate check APIs** available in the applications.




Anthology Student and Anthology Reach use a Durable Azure Function for **selective updates** that impact only specific fields in a database record such as [Student Course Status Has Changed].

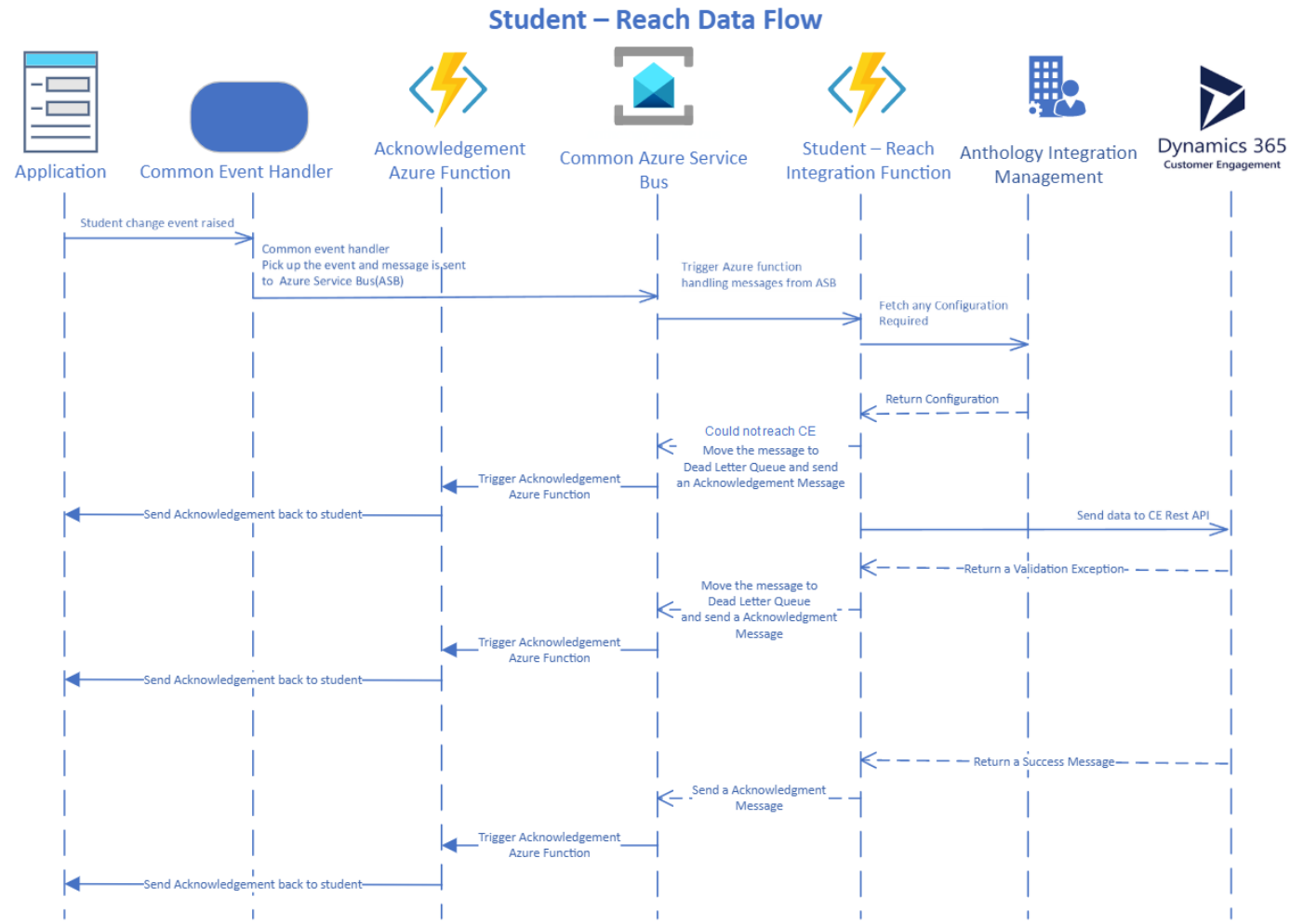


Configurations in Anthology Reach are provided to configure **extensions**. Extensions are in the form of HTTP URLs. The URLs will be used to post any data from the Azure Function.

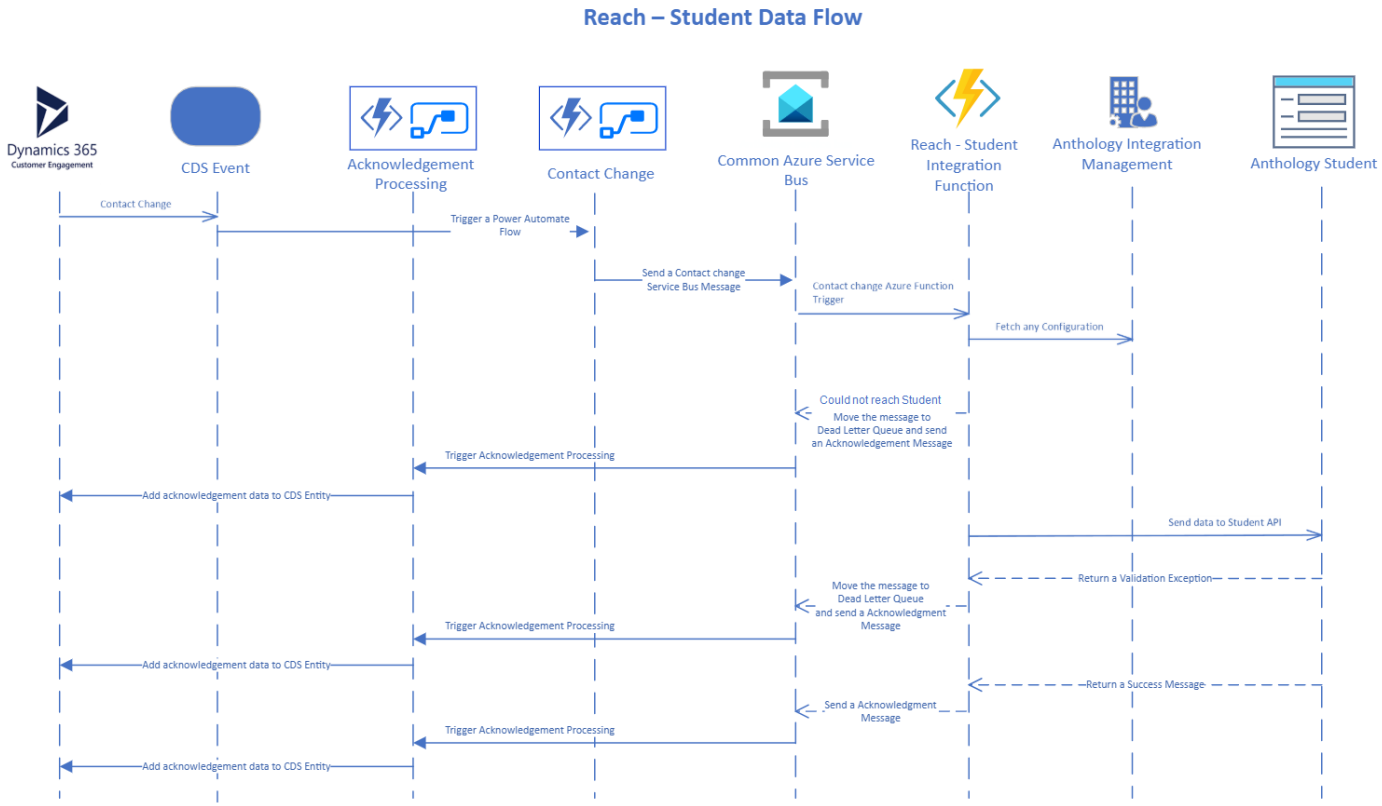



### Message Flow

The following image shows the message flow from Anthology Student to Anthology Reach.




The following image shows the message flow from Anthology Reach to Anthology Student.




## Configure Anthology Student Settings

The Azure Service Bus integration between Anthology Reach and Anthology Student requires the system administrator to configure topic subscriptions in the Settings area of Anthology Student.

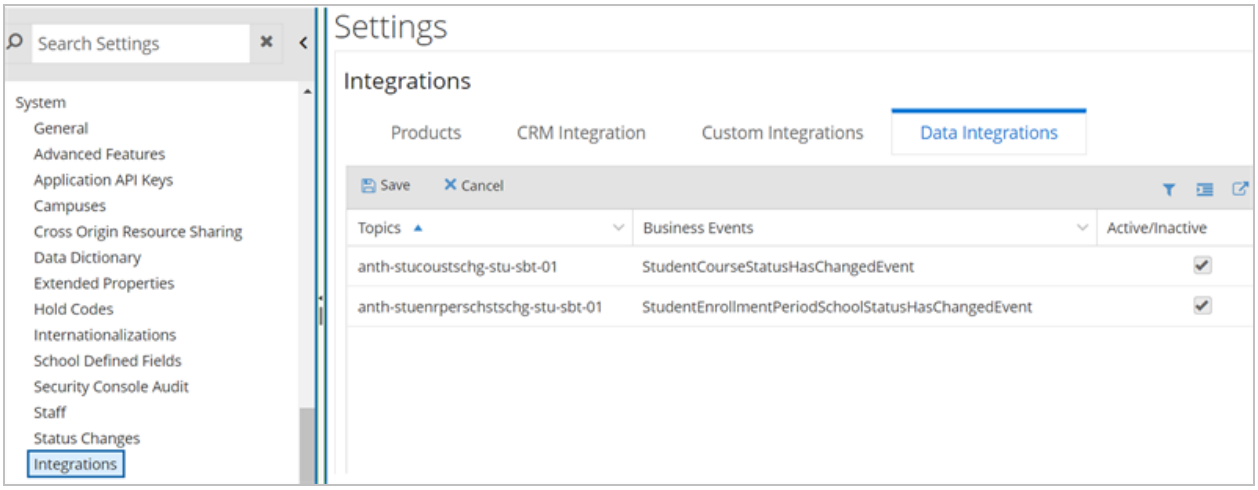
The connection between Anthology Student and the Azure Service Bus is created through pipelines. All available topics will be deployed as part of the pipelines.

Once the Azure Service Bus connection is established, the system administrator can view the list of available topics and activate topics. Once a topic is activated, it can only be deactivated through SQL scripts.

The system administrator must have "System - Settings - Manage" authorization in the Security Console for Anthology Student.

To activate topics:

1. In the cloud-based Anthology Student web app, navigate to **Settings** > **System** > **Integrations**.
2. Click the **Data Integrations** tab.
3. Select the check box in the **Activate/Inactive** column to activate a topic.
4. Click **Save**.




## Business Events

Business events are a set of events (and underlying code) that clients can subscribe to in the Anthology cloud. The business events, also referred to as topics, provide integrations for specific tasks and products.

**Student Course Status Has Changed**

Event: **StudentCourseStatusHasChangedEvent**

Topic: **anth-stucoustschg-stu-rch-sbts-01**

The StudentCourseStatusHasChanged business event is raised in every place within the Anthology Student application where a saved unit of work includes a change to the status property on the StudentCourse entity (Status column in the AdEnrollSched table). The event is raised if the column that was updated is the Status column AND the value is different from what it was before. The event is also raised when new rows are inserted into the StudentCourse entity where the initial value of Status is set to "F".

**Student Course Status Change Events**

| Event Name | Raised when a student course status is updated to ... |
|---|---|
| StudentCourseStatusHasChanged | Any value |
| StudentCourseStatusHasChanged *ToRegistered* | Registered (“S”) |
| StudentCourseStatusHasChanged *ToCurrentlyAttending* | Currently Attending (“C”) |
| StudentCourseStatusHasChanged *ToGradePosted* | Grade Posted (“P”) |
| StudentCourseStatusHasChanged *ToDrop* | Drop (“D”) |
| StudentCourseStatusHasChanged *ToFuture* | Unregister or newly added course row |

The StudentCourseStatusHasChanged event will be raised for individual students, in batch processing, or within other processes that change student statuses, including inside stored procedures. These processes include:

- Course Registration
- Attendance Posting
- Grade Posting
- Student School Status Changes

The event can be raised from Anthology Student, Student Portal, Faculty Portal, and APIs (Registration API, Attendance API, Transfer Credit API, and Post Grade API).

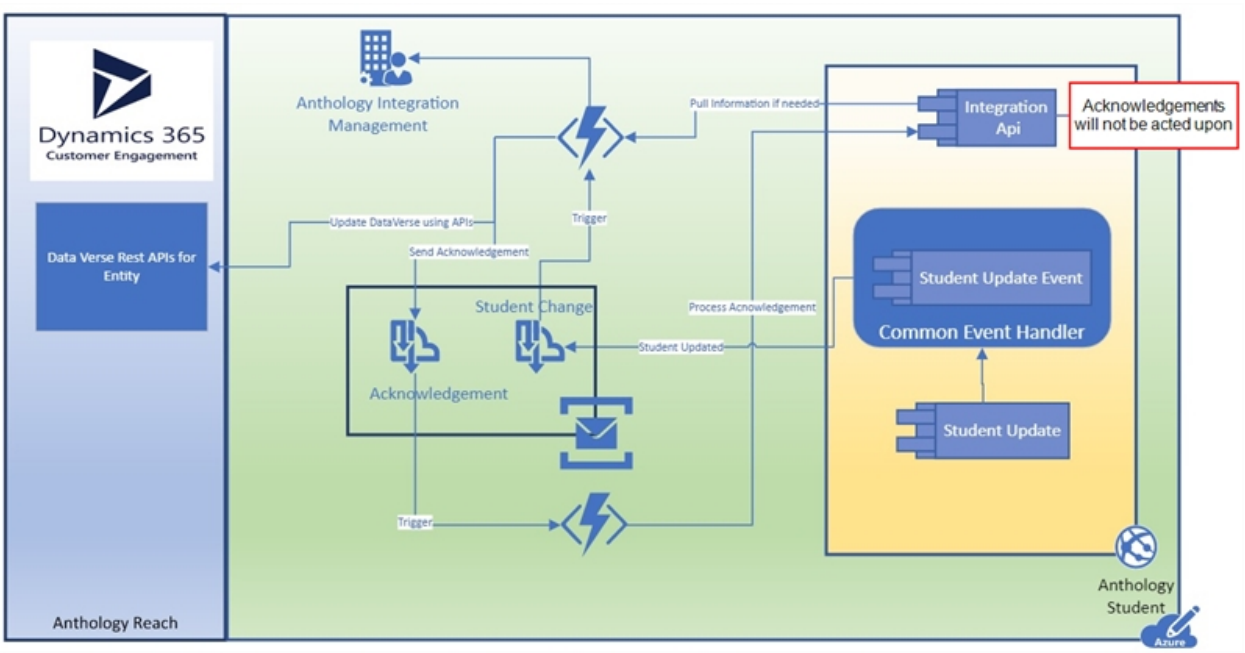
When the event is triggered, the following properties are pushed to the Azure Service Bus:

- Student Course Status
- Previous Course Status
- Enrollment ID
- Student ID
- Term ID
- Course ID
- Enrollment Period ID
- AD Program ID
- Course Status Change Reason

- Class Section ID
- Person ID

The event message is populated with the applicable data. The message includes properties that are commonly needed for all business events raised as well as properties that are specific to the context of the business event being raised. For these specific business events, the previous and new values of student course status are included in the message. Other properties may be included as well.

**Note**: The event will not act upon acknowledgements.




For events that originate from the Anthology Student database such as "Student Course Status Has Changed", triggers on the relevant tables are the mechanism by which events are raised when records in the tables are updated. Triggers insert messages into the queue tables that are then processed by the Service Module Host. For any new events being added, messages are inserted into the DatabaseTriggeredEventsServiceQueue table by new database triggers.

A common event handler (CE) monitors all Anthology Student database events. The CE extracts the event names and payloads and places the messages on the Service Bus or on multiple Service Buses in multi-tenant environments.

Once the data is posted successfully to the Dataverse, an acknowledgement is returned to Anthology Student via an Azure Function.

# Integration Example

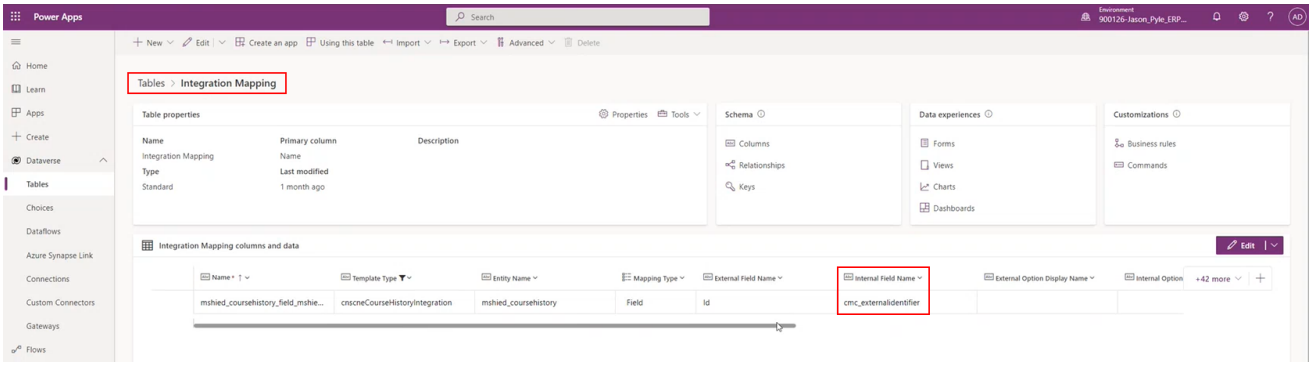
- Anthology Student and Anthology Reach are integrated in terms of reference data such as:
    - Courses
    - Course/class sections

- Students
- Academic periods/terms

If the reference data required for the use case is not available, handled exceptions are returned to the source system.

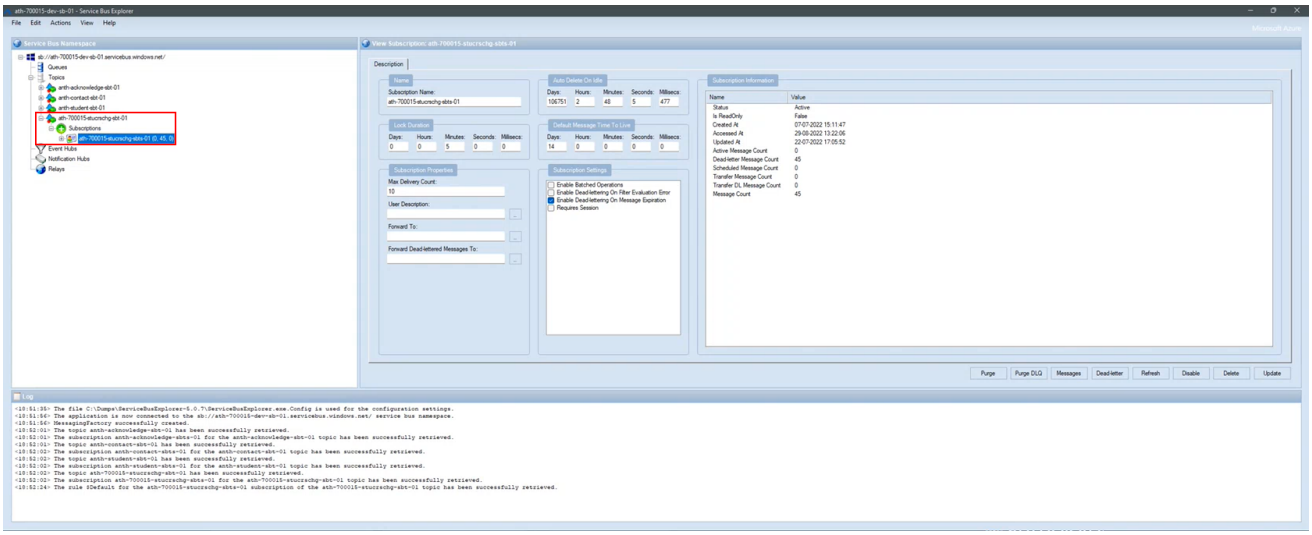
The integration of reference data is part of flow-based events in Anthology Reach. Please see Integrated Entities.

- In the Power Apps for Anthology Reach, the **cmc_externalidentifier** is mapped under Tables > Integration Mapping.




- In Service Bus Explorer, a subscription exists for the "StudentCourseStatusChange" topic.

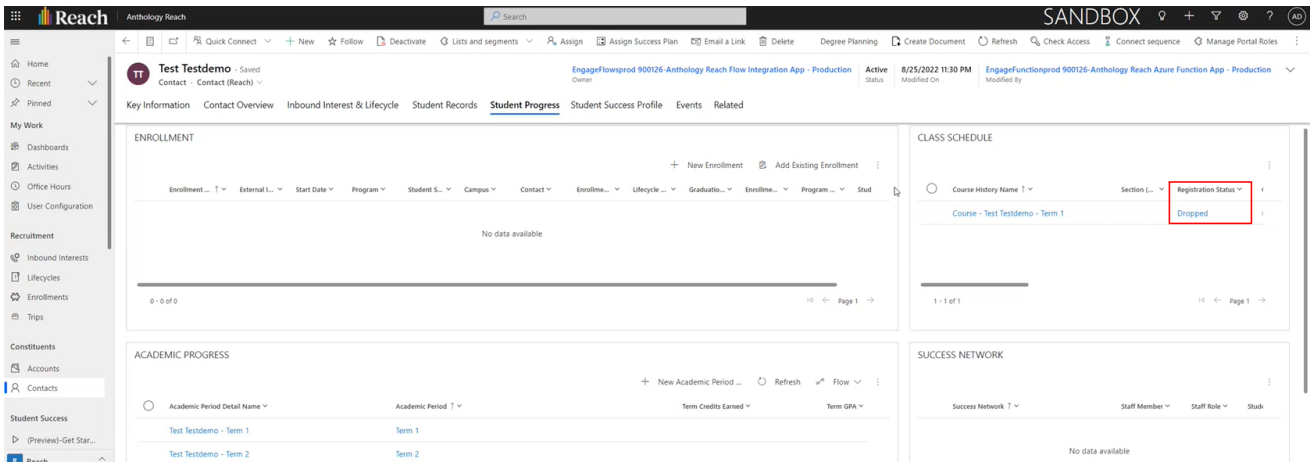
**Note**: A topic is a collection of Azure Functions and logic apps. Available topics are based on business needs. Each topic can cover multiple entities. A topic can have multiple subscriptions. Azure Service allows 10,000 topics and each topic can have 200 subscriptions. The Service Bus creates a dead letter queue for multiple subscriptions.



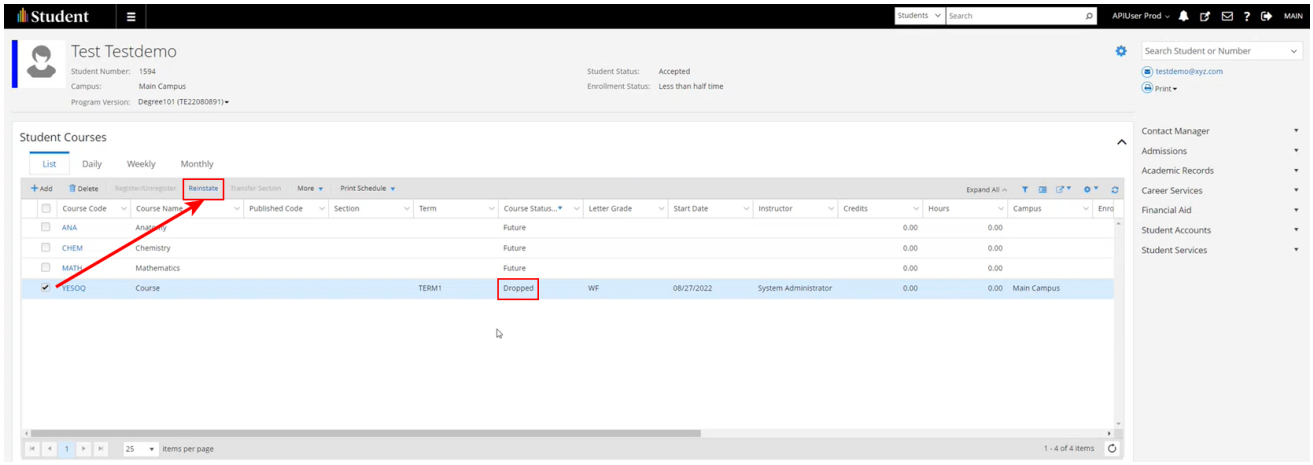
- The StudentCourseStatusChange topic is activated in the Anthology Student settings (see Configure Anthology Student Settings).

# Check the Data Flow

1. In Anthology Reach under Contacts > Student Progress, check the enrollment record for a sample student. The Registration Status is "Dropped".

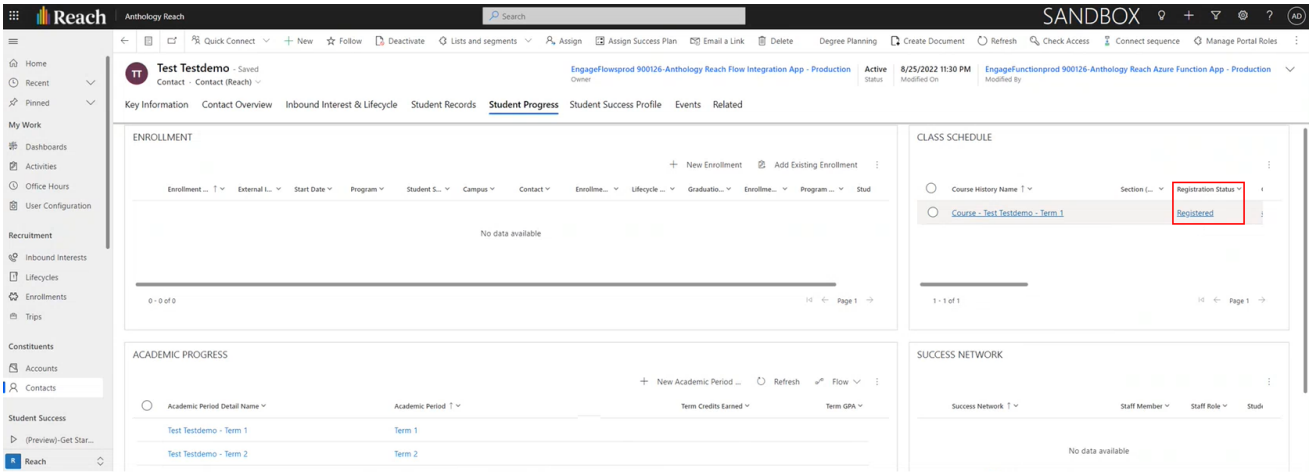



2. In Anthology Student under Academic Records > Student Courses, reinstate the student's enrollment .




At this point, the Service Module Host invokes the Azure Service Bus and triggers an Azure Function named "StudentCourseStatusChangeServiceBusTrigger". An acknowledgement is returned to the Integration API, and the Status change is registered in the Dataverse.

3. In Anthology Reach, refresh the Registration Status. Note that it changed from "Dropped" to "Registered".



## Deployment

### Flow-based Integration

#### Task Matrix

| No. | Resource | Procedure | Anthology Team | Customer |
|---|---|---|---|---|
| 1 | PowerApps | Prepare Flow Connections<br>• Create CDS (Current Environment) Connection<br>• Create Anthology Student Custom Connector<br>• Create Anthology Student Custom Connector Connection | Professional Services | All customers |
| 2 | MS Dynamics | Export Enabled Flows | | |
| 3 | Azure DevOps | Update Pipeline Variables and Run the Release Pipeline | | |
| 4 | Anthology Reach | Run Default Data for Anthology Student Integration<br>• Map the Option Sets in Integration Mapping<br>• From Anthology Reach to Anthology Student | | |
| 5 | Anthology Reach | Update Default Integration Mapping Records<br>• Import Integration Mapping Templates | | |
| 6 | MS Dynamics | Configure Integration Settings in the Configuration Record | | |
| 7 | Anthology Student | Configure the CNS-CNE Integration URL | | |
| 8 | Workflow Composer | Install Anthology Student Workflows | | |
| 9 | Anthology Reach | Enable Power Automate Flows | | |

### Service Bus Integration

#### Task Matrix

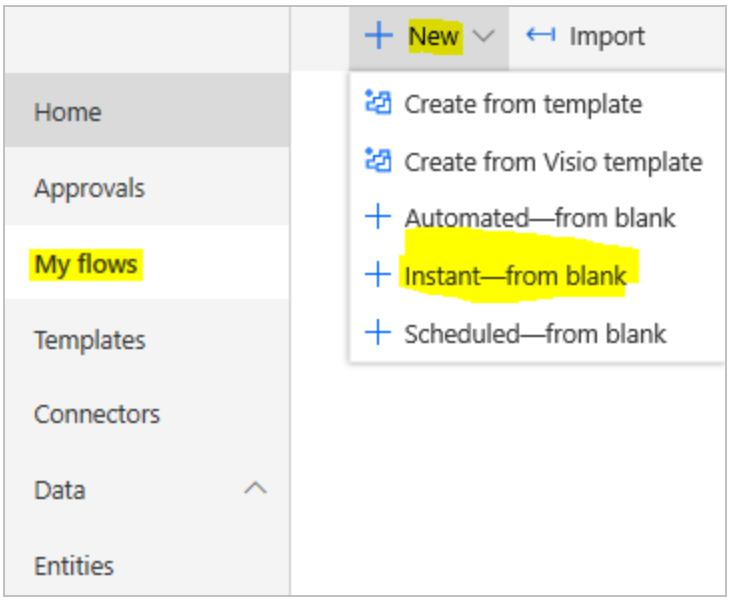
| No. | Resource | Procedure | Anthology Team | Customer |
|---|---|---|---|---|
| 1 | Anthology Tenant | Configure Azure Service Bus | | |
| 2 | | Configure Azure Functions | | |
| 3 | | Deploy DB scripts to register events to the common event handler (CE) | | |
| 4 | Client Tenant | Deploy Anthology Reach to the MS Dynamics environment | | |
| 5 | | Configure Azure applications to authenticate Dataverse API calls | | |
| 6 | Anthology Student | Deploy customer event handler (DLL) to the Anthology Student working directory (bin folder) | | |

### Deploy Flow-based Integration

#### Prepare Flow Connections

## Create CDS (Current Environment) Connection

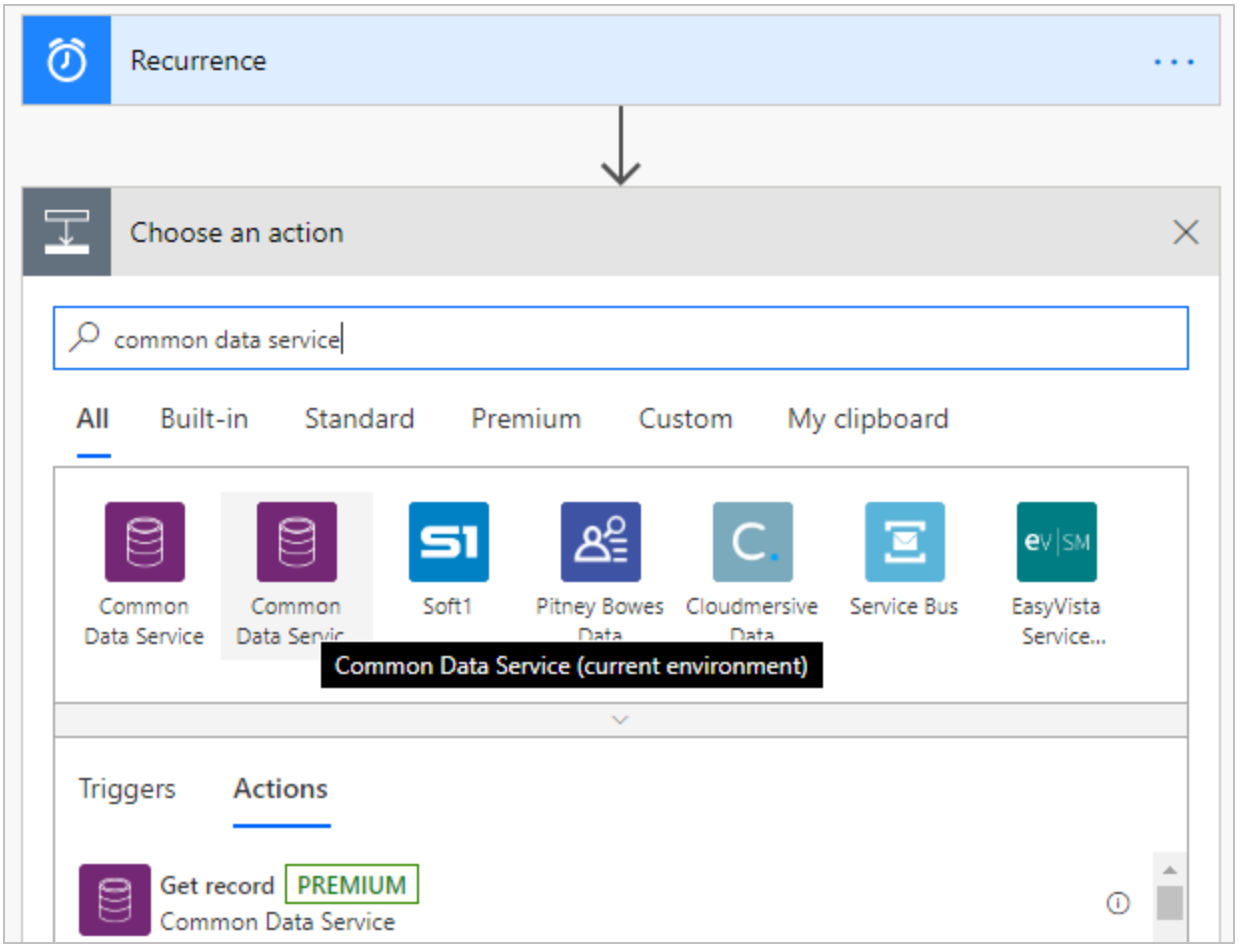
1. Go to https://make.powerapps.com and select the org for which you need to create the connection.
2. Select **My flows** in the right pane and click **New** to create a new flow.



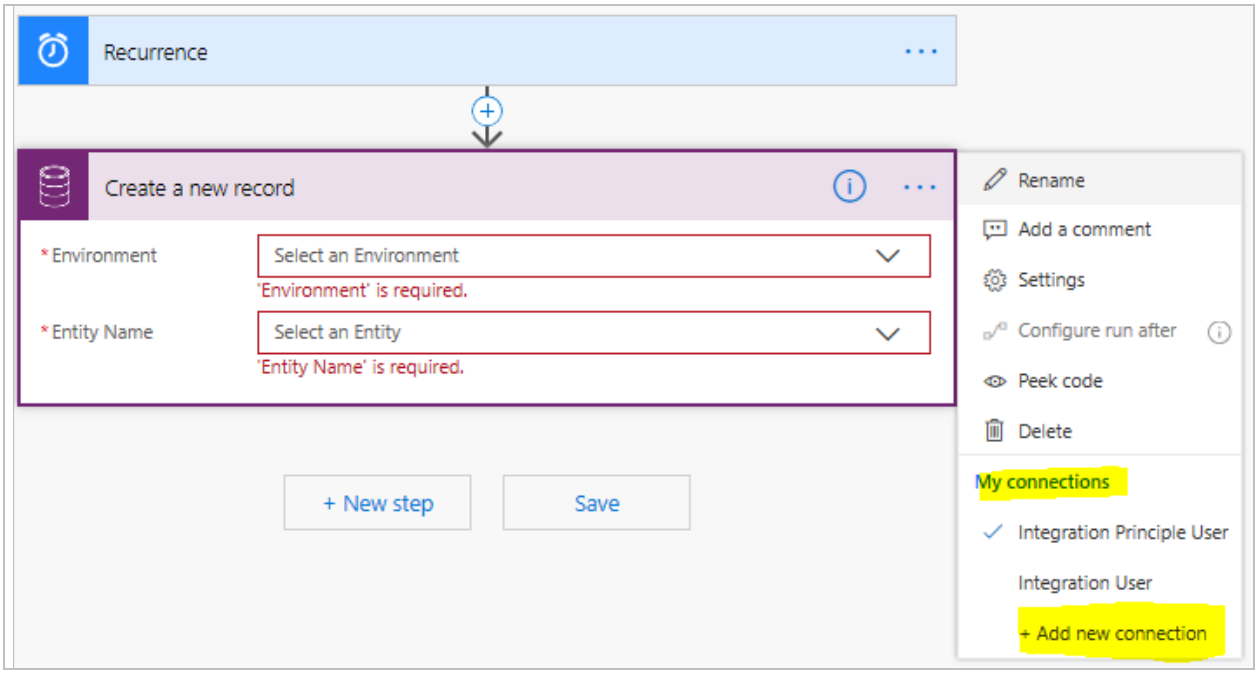

**Note**: The system should have a dedicated application user created for the integration purpose.

3. Select **Schedule** trigger and add a new step to any **CDS (current environment)** action.

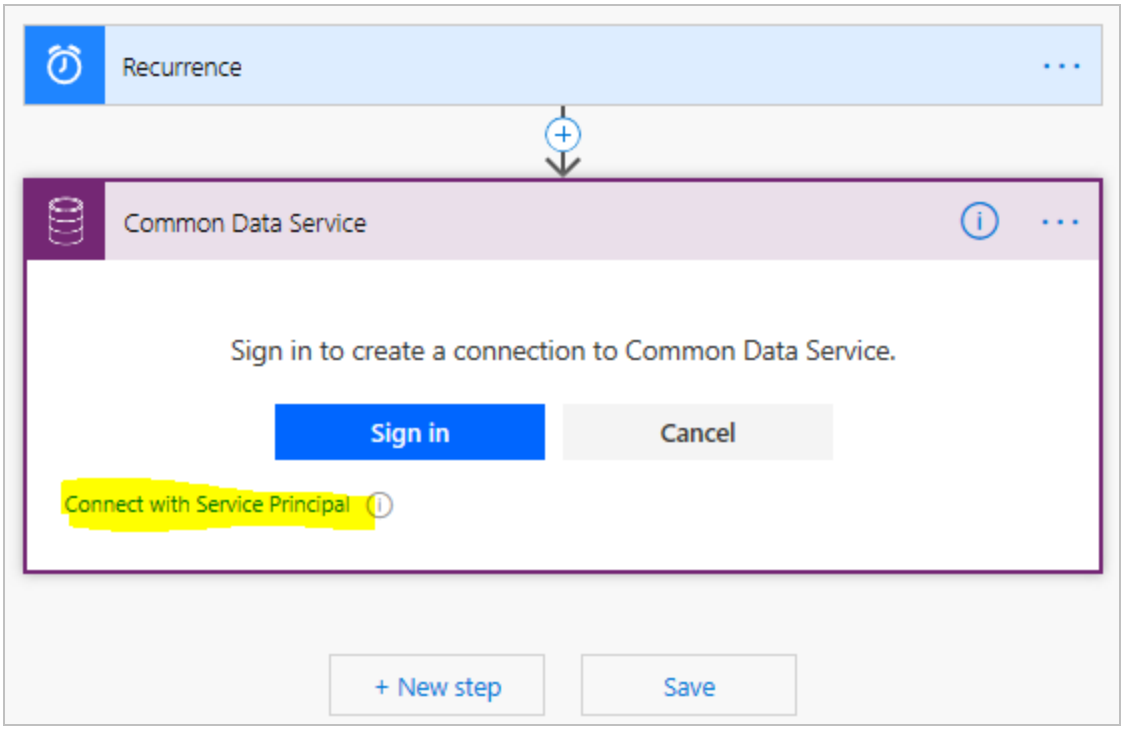
   **Note**: Ensure you are using Common Data Service (current environment). It is usually the second icon. Confirm by hovering on the icon and checking the tooltip as shown below.




- Click the ellipsis button **(...)** in the top right corner of the action step.
- Click the **Add new connection** menu item.

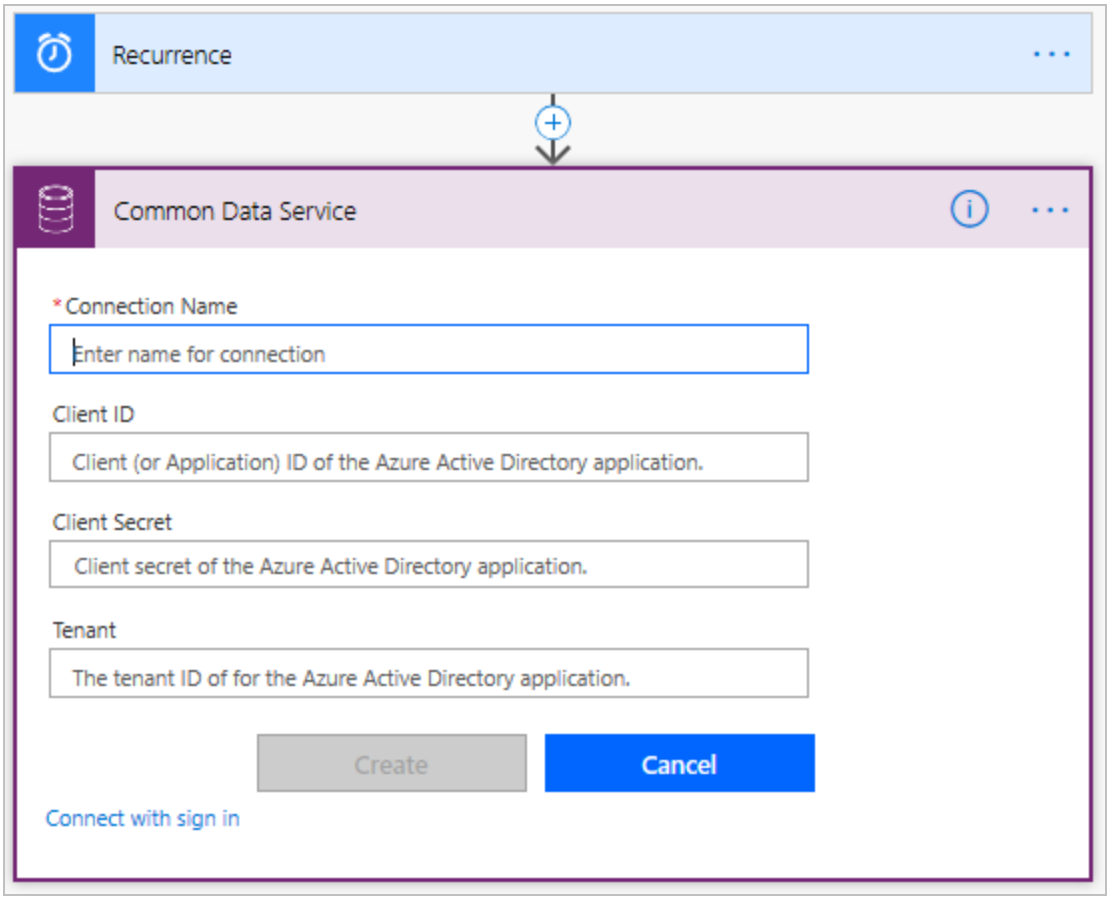



- Choose the **Connect with Service Principal** link in the action step.

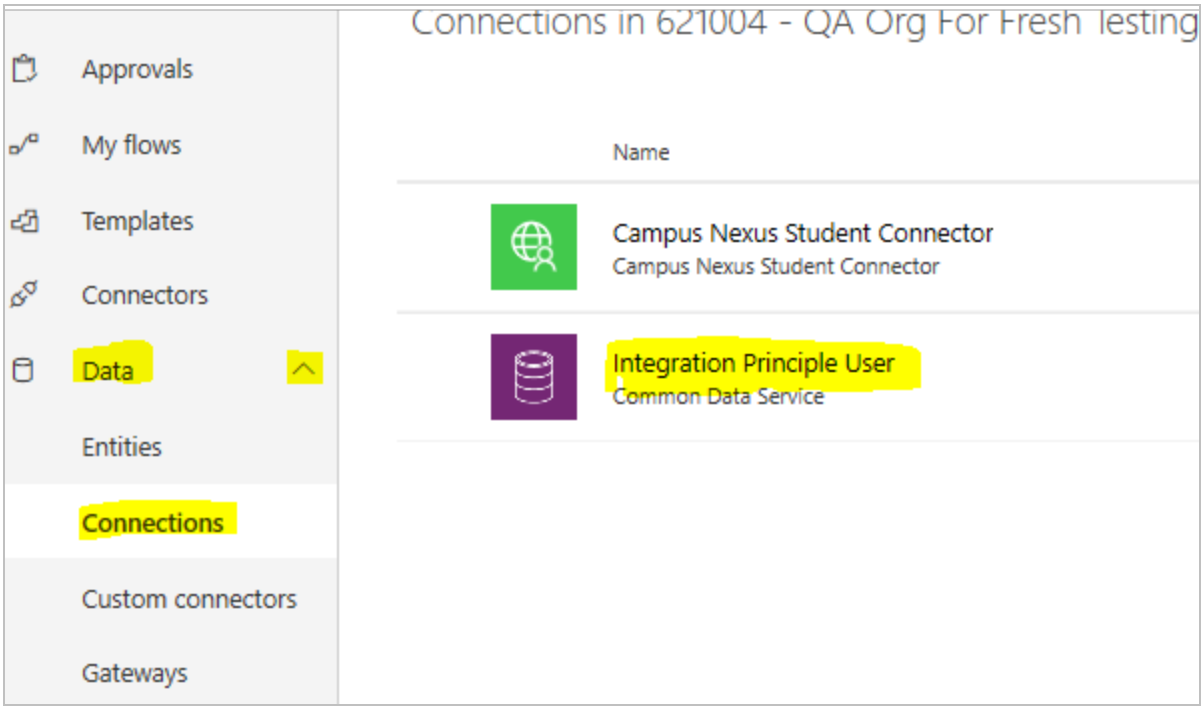



4. Provide necessary details and click **Create**.

   **Note**: You can ignore the flow created. It does not need to be saved since the connection is created in the org by the steps above.




5. After you have created the connection, navigate to **Data > Connection** in the right pane and select the newly created connection.



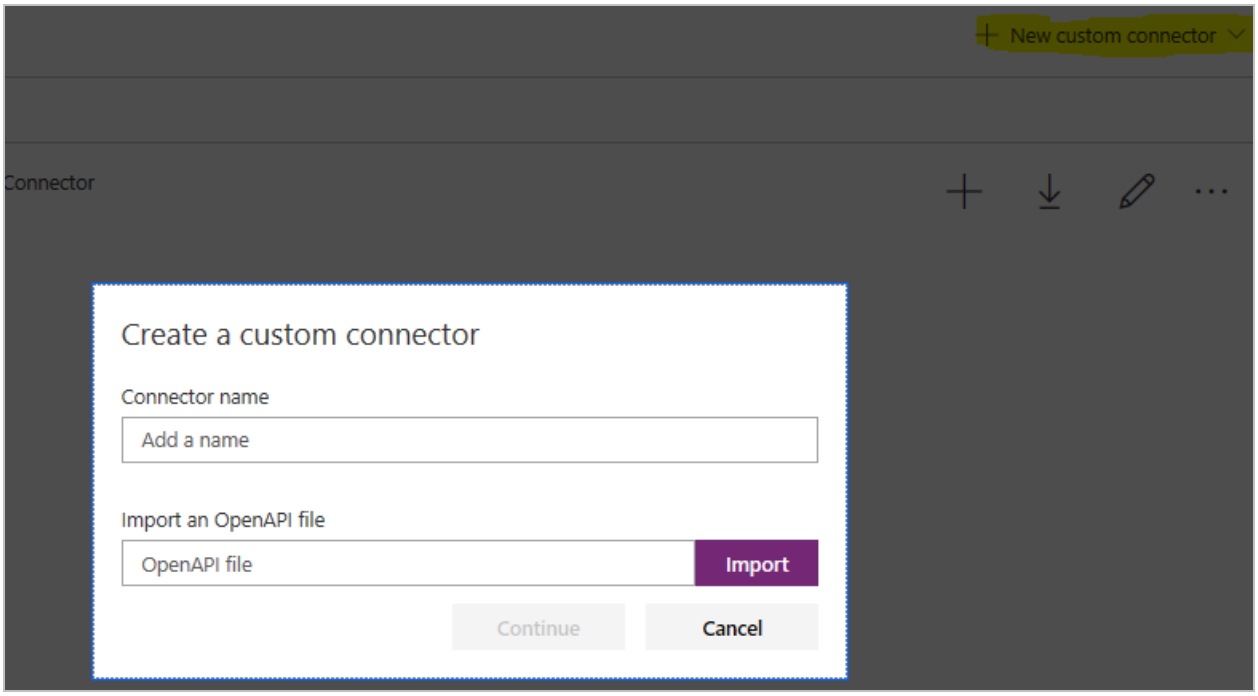

6. In the URL of the browser window, look for the **connectionName** query param, for example https://us.-flow.microsoft.com/manage/environments/fc6b3f3a-b423-4685-bafa-fe22-fafe38bb/connections?apiName=shared_commondataservice&**connectionName=shared-commondataser-1c9c75f6-9fa2-4da5-8190-4f7d7f88ce00**

   The connection name will be used in the pipeline variables.

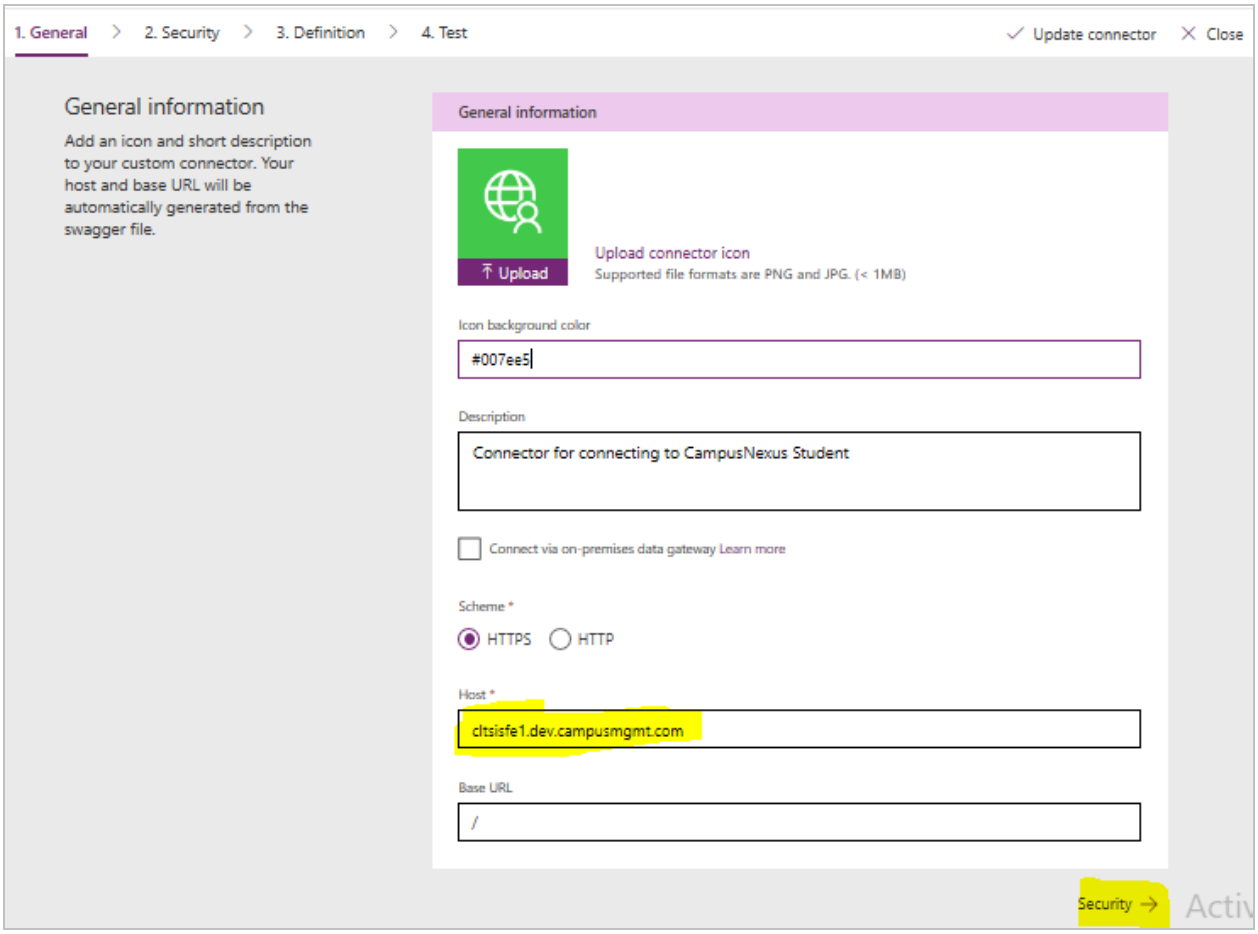
   Only Connection owners can **Turn On** the MS flows.

## Create Anthology Student Custom Connector

1. Go to https://make.powerapps.com and select the org for which you need to create the Custom Connector.
2. Select **Data > Custom connectors** in the right pane and click **New custom Connector > Import an OpenAPI file**.



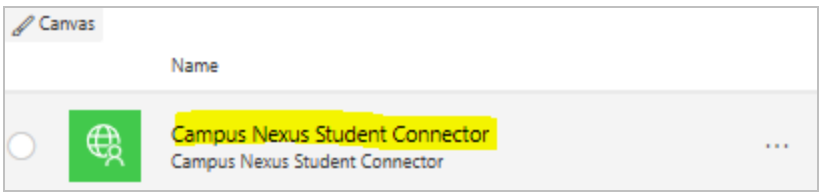

3. Specify the Connector Name as **Campus Nexus Student Connector**.
4. In the "Import an OpenAPI file" field, select the following file from ADO: **$\Engage\Dynamics365Solutions\CampusManagementStudentConnector\CampusNexus-Student-Connector.swagger.json**
5. Update the URL to the Anthology Student URL.




6. Proceed to further tabs and click **Create Connector**.
7. On the "Test" tab, click **New Connection**, provide the API key in the modal pop-up, and **create** the connection.

## Create Anthology Student Custom Connector Connection

1. Go to [https://make.powerapps.com](https://make.powerapps.com) and select the org for which you need to create the Custom Connector's Connection.
2. Navigate to **Data > Connections**.




3. Open the Connector Connection.

4. The Connector Param Name and Connection Name will be available in the URL. These values which will be used in the pipeline variables.

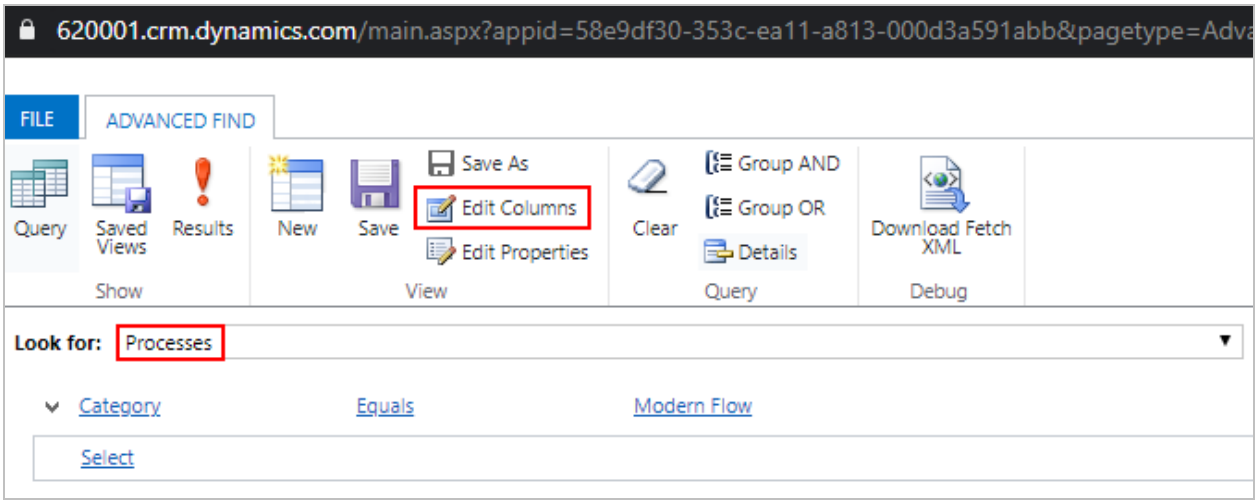
   https://make.powerapps.com/environments/fc6b3f3a-b423-4685-bafa-fe22fafe38bb/connections/shared_campus.20nexus.20student.20connector.5f2540c06b-01532217567572bb/4918ef15625b4dca9a51d952f7022be3/details#

   Where:
   = Connection Param Name
   = Connection Name for the Custom Connector

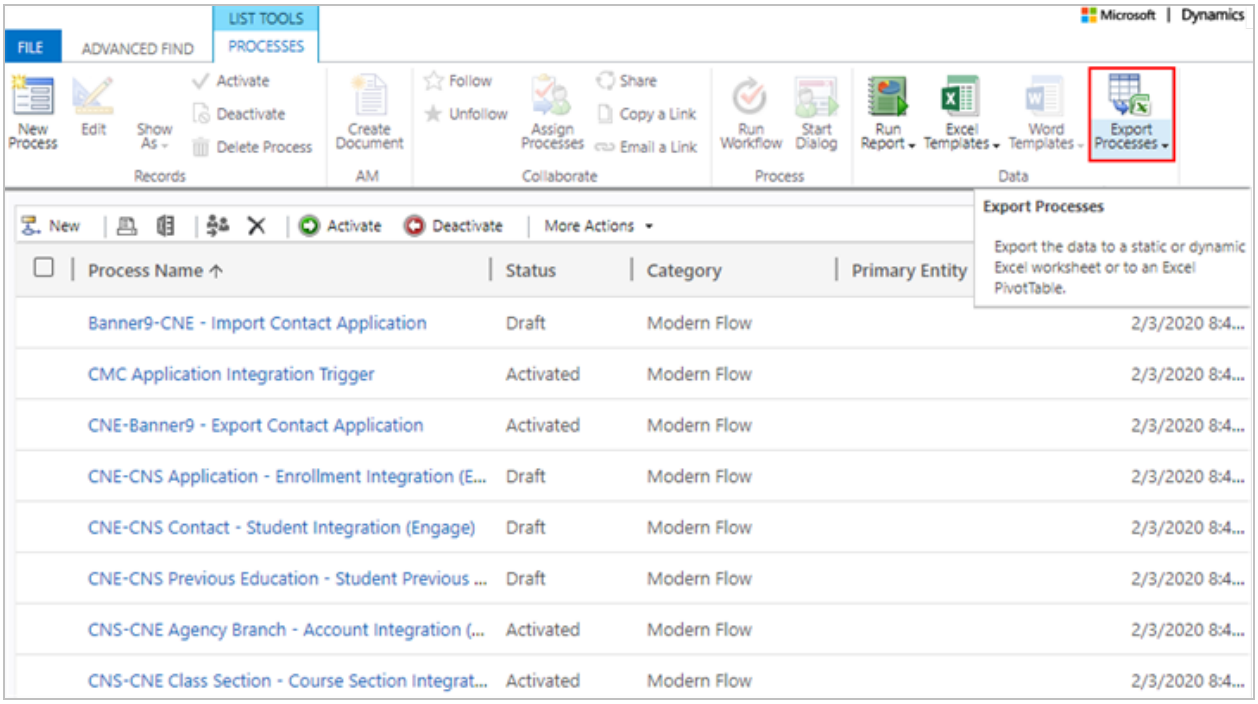
**Export Enabled Flows**

While doing upgrades, the upgraded flows will be set to a disabled state to allow you to identify which flows need to be enabled. You will need to export the flows in the system along with their status.

1. Log in to the **Dynamics org**.
2. Open the **Advanced Find Dialog**.
3. Create a query on the **Processes** entity. Add a query condition for **Category equals 'Modern Flow'**.




4. Ensure that the Status column is selected by clicking the **Edit Columns** button.
5. Click the **Results** button.
6. Click **Export Processes** to export the data.




7. Use the resulting Excel sheet as a reference to identify which flows need to be re-enabled for upgrades.

**Update Pipeline Variables and Run the Release Pipeline**

1. Update the pipeline variables using the:
    - Connections you have created above
    - Azure Function Server URL
2. Execute the release pipeline.

**Run Default Data for Anthology Student Integration**

The default data includes the Option Set mappings that are defined in the Integration Mappings entity. These mappings can later be modified per client implementation requirements.

The default data is available in the following ADO folder:

**$\Dynamics 365 Solutions\CampusManagementDefaultData\CampusManagementStudentIntegration**

**Update Default Integration Mapping Records**

In Integration Mappings, update the following values:

| Name | Value |
|---|---|
| DefaultApplicationSchoolStatus | Application Complete |
| DefaultSourceCategory | Internet |
| DefaultSourceMethod records | Email |

| Name ↑ | Internal Option Display Name |
| --- | --- |
| DefaultApplicationSchoolStatus | Application Complete |
| DefaultSourceCategory | Internet |
| DefaultSourceMethod | Email |

**Configure Integration Settings in the Configuration Record**

Update the following values in the flows:

- Integration Client ID
- Integration Client Secret
- Integration Audience URL
- Integration Default User
- Service Principal User (used by the Selective Update step in the flows)

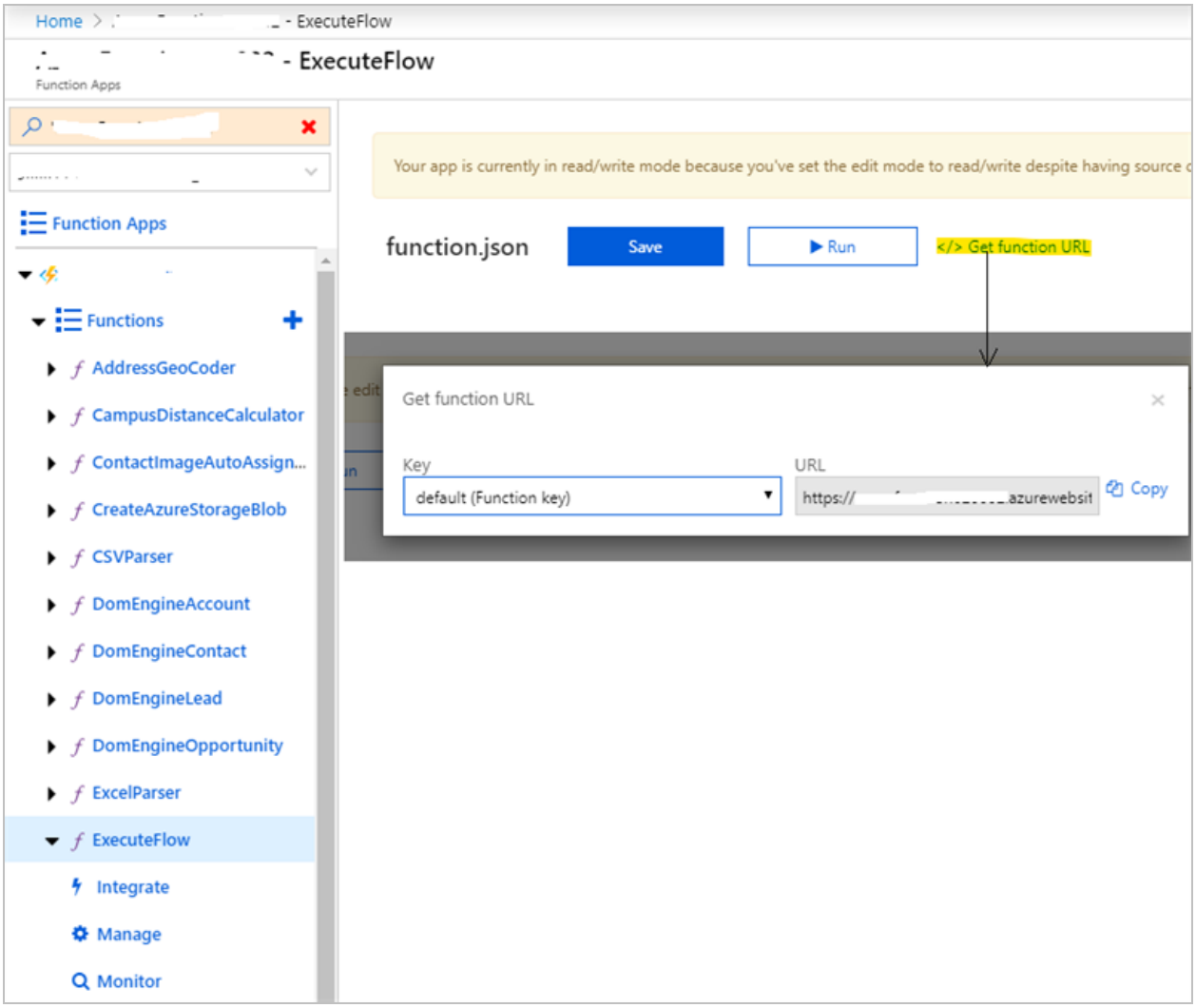
**Notes**:

- Integration Default User and Service Principal User value will be the Application User created for the integration purpose.
- The Integration Audience URL should not end with "/".

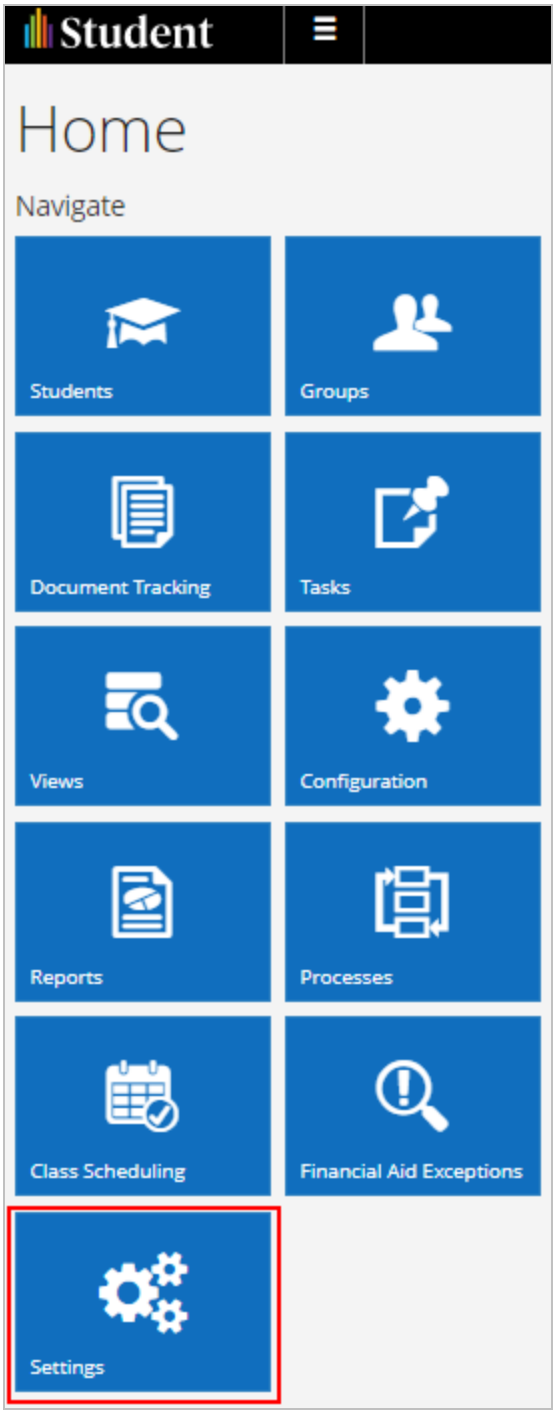
  *Example*: https://621002.crm.dynamics.com and not: https://621002.crm.dynamics.com/

**Configure the CNS-CNE Integration URL**

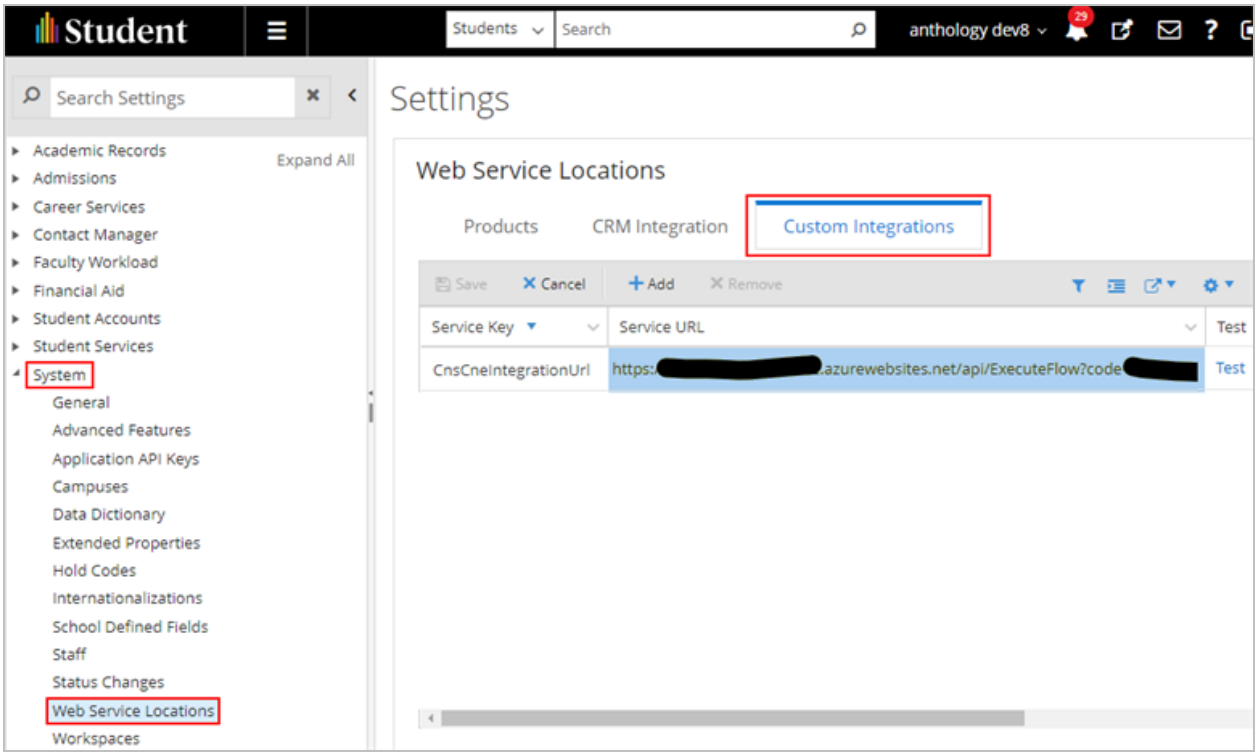
1. Navigate to the **Azure Functions** associated with the Dynamics Org.
2. Select the **ExecuteFlow** function.



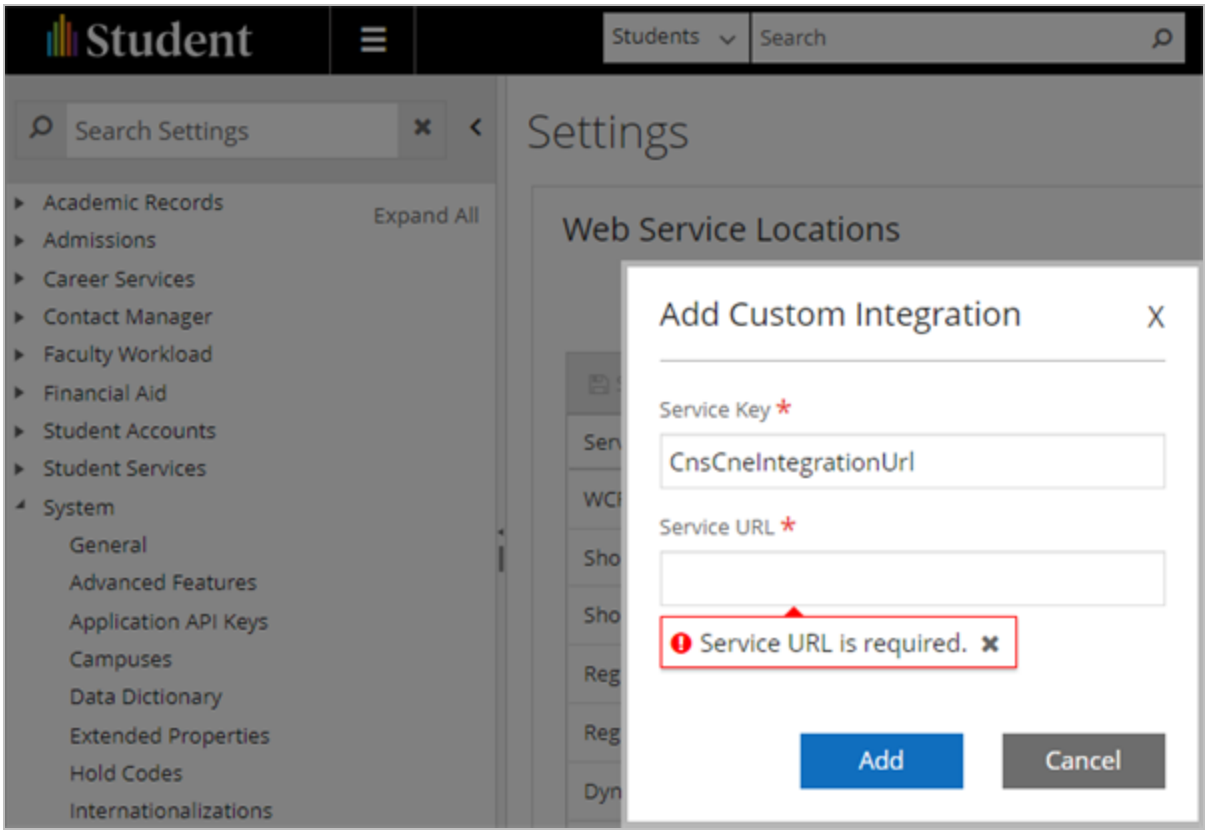

3. In the right pane, click **Get Function URL**, copy the function URL, and save it in a text editor.
4. Log in to the Anthology Student Web App and click the **Settings** tile.




5. Select **System > Web Service Locations** in the left menu.
6. On the **Custom Integrations** tab, search the **Service Key** column for the name **CnsCneIntegrationUrl**.




7. If the key is not present, click **+Add** in the grid toolbar.
    - In the Service Key field, specify **CnsCneIntegrationUrl**.
    - In the Service URL field, specify the **function URL** which has been copied in step 3 of Configure the CNS-CNE Integration URL




8. If the key is already present, modify the URL by editing it within the grid.
9. Click **Save** after modifying the URL.

> ⚠ **Upgrade Considerations**: Ensure that the earlier Service Keys starting with "CnsCne" are removed. Only the key **CnsCneIntegrationUrl** should be present.

**Install Anthology Student Workflows**

Install Workflows in Anthology Student using Workflow Composer.

The Workflows are available in ADO in the following location:

**$\Engage\Dynamics 365 Solutions\CampusManagementStudentWorkflow**

**Enable Power Automate Flows**

For new installations, refer to the Power AutomateDetail Excel file to identify which flows need to be enabled.

For upgrades, use the flows list exported in Export Enabled Flows to enable the flows which had been enabled previously.

For any new flows as part of the upgrade, refer to the Power AutomateDetail Excel file to identify which of the new flows need to be enabled.

## Deploy Service Bus Integration

Service Bus integration of the Anthology Student and Anthology Reach products requires multiple resources as listed below.

**Service Bus Integration Resources**

| Type | Resource Description | Responsibilities |
|---|---|---|
| Common resources | Common resources are deployed irrespective of products when a tenant (client) is provisioned.<br>These resources include:<br>Azure Service Bus (ASB)<br>During the deployment of Anthology Student, the SCM team adds a record to the `Core.ServiceBusNamespace` table with the necessary details of the ASB connectivity. | The SCM team deploys and maintains the common resources. |
| Publisher resources | Publisher resources are the resources required to publish data. These resources do not need to know who consumes the published data. Publisher resources are deployed using the pipeline meant for the product to deploy its productized integration resources.<br>These resources include:<br>ASB topics<br>Azure functions<br>Publisher resources are included in each release of the product/product integration.<br>For Anthology Student changes, when the data is published using business events/technical events, a configuration entry is added to the `Core.IntegrationDisclosure` table to raise business events and to push the data to an Azure Service Bus topic. The entry handles the details of the entity, event, property list, and topic details. | The SCM team deploys the publisher resources.<br>The publishing product teams maintain the publisher resources. |
| Subscriber resources | Subscriber resources are required to subscribe to the data published. These resources are deployed using the pipeline meant for the product to deploy its productized integration resources.<br>These resources include:<br>ASB subscriptions<br>Azure functions<br>Deployment of these resources is included in each release of the product/product integration. | The SCM team deploys the subscriber resources.<br>The subscribing product teams maintain the subscriber resources. |

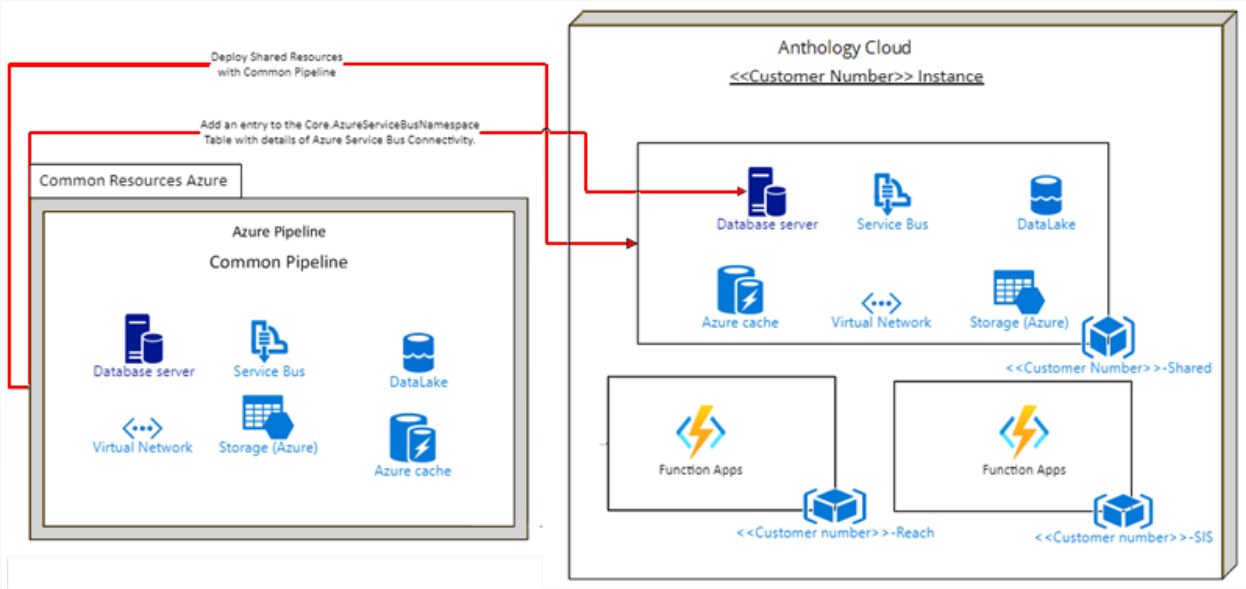
### Common Pipeline

The Common Pipeline deploys the shared resources to the Anthology Cloud.

Shared resources include:

- Database Server
- Service Bus
- Database
- Virtual Network
- Azure Storage
- Azure Cache

The Common pipeline also adds an entry to the `Core.AzureServiceBusNamespace` table with details about the Azure Service Bus connectivity.

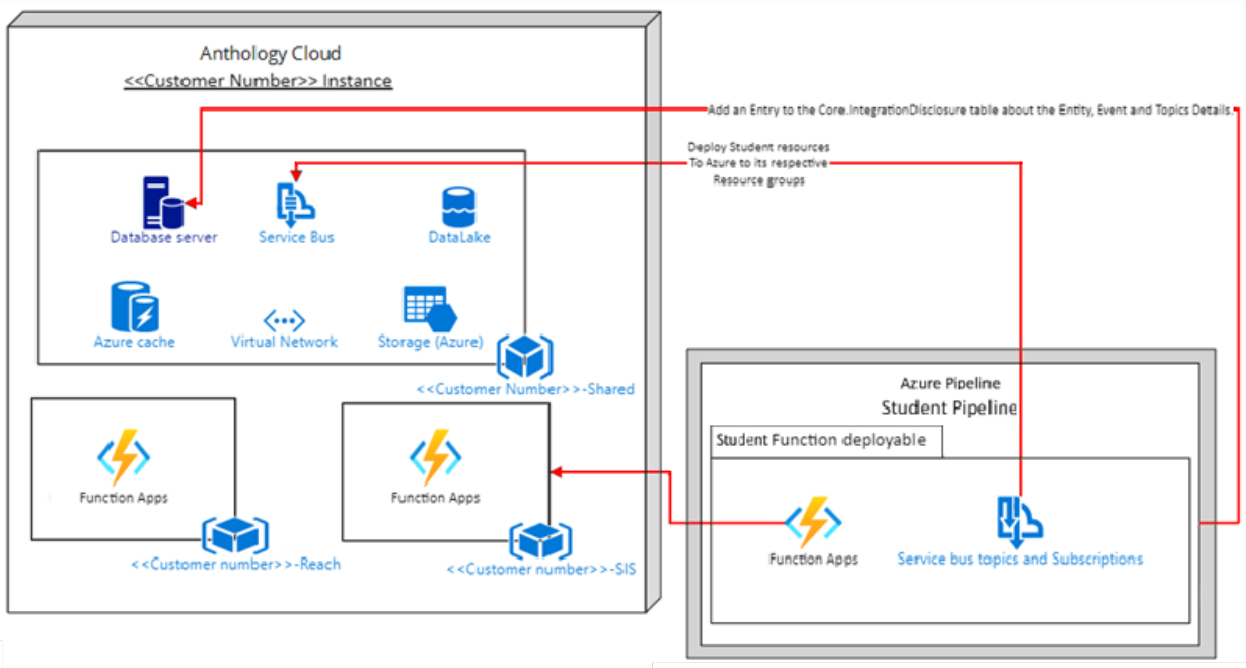



**Student Pipeline**

The Anthology Student pipeline deploys:

- Anthology Student function apps to the Anthology Cloud
- Anthology Student resources to the respective resource groups in the Anthology Cloud

The Student pipeline also adds an entry to the `Core.IntegrationDisclosure` table about the entity, event, and topic details.

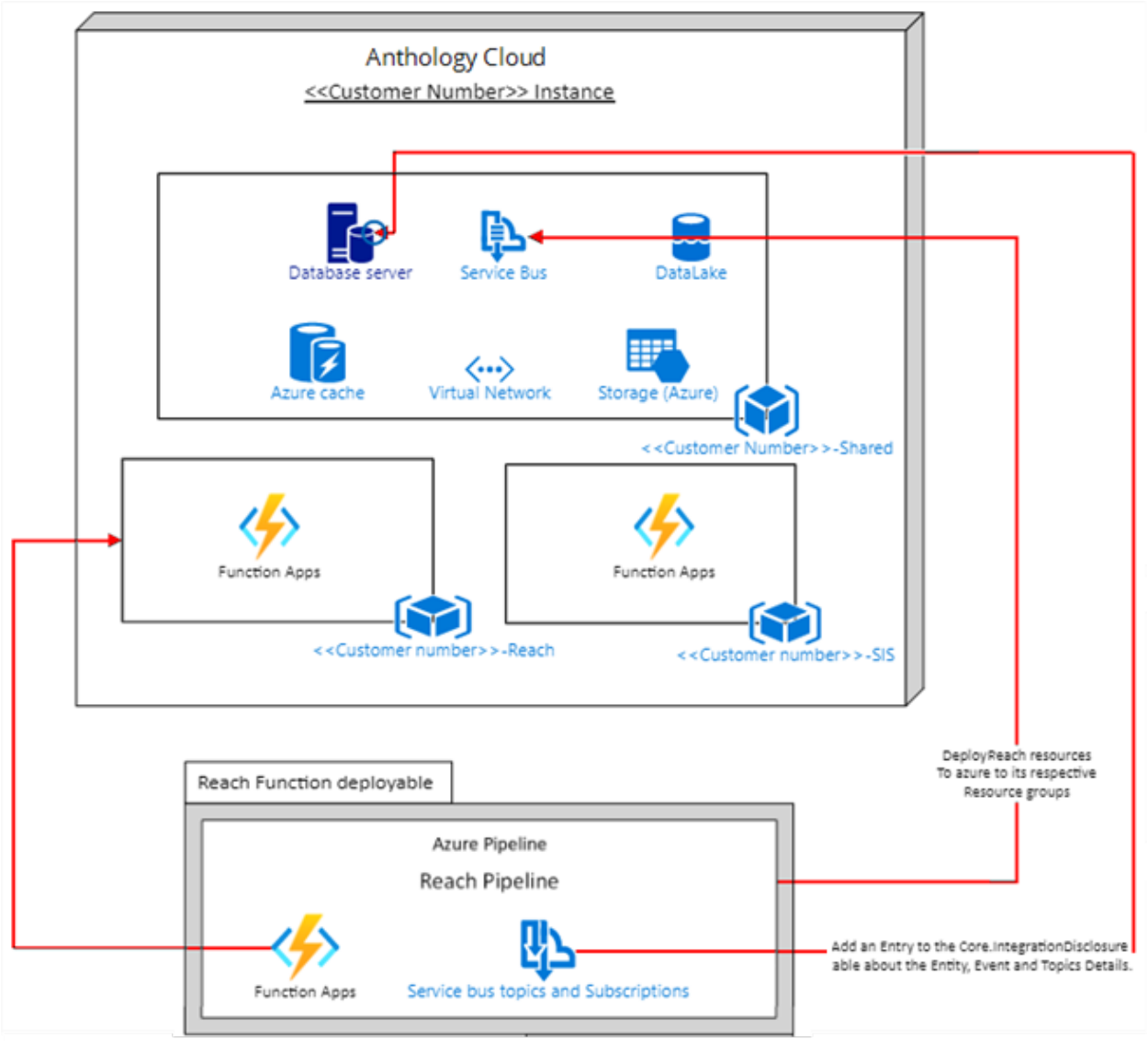



**Reach Pipeline**

The Anthology Reach pipeline deploys:

- Anthology Reach function apps to the Anthology Cloud
- Anthology Reach resources to the respective resource groups in the Anthology Cloud

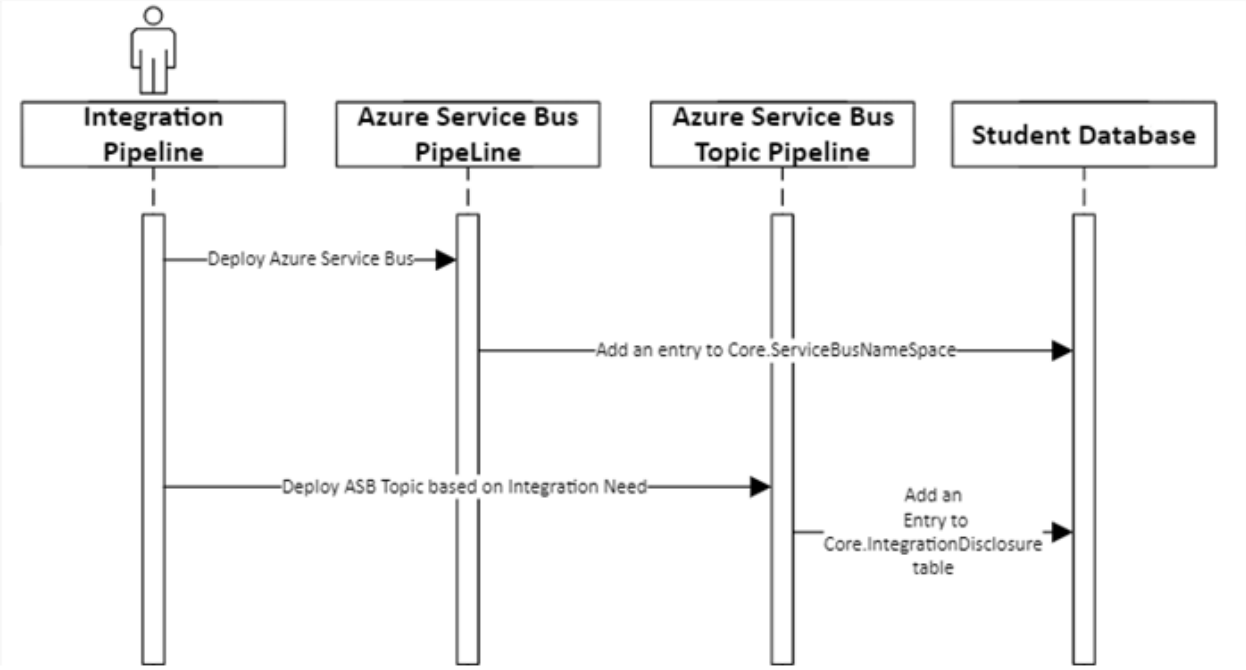
The Reach pipeline also adds an entry to the `Core.IntegrationDisclosure` table about the entity, event, and topic details.




**DB Update Sequence**

The sequence of database updates is as follows:

1. The SCM team deploys the integration pipeline to the Azure Service Bus.
2. The Azure Service Bus pipeline adds an entry to the `Core.ServiceBusNamespace` in the Anthology Student database.
3. The integration pipeline deploys topics to the Azure Service Bus Topic pipeline based on the integration needs.
4. The Azure Service Bus topic pipeline adds an entry to the `Core.IntegrationDisclosure` table in the Anthology Student database.




**Naming Conventions**

1. Azure Service Bus topics and subscriptions:
    a. Topics:

    “anth-<Business/Functionality short form>-<Source product short form>-<Resource short form>-<Resource version number>”

    The short form for a topic is “sbt”.

    *Example*: `“anth-stucorstschg-stu-sbt-01”`

    b. Subscriptions:

    “anth-<Business/Functionality short form>-<Source product shortform>-<Destination short form>-<Resource short form>-<Resource Version number>”

    The short form for a subscription is “sbts”.

    *Example*: `“anth-stucorstschg-stu-rch-sbts-01”`

2. Azure functions:

    Since Azure functions are at the subscription level, we can add the destination to identify it clearly.

    “anth-<Business/Functionality short form>-<Source Product short form>-<Destination short form>-<Resource short form>-<Resource version number>”

    The short form for function is “func”.

    *Example*: `“anth-stucorstschg-stu-rch-func-01”`

3. Business/functionality:

   "anth-<Business/Functionality short form>-<Source product short form>-<Destination short form>-<Resource short form>-<Resource version number>"

   The business/functionality naming is very trick due to the following limitations:

   - Maximum size of any messaging entity path: queue or topic: 260 characters.
   - Maximum size of any messaging entity name: namespace, subscription, or subscription rule: 50 characters.

   Our recommendation is to have three characters from each word of the full meaningful phrase of the business functionality.

   *Example*: “Student Course Status change” would translate to `stucoustschg`.

4. Ownership:

   a. Team owning the business/functionality naming: Product team.

   b. Team owning overall naming of resources associated with products: Product Development.

   c. Team owning the naming of common resources using the conventions: SCM.

**ASB Topic Atomicity**

1. An ASB topic is considered as a data publishing medium in Anthology for all disconnected integrations.
2. Each topic holds the data of a business process, having multiple subscriptions.
3. Atomicity of an ASB topic is the business process.

   For a scenario where...

   a given business process has multiple stages and

   each stage has a business event and

   a sequencing of these stages is required and

   all the data related to the business process be available at a single place and

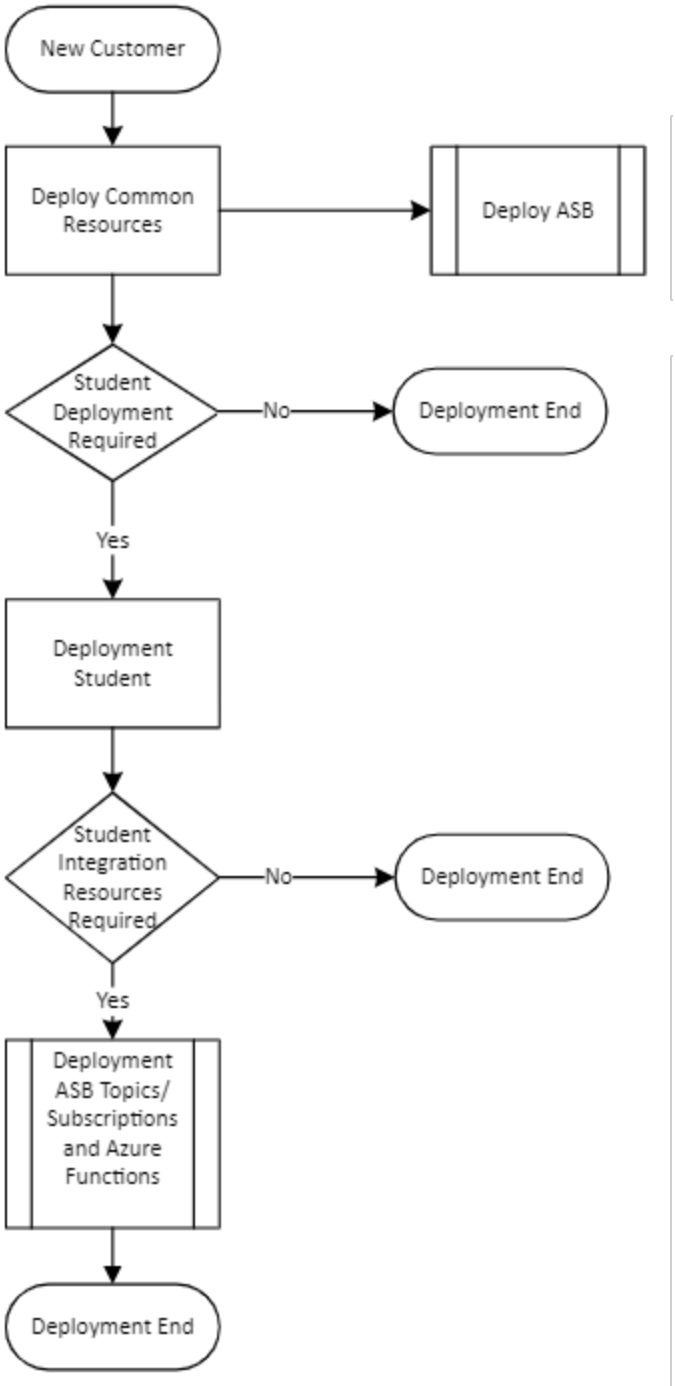
   with the chronological order of the event execution,

   ... our recommendation is to have a single ASB topic for the business process and have all the stages and events publish data to the same topic in chronological order.

   The advantage of the above convention is to allow the subscribers to expect and prepare for the data.

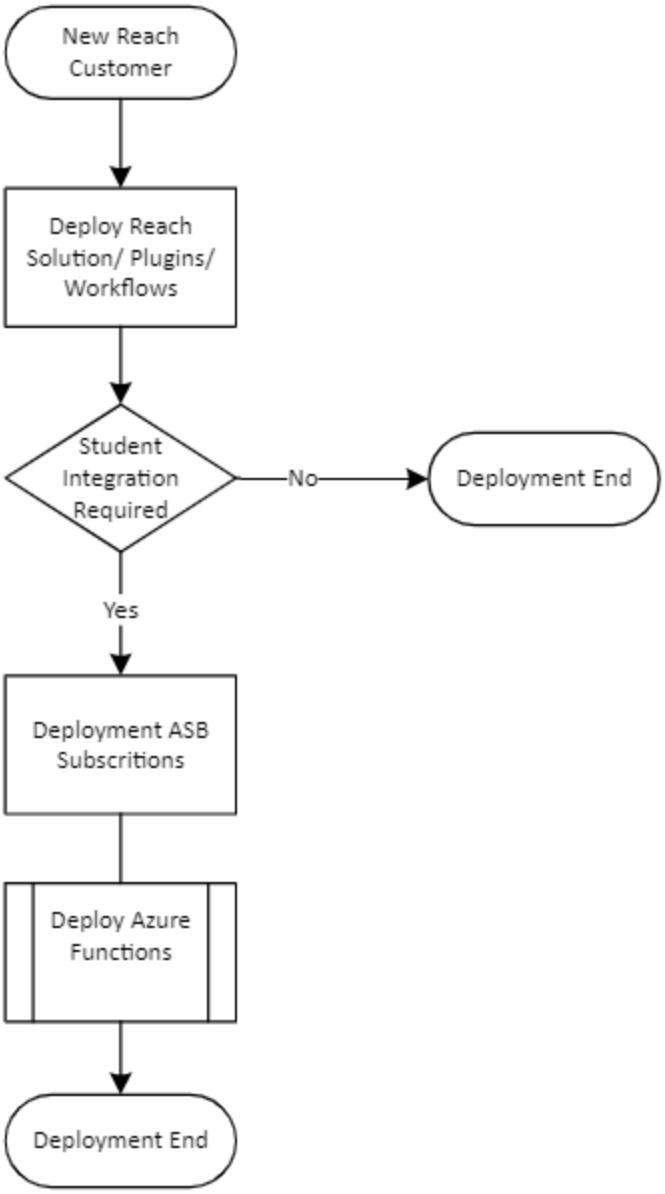
**Anthology Student Deployment Flow**

The following image depicts the deployment flow for Anthology Student clients.




**Anthology Reach Deployment Flow**

The following image depicts the deployment flow for Anthology Reach clients.


New Reach Customer
Deploy Reach Solution/ Plugins/ Workflows
Student Integration Required
No
Deployment End
Yes
Deployment ASB Subscritions
Deploy Azure Functions
Deployment End

# Field Mappings

This section provides reference information on the predefined field mappings for flow-based integrations and Service Bus integrations between Anthology Reach and Anthology Student. The mappings listed here are the out-of-the-box mappings based on a fixed data model.

Clients can modify the field mappings by adding custom fields and options (see From Anthology Student to Anthology Reach).

Clients also have the option to create applications and functions that extend the integrations as appropriate for their environments. For details about extension options, see Extensibility.

## Field Mappings for Flow-based Integrations

This page presents the entity field mappings between Anthology Student and Anthology Reach categorized by the data flow directions:

- Uni-directionally from Anthology Student to Anthology Reach
- Uni-directionally from Anthology Reach to Anthology Student
- Bi-directionally between Anthology Reach and Anthology Student

The headings below indicate the entity names in Anthology Student and Anthology Reach. You can also view this data in the attached Excel file: Field Mappings for Anthology Student and Anthology Reach

These mappings reflect the predefined (out-of-the-box) integration of reference and operational data (see Integrated Entities).

Administrators can download the StudentIntegration_OptionSetMapping.csv file and use it to load the predefined mapping templates and option sets into their Anthology Reach environment. For procedure details, see Import Integration Mapping Templates.

### Data Sent from Anthology Student to Anthology Reach

### Agency Branch > Account (Operational Data)

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| Agency-Branch | City | AmAgency-Branch | City | String | Accoount | account | Address 1: City | address1_city | String | Account Type = "-Other" |
| Agency-Branch | Country | AmAgency-Branch | SyCountryID | Optionset | Accoount | account | Address 1: Country/ Region | address1_country | String | Account Type = "-Other" |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| Agency-Branch | Address | AmAgency-Branch | Addr1 | String | Accoount | account | Address 1: Street 1 | address1_line1 | String | Account Type = "-Other" |
| Agency-Branch | Postal Code | AmAgency-Branch | Zip | String | Accoount | account | Address 1: ZIP/ Postal Code | address1_postalcode | String | Account Type = "-Other" |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| Agency-Branch | State | AmAgency-Branch | State | Optionset | Accoount | account | Address 1: State/ Province | address1_stateorprovince | String | Account Type = "-Other" |
| Agency-Branch | Email Address | AmAgency-Branch | EMail | String | Accoount | account | Email | emailaddress1 | String | Account Type = "-Other" |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| Agency-Branch | N/A | AmAgency-Branch | N/A | N/A | Account | account | Account Type | mshied_accounttype | Optionset | Account Type = "-Other" |
| Agency-Branch | Id | AmAgency-Branch | amAgencyID | String | Account | account | External Identifier | mshied_externalidentifier | String | Account Type = "-Other" |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| Agency-Branch | N/A | AmAgency-Branch | N/A | Optionset | Account | account | External Source System | mshied_external source system | Optionset | Account Type = "-Other" |
| Agency-Branch | Name | AmAgency-Branch | Descrip | String | Account | account | Name | name | string | Account Type = "-Other" |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| Agency-Branch | Phone Number | AmAgency-Branch | Phone | String | Accoount | account | Main Phone | telephone1 | String | Account Type = "-Other" |
| Agency-Branch | Statement Commement | AmAgency-Branch | Stmt Commment | String | Accoount | account | Description | description | String | Account Type = "-Other" |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| AgencyBranch | DateAdded | AmAgencyBranch | DateAdded | DateTime | Accoount | accoount | RecordCreatedOn | overriddencreatedon | DateTime | AccountType = "-Other" |

**Class Section > Course Section (Reference Data)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| ClassSection | Id | AdClassSched | AdClassSchedId | String | CourseSection | mshied_coursesection | External Identifier | cmc_externalidentifier | String |
| ClassSection | SectionName | AdClassSched | Descrip | String | CourseSection | mshied_coursesection | Course Section Name | mshied_name | String |
| ClassSection | InstructorId | AdClassSched | AdTeacherID | Lookup | CourseSection | mshied_coursesection | Instructor | cmc_staffid | Lookup |
| ClassSection | CourseId | AdClassSched | AdCourseID | Lookup | CourseSection | mshied_coursesection | Course | mshied_courseid | Lookup |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| ClassSection | CampusId | AdClassSched | SyCampusId | Lookup | CourseSection | mshied_coursesection | Campus | mshied_campusid | Lookup |
| ClassSection | Terms | AdClassSched Term | AdTermId | Lookup | CourseSection | mshied_coursesection | Academic Period | mshied_academicperiodid | Lookup |
| ClassSection | N/A | AdClassSched | N/A | | CourseSection | mshied_coursesection | External Source System | cmc_externalsourcesystem | Optionset |

**College > Account (Operational Data)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| College | City | AmCollege | city | String | Account | account | Address 1: City | address1_city | String | (Account Type = College) |
| College | Country | AmCollege | SyCountryID | Optionset | Account | account | Address 1: Country/ Region | address1_country | String | (Account Type = College) |
| College | Address | AmCollege | Addr1 | String | Account | account | Address 1: Street 1 | address1_line1 | String | (Account Type = College) |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| College | Postal Code | AmCollege | Zip | String | Account | account | Address 1: ZIP/Postal Code | address1_postalcode | String | (Account Type = College) |
| College | Code | AmCollege | Code | String | Account | account | Code | cmc_accountcode | String | (Account Type = College) |
| College | Email Address | AmCollege | email | String | Account | account | Email | email address1 | String | (Account Type = College) |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| College | N/A | AmCollege | N/A | | Account | account | Account Type | mshied_accounttype | Optionset | (Account Type = College) |
| College | Id | AmCollege | AmCollegeID | String | Account | account | External Identifier | mshied_externalidentifier | String | (Account Type = College) |
| College | N/A | AmCollege | N/A | | Account | account | External Source System | mshied_externalsourcesystem | Optionset | (Account Type = College) |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| College | Name | AmCollege | Descrip | String | Account | account | Name | name | string | (Account Type = College) |
| College | Phone Number | AmCollege | Phone | String | Account | account | Main Phone | telephone1 | String | (Account Type = College) |
| College | Date Added | AmCollege | DateAdded | DateTime | Account | account | Record Created On | overriddencreatedon | DateTime | (Account Type = College) |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| College | TranscriptFee | AmCollege | TranscriptFee | Integer | Account | account | Fee | cmc_transcriptofficefee | Decimal | (Account Type = College) |
| College | Note | AmCollege | Comments | String | Account | account | Description | description | String | (Account Type = College) |
| College | State | AmCollege | State | Optinset | Account | account | Address 1: State/Province | address1_stateorprovince | String | (Account Type = College) |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| College | Fax Number | AmCollege | Fax | String | Account | account | Fax | fax | String | (Account Type = College) |

**Concentration > Area of Study (Reference Data)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Concentration | Id | AdConcentration | AdConcentrationId | String | AreaofStudy | mshied_areaofstudy | ExternalIdentifier | mshied_externalidentifier | String |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Concentration | Code | AdConcentration | Code | String | AreaofStudy | mshied_areaofstudy | Code | mshied_code | String |
| Concentration | Name | AdConcentration | Descrip | String | AreaofStudy | mshied_areaofstudy | Area of Study Name | mshied_name | String |
| Concentration | AreaOfStudy TypeId | AdConcentration | AdConcentration TypeId | Lookup | AreaofStudy | mshied_areaofstudy | Program Type | mshied_programtypeid | Lookup |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Concentration | MinimumGpaRequired | AdConcentration | MinGPA | Number | AreaofStudy | mshied_areaofstudy | MinimumRequiredGPA | mshied_minimumrequiredgpa | Number |
| Concentration | CreditHoursRequired | AdConcentration | CreditsReq | Number | AreaofStudy | mshied_areaofstudy | RequiredCredits | mshied_requiredcredits | Number |
| Concentration | N/A | AdConcentration | N/A | | AreaofStudy | mshied_areaofstudy | ExternalSourceSystem | mshied_externalsourcesystem | Optionset |

**Course > Course (Reference Data)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Course | Id | AdCourse | AdCourseID | String | Course | mshied_course | External Identifier | mshied_externalidentifier | String |
| Course | Name | AdCourse | Descrip | String | Course | mshied_course | Course Name | mshied_name | String |
| Course | Catalog Code | AdCourse | CatalogCode | String | Course | mshied_course | Subject | mshied_subject | String |
| Course | Code | AdCourse | Code | String | Course | mshied_course | Course Number | mshied_coursenumber | String |
| Course | Course Level | AdCourse | AdCourseLevelID | Option-set | Course | mshied_course | Academic Level | mshied_academicleve-l | Option-set |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Course | N/A | AdCourse | N/A | | Course | mshied_course | External Source System | mshied_externalsourcesystem | Optionset |

**Degree > Program Level (Reference Data)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Degree | Id | AdDegree | AdDegreeID | String | Program Level | mshied_programlevel | External Identifier | mshied_externalidentifier | String |
| Degree | Name | AdDegree | Descrip | String | Program Level | mshied_programlevel | Program Level Name | mshied_name | String |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Degree | Code | AdDegree | Code | String | Program Level | mshied_programlevel | Code | mshied_code | String |
| Degree | N/A | AdDegree | N/A | | Program Level | mshied_programlevel | External Source System | mshied_externalsourcesystem | Optionset |

**Document > External Document Status (Special Handling)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Document | Id | CmDocument | CmDocumentID | int | External Document Status | cmc_externaldocumentstatus | External Identifier | cmc_externalidentifier | String |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Document | Student. FullName | cmdocument | SyStudentID (this is a primary key of the table systudent) | int | External Document Status | cmc_externaldocumentstatus | Name | cmc_name | String |
| Document | Student. PersonId | cmdocument | SyStudentID | int | External Document Status | cmc_externaldocumentstatus | Contact | cmc_contactid | Lookup |
| Document | Student Enrollment PeriodId | cmdocument | Adenrollid | int | External Document Status | cmc_externaldocumentstatus | Enrollment | cmc_enrollmentid | Lookup |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Document | Requested Date | CmDocument | DateReq | DateTime | External Document Status | cmc_external document status | Date Requested | cmc_date requested | DateTime |
| Document | Expires Date | CmDocument | DateExpires | DateTime | External Document Status | cmc_external document status | Date Expires | cmc_date expires | DateTime |
| Document | Approved Date | CmDocument | DateApproved | DateTime | External Document Status | cmc_external document status | Date Approved | cmc_date approved | DateTime |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Document | DueDate | CmDocument | DateDue | DateTime | External Document Status | cmc_external document status | Due Date | cmc_duedate | DateTime |
| Document | ReceivedDate | CmDocument | DateRecv | DateTime | External Document Status | cmc_external document status | Date Received | cmc_date received | DateTime |
| Document | SentDate | CmDocument | DateSent | DateTime | External Document Status | cmc_external document status | Date Sent | cmc_datesent | DateTime |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Document | Module. Name | CmDocument | SyModuleID | int | External Document Status | cmc_external document status | Module | cmc_module | Optionset |
| Document | Document Status. Name | CmDocument | CmDocStatusID ( this is a primary key of the table CmDocStatus) | int | External Document Status | cmc_external document status | Documentstat | cmc_document status | String |
| Document | Document Type. Name | CmDocument | CmDocTypeID ( this is a primary key of the table CmDocType) | int | External Document Status | cmc_external document status | Documentty | cmc_document type | String |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Document | NA | NA | NA | NA | External Document Status | cmc_external document status | External Source System | cmc_external source system | Optionset |

**Enrollment > Enrollment (Reference Data)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Enrollment | Enrollment Number | AdEnroll | StuNum | String | Enrollment | cmc_enrollment | Enrollment Number | cmc_external enrollment number | String |
| Enrollment | Id | AdEnroll | AdEnrollID | String | Enrollment | cmc_enrollment | External Identifier | cmc_external identifier | String |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Enrollment | Enrollment Date | AdEnroll | EnrollDate | DateTime | Enrollment | cmc_enrollment | Enrollment Date | cmc_enrollment date | DateTime |
| Enrollment | Expected Start Date | AdEnroll | ExpStartDate | DateTime | Enrollment | cmc_enrollment | Start Date | cmc_startdate | DateTime |
| Enrollment | Last Date Attended | AdEnroll | LDA | DateTime | Enrollment | cmc_enrollment | Last Day of Attendance | cmc_lastdayof attendance | DateTime |
| Enrollment | Graduation Date | AdEnroll | GradDate | DateTime | Enrollment | cmc_enrollment | Graduation Date | cmc_graduation date | DateTime |
| Enrollment | Enrollment Status | AdEnroll | | Optionset | Enrollment | cmc_enrollment | Enrollment Status | cmc_enrollment status | Optionset |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Enrollment | Campus | AdEnroll | SyCampusID | Lookup | Enrollment | cmc_enrollment | Campus | cmc_campusid | Lookup |
| Enrollment | Program | AdEnroll | AdProgramID | Lookup | Enrollment | cmc_enrollment | Program | cmc_programid | Lookup |
| Enrollment | Program Version | AdEnroll | adProgramVersionID | Lookup | Enrollment | cmc_enrollment | Program Version Name | cmc_programversionid | Lookup |
| Enrollment | Shift | AdEnroll | adShiftID | Lookup | Enrollment | cmc_enrollment | Shift | cmc_shiftid | Lookup |
| Enrollment | Start Term | AdEnroll | adTermID | Lookup | Enrollment | cmc_enrollment | Academic Period | cmc_academicperiodid | Lookup |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Enrollment | Clock Hours Required | AdEnroll | HoursReq | Decimal | Enrollment | cmc_enrollment | Program Hours Required | cmc_programhoursrequired | Decimal |
| Enrollment | Credit Hours Required | AdEnroll | CreditsReq | Decimal | Enrollment | cmc_enrollment | Program Credits Required | cmc_programcreditsrequired | Decimal |
| Enrollment | Clock Hours Scheduled | AdEnroll | HoursSched | Decimal | Enrollment | cmc_enrollment | Program Hours Scheduled | cmc_programhoursscheduled | Decimal |
| Enrollment | Credit Hours Scheduled | AdEnroll | CreditsSched | Decimal | Enrollment | cmc_enrollment | Program Credits Scheduled | cmc_programcreditsscheduled | Decimal |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Enrollment | Transfer Credit Hours | AdEnroll | TransferCredits | Decimal | Enrollment | cmc_enrollment | Program Transfer Hours | cmc_programtransferhours | Decimal |
| Enrollment | Expected Credits Hours Per Term | AdEnroll | ExpectedCreditsPerTerm | Decimal | Enrollment | cmc_enrollment | Expected Credits per Term | cmc_expectedcreditsperterm | Decimal |
| Enrollment | Gpa | AdEnroll | GPA | Decimal | Enrollment | cmc_enrollment | Program Cumulative GPA | cmc_programcumulativegpa | Decimal |
| Enrollment | Credit Hours Earned | AdEnroll | CreditsEarned | Decimal | Enrollment | cmc_enrollment | Credits Earned | cmc_creditsearned | Decimal |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Enrollment | Credit Hours Attempted | AdEnroll | CreditsAttempt | Decimal | Enrollment | cmc_enrollment | Credits Attempted | cmc_creditsattempted | Decimal |
| Enrollment | Clock Hours Earned | AdEnroll | HoursEarned | Decimal | Enrollment | cmc_enrollment | Hours Earned | cmc_hoursearned | Decimal |
| Enrollment | Clock Hours Attempted | AdEnroll | HoursAttempt | Decimal | Enrollment | cmc_enrollment | Hours Attempted | cmc_hoursattempted | Decimal |
| Enrollment | Id | AdEnroll | | Lookup | Enrollment | cmc_enrollment | Lifecycle Name | cmc_opportunityid | Lookup |
| Enrollment | School Status | AdEnroll | SyScholStatusID | Lookup | Enrollment | cmc_enrollment | Student Status | cmc_studentstatusid | Lookup |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Enrollment | N/A | AdEnroll | N/A | Optionset | Enrollment | cmc_enrollment | External Source System | cmc_externalsourcesystem | Optionset |

**Extracurricular Activities > Extra Curricular Activity (Operational Data)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Extracurricular Activities | Id | AmExtraCurr | AmExtraCurrID | String | ExtraCurricular Activity | mshied_extracurricular activities | External Identifier | mshied_externalidentifier | String |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Extracurricular Activities | Name | AmExtraCurr | Descrip | String | Extra Curricular Activity | mshied_extracurricular activities | Extra Curricular Activity Name | mshied_name | String |
| Extracurricular Activities | Code | AmExtraCurr | Code | String | Extra Curricular Activity | mshied_extracurricular activities | Code | mshied_code | String |
| Extracurricular Activities | N/A | AmExtraCurr | N/A | Optionset | Extra Curricular Activity | mshied_extracurricular activities | External Source System | mshied_externalsourcesystem | Optionset |

**High School > Account (Operational Data)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| HighSchool | City | AmHighSchool | city | String | Account | account | Address 1: City | address1_city | String | (AccountType = HighSchool) |
| HighSchool | Country | AmHighSchool | Country | String | Account | account | Address 1: Country/Region | address1_country | String | (AccountType = HighSchool) |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| HighSchool | ZIP Code/ Postal Code | AmHighSchool | Zip | String | Account | account | Address 1: ZIP/ Postal Code | address1_postalcode | String | (Account Type = High School) |
| HighSchool | State | AmHighSchool | State | Optionset | Account | account | Address 1: State/ Province | address1_stateorprovince | String | (Account Type = High School) |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| HighSchool | High School Code | AmHighSchool | Code | String | Account | account | Code | cmc_accountcode | String | (AccountType = HighSchool) |
| HighSchool | Transcript Fee | AmHighSchool | TranscriptFee | Integer | Account | account | Fee | cmc_transcript office fee | Decimal | (AccountType = HighSchool) |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| HighSchool | Email Address | AmHighSchool | Email | String | Account | account | Email | email address1 | String | (Account Type = High School) |
| HighSchool | Transcript Office Fax Number | AmHighSchool | FaxNo | String | Account | account | Fax | fax | String | (Account Type = High School) |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| HighSchool | N/A | AmHighSchool | N/A | | Account | account | Account Type | mshied_accounttype | Optionset | (Account Type = High School) |
| HighSchool | High School Name | AmHighSchool | Descrip | String | Account | account | Name | name | String | (Account Type = High School) |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| HighSchool | PhoneNumber | AmHighSchool | Phone | String | Account | account | Main Phone | telephone1 | String | (AccountType = HighSchool) |
| HighSchool | URL | AmHighSchool | URL | String | Account | account | Website | websiteurl | String | (AccountType = HighSchool) |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| HighSchool | Id | AmHighSchool | AmHighSchoolId | String | Account | account | External Identifier | mshied_externalidentifier | String | (AccountType = HighSchool) |
| HighSchool | N/A | AmHighSchool | N/A | | Account | account | External Source System | mshied_externalsourcesystem | Optionset | (AccountType = HighSchool) |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| HighSchool | AcademicYearEnd | AmHighSchool | AcadYearEnd | Optionset | Account | account | Academic/ Fiscal Year End | cmc_academicorfiscalyearend | String | (AccountType = HighSchool) |
| HighSchool | Notes | AmHighSchool | Comments | String | Account | account | Description | description | String | (AccountType = HighSchool) |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| HighSchool | Address | AmHighSchool | Addr-1 | String | Account | account | Address 1: Street 1 | address-1_line1 | String | (AccountType = HighSchool) |
| HighSchool | Date Added | AmHighSchool | DateAdded | DateTime | Account | account | Record Created On | overriddencreatedon | DateTime | (AccountType = HighSchool) |

**HoldCode > cmc_hold (Special Handling)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| HoldCode | HoldCode | SyHoldCode | HoldCode | NA | cmc_hold | cmc_code | Code | cmc_code | String | |
| HoldCode | HoldCode | SyHoldCode | HoldCode | NA | cmc_hold | cmc_externalidentifier | External Identifier | cmc_externalidentifier | String | |
| HoldCode | | | | NA | cmc_hold | cmc_externalsourcesystem | External Source System | cmc_externalsourcesystem | Picklist | default to CampusNexus Student |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| Hold Code | Action | SyHoldCode | Description | NA | cmc_hold | cmc_holdimpact | Impact | cmc_holdimpact | Picklist | |
| Hold Code | Module | SyModule | Descrip | NA | cmc_hold | cmc_module | Module | cmc_module | Picklist | |
| Hold Code | Description | SyHoldCode | Description | NA | cmc_hold | cmc_name | Name | cmc_name | String | |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| HoldCode | seenotes | SyHold | syGroupsId | NA | cmc_cmc_group_cmc_hold | cmc_groupid | GroupId | cmc_groupid | | HoldCode and Groups have a M:M relationship. syHold table in CNS maintains this data |
| seenotes | NA | NA | NA | NA | cmc_holdassignment | cmc_contactid | Contact | cmc_contactId | Lookup | Group Membership. Contact |
| seenotes | NA | NA | NA | NA | cmc_holdassignment | cmc_enddate | End Date | cmc_enddate | DateTime | Group Membership. End Date |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| see notes | NA | NA | NA | NA | cmc_holdassignment | cmc_hold | Hold | cmc_hold | Lookup | Lookup to Hold with GroupMembership.Group |
| see notes | NA | NA | NA | NA | cmc_holdassignment | cmc_name | Name | cmc_name | String | Defaulted to the format - HoldAssignment-{GroupMembership Name} |
| see notes | NA | NA | NA | NA | cmc_holdassignment | cmc_startdate | Start Date | cmc_startdate | DateTime | GroupMembership.Start Date |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| see notes | NA | NA | NA | NA | cmc_cmc_groupmembership_cmc_holdassignmentset | cmc_groupmembershipid | GroupMembership | cmc_groupmembershipid | Uniqueidentifier | GroupMembership and HoldAssignment records have a M:M relationship |

**Previous Education > Education Level (Reference Data)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Previous Education | Id | AmPrevEduc | amPrevEducID | String | Education Level | mshied_educationlevel | External Identifier | mshied_externalidentifier | String |
| Previous Education | Name | AmPrevEduc | Descrip | String | Education Level | mshied_educationlevel | Education Level Name | mshied_name | String |
| Previous Education | Code | AmPrevEduc | Code | String | Education Level | mshied_educationlevel | Code | mshied_code | String |
| Previous Education | N/A | AmPrevEduc | N/A | | Education Level | mshied_educationlevel | External Source System | mshied_externalsourcesystem | Optionset |

**Program > Program (Reference Data)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Program | Id | AdProgram | AdProgramID | String | Program | mshied_program | External Identifier | mshied_externalidentifier | String |
| Program | Description | AdProgram | Descrip | String | Program | mshied_program | Program Name | mshied_name | String |
| Program | Code | AdProgram | Code | String | Program | mshied_program | Code | mshied_code | String |
| Program | N/A | AdProgram | N/A | | Program | mshied_program | External Source System | mshied_externalsourcesystem | Optionset |
| Program | Active | AdProgram | Active | Boolean | Program | mshied_program | Status | statecode | Optionset |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Program | Active | AdProgram | Active | Boolean | Program | mshied_program | Status Reason | statuscode | Optionset |

**Program Group > Area of Interest (Reference Data)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Program Group | Id | AdProgramGroup | AdProgramGroupID | String | Area of Interest | mshied_areaofinterest | External Identifier | mshied_external identifier | String |
| Program Group | Code | AdProgramGroup | Code | String | Area of Interest | mshied_areaofinterest | Code | mshied_code | String |
| Program Group | Name | AdProgramGroup | Descrip | String | Area of Interest | mshied_areaofinterest | Area of Interest Name | mshied_name | String |
| Program Group | N/A | AdProgramGroup | N/A | | Area of Interest | mshied_areaofinterest | External Source System | mshied_external source system | Optionset |

**Program Version > ProgramVersion (Reference Data)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| ProgramVersion | Id | AdProgramVersion | AdProgramVersionID | String | ProgramVersion | mshied_programversion | ExternalIdentifier | mshied_externalidentifier | String |
| ProgramVersion | VersionCode | AdProgramVersion | Code | String | ProgramVersion | mshied_programversion | Code | mshied_code | String |
| ProgramVersion | VersionName | AdProgramVersion | Descrip | String | ProgramVersion | mshied_programversion | ProgramVersionName | mshied_name | String |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| ProgramVersion | Program | AdProgramVersion | AdProgramID | Lookup | Program Version | mshied_programversion | Program | mshied_programid | Lookup |
| ProgramVersion | Degree | AdProgramVersion | AdDegreeID | Lookup | Program Version | mshied_programversion | Program Level | mshied_programlevelid | Lookup |
| ProgramVersion | N/A | AdProgramVersion | N/A | | Program Version | mshied_programversion | External Source System | mshied_externalsourcesystem | Optionset |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| ProgramVersion | Active | AdProgramVersion | Active | Boolean | ProgramVersion | msdyn_programversion | Status | statecode | Optionset |
| ProgramVersion | Active | AdProgramVersion | Active | Boolean | ProgramVersion | msdyn_programversion | Status Reason | statuscode | Optionset |

**Shift > Shift (Reference Data)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Shift | Id | AdShift | AdShiftID | String | Shift | cmc_shift | External Identifier | cmc_externalsisid | String |
| Shift | Description | AdShift | Descrip | String | Shift | cmc_shift | Shift Name | cmc_shiftname | String |
| Shift | Code | AdShift | Code | String | Shift | cmc_shift | Code | cmc_code | String |
| Shift | N/A | AdShift | N/A | | Shift | cmc_shift | N/A | cmc_externalsourcesystem | Optionset |

**Start Date > Program Version Detail (Reference Data)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| StartDate | Id | AdStartDate | AdStartDateID | String | ProgramVersionDetail | mshied_programversiondetail | External Identifier | mshied_externalidentifier | String |
| StartDate | Name | AdStartDate | Descrip | String | ProgramVersionDetail | mshied_programversiondetail | Program Version Detail Name | mshied_name | String |
| StartDate | Code | AdStartDate | Code | String | ProgramVersionDetail | mshied_programversiondetail | Code | mshied_code | String |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| StartDate | StartDate | AdStartDate | StartDate | DateTime | ProgramVersionDetail | mshied_programversiondetail | StartDate | mshied_startdate | DateTime |
| StartDate | ExpectedGradDate | AdStartDate | ExpGradDate | DateTime | ProgramVersionDetail | mshied_programversiondetail | ExpectedGraduationDate | mshied_expectedgraduationdate | DateTime |
| StartDate | MidPointDate | AdStartDate | MidPointDate | DateTime | ProgramVersionDetail | mshied_programversiondetail | MidpointDate | mshied_midpointdate | DateTime |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| StartDate | MaxGradDate | AdStartDate | MaxGradDate | DateTime | ProgramVersionDetail | mshied_programversiondetail | MustGraduateBefore | mshied_mustgraduatebefore | DateTime |
| StartDate | ProgramVersionId | AdStartDate | AdProgramVersionId | Lookup | ProgramVersionDetail | mshied_programversiondetail | ProgramVersion | mshied_programversionid | Lookup |
| StartDate | N/A | AdStartDate | N/A | | ProgramVersionDetail | mshied_programversiondetail | ExternalSourceSystem | mshied_externalsourcesystem | Optionset |

**Student Group > Group (Operational Data)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Student Group | Id | SyGroups | SyGroupsID | String | Group | cmc_group | External Identifier | cmc_externalidentifier | String |
| Student Group | Name | SyGroups | Descrip | String | Group | cmc_group | Name | cmc_name | String |
| Student Group | Code | SyGroups | Code | String | Group | cmc_group | Code | cmc_code | String |
| Student Group | Date Added | SyGroups | DateAdded | DateTime | Group | cmc_group | Record Created On | overriddencreatedon | DateTime |
| Student Group | Expiration Date | SyGroups | DateExpires | DateTime | Group | cmc_group | Expiration Date | cmc_expirationdate | DateTime |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Student Group | owner Staff | SyGroups | SyStaffID | Lookup | Group | cmc_group | Owner | ownerid | Lookup |
| Student Group | N/A | SyGroups | N/A | Optionset | Group | cmc_group | Membership Required | cmc_membership required | Boolean |
| Student Group | N/A | SyGroups | N/A | N/A | Group | cmc_group | External Source System | cmc_external source system | Optionset |
| Student Group | Active | SyGroups | Active | Boolean | Group | cmc_group | Status | statecode | Boolean |
| Student Group | Active | SyGroups | Active | Boolean | Group | cmc_group | Status Reason | statuscode | Boolean |

**Student Group Membership > Group Membership (Special Handling)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Student Group Membership | NA | NA | NA | NA | Group Membership | cmc_ group membership | External Source System | cmc_ external source system | Optionset |
| Student Group Membership | Id | SyStudGrp | SyStudGrpID | int | Group Membership | cmc_ group membership | External Identifier | cmc_ external identifier | String |
| Student Group Membership | EffectiveDate | SyStudGrp | effectivedate | DateTime | Group Membership | cmc_ group membership | Start Date | cmc_ startdate | DateTime |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| StudentGroupMembership | RemovedDate | SyStudGrp | DateOff | DateTime | GroupMembership | cmc_groupmembership | EndDate | cmc_enddate | DateTime |
| StudentGroupMembership | CreatedDateTime | SyStudGrp | DateAdded | DateTime | GroupMembership | cmc_groupmembership | CreatedOn | createdon | Datetime |
| StudentGroupMembership | StudentGroupId | SyStudGrp | SyGroupsID | int | GroupMembership | cmc_groupmembership | Group | cmc_group | Lookup |
| StudentGroupMembership | Student.PersonId | SyStudGrp | sypersonid | int | GroupMembership | cmc_groupmembership | Contact | cmc_contact | Lookup |

**Student Relationship Address > Address (Operational Data)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Student Relationship Address | Id | SyAddress | SyAddressID | String | Address | customer address | External Identifier | mshied_external identifier | String |
| Student Relationship Address | N/A | SyAddress | N/A | N/A | Address | customer address | External Source System | mshied_external source system | Optionset |
| Student Relationship Address | Person | SyAddress | SyStudentID | Lookup | Address | customer address | Parent | parentid | Lookup |
| Student Relationship Address | Address Type | SyAddress | SyAddrTypeID | Optionset | Address | customer address | Address Type | addressstype code | Optionset |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Student Relationship Address | Country Name | SyAddress | SyCounrtyID | Optionset | Address | customer address | Country/ Region | cmc_countryregion | Optionset |
| Student Relationship Address | City | SyAddress | City | String | Address | customer address | City | city | String |
| Student Relationship Address | State | SyAddress | State | Optionset | Address | customer address | State/ Province | cmc_state | Optionset |
| Student Relationship Address | County | SyAddress | SyCountyID | Optionset | Address | customer address | County | county | Optionset |
| Student Relationship Address | Street Address | SyAddress | Addr1 | String | Address | customer address | Street 1 | line1 | String |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Student Relationship Address | Street Address2 | SyAddress | Addr2 | String | Address | customer address | Street 2 | line2 | String |
| Student Relationship Address | Address End Date | SyAddress | EndDate | DateTime | Address | customer address | Address End | mshied_enddate | DateTime |
| Student Relationship Address | Address Begin Date | SyAddress | BeginDate | DateTime | Address | customer address | Address Start | mshied_startdate | DateTime |
| Student Relationship Address | Postal Code | SyAddress | Zip | String | Address | customer address | ZIP/Postal Code | postalcode | String |
| Student Relationship Address | Date Added | SyAddress | DateAdded | DateTime | Address | customer address | Record Created On | overridden createdon | DateTime |

**Student Status History > Student Status History (Special Handling)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Student Status History | NA | NA | NA | NA | Student Status History | cmc_student status history | External Identifier | cmc_external identifier | String |
| Student Status History | Id | SyStatChange | SyStat ChangeID | int | Student Status History | cmc_student status history | External Identifier Numeric | cmc_external identifier numeric | Number |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Student Status History | Student.PersonId | SySt-atCh-ange | SySt-udent-ID | int | Stu-den-t St-at-us Hi-st-or-y | c-m-c_ st-ud-en-t st-at-us hi-st-or-y | Con-tact | cm-c_ con-tacti-d | Loo-kup |
| Student Status History | Stu-dent Enrol-lmen-t Peri-odId | SySt-atCh-ange | aden-rollID | int | Stu-den-t St-at-us Hi-st-or-y | c-m-c_ st-ud-en-t st-at-us hi-st-or-y | Enr-oll-me-nt | cm-c_ enro-llme-ntid | Loo-kup |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Student Status History | NewSchoolStatusId | SySt atChange | NewSySchoolStatusID | int | Student Status History | cmc_student status history | New Status | cmc_newstatusid | Lookup |
| Student Status History | PreviousSchoolStatusId | SySt atChange | PrevSySchoolStatusID | int | Student Status History | cmc_student status history | Previous Status | cmc_previous statusid | Lookup |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| StudentStatusHistory | EffectiveDate | SyStatChange | EffectiveDate | DateTime | StudentStatusHistory | cmc_studentstatushistory | StartDate | cmc_startdate | Datetime |
| StudentStatusHistory | CreatedDateTime | SyStatChange | DateAdded | DateTime | StudentStatusHistory | cmc_studentstatushistory | DateAdded | cmc_dateadded | Datetime |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Student Status History | Note | SySt-atCh-ange | Com-ment | stri-ng | Stu-den-t St-at-us Hi-st-or-y | c-m-c_ st-ud-en-t st-at-us hi-st-or-y | Co-mm-ents | cm-c_ com-men-ts | Stri-ng |
| Student Status History | NA | NA | NA | NA | Stu-den-t St-at-us Hi-st-or-y | c-m-c_ st-ud-en-t st-at-us hi-st-or-y | Ext-ern-al Sou-rce Sys-tem | cm-c_ exte-rnal sour-ce sys-tem | Opt-ion-set |

**Term > Academic Period (Reference Data)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Term | Id | AdTerm | AdTermID | String | Academic Period | mshied_academic period | External Identifier | mshied_external identifier | String |
| Term | Code | AdTerm | Code | String | Academic Period | mshied_academic period | Code | mshied_code | String |
| Term | Name | AdTerm | Descrip | String | Academic Period | mshied_academic period | Academic Period Name | mshied_name | String |
| Term | StartDate | AdTerm | StartDate | DateTime | Academic Period | mshied_academic period | Start Date | mshied_startdate | DateTime |
| Term | EndDate | AdTerm | EndDate | DateTime | Academic Period | mshied_academic period | End Date | mshied_enddate | DateTime |
| Term | IsActive | AdTerm | Active | Option-set | Academic Period | mshied_academic period | Status | statecode | Option-set |
| Term | IsActive | AdTerm | Active | Option-set | Academic Period | mshied_academic period | Status Reason | statuscode | Option-set |
| Term | N/A | AdTerm | N/A | | Academic Period | mshied_academic period | External Source System | mshied_external source system | Option-set |

**Data Sent from Anthology Reach to Anthology Student**

**Enrollment > Application (Operational Data)**

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| Enrollment | applicationReceivedDate | Ad-Enroll | AppRecDate | DateTime | Application | cmc_application | SubmittedDate | cmc_submitteddate | DateTime | |
| Enrollment | programId | Ad-Enroll | AdProgramID | Lookup | Application | cmc_application | Program | cmc_programid | Number | |
| Enrollment | programVersionId | Ad-Enroll | adProgramVersionID | Lookup | Application | cmc_application | Program Version | cmc_programversionid | Number | |
| Enrollment | shiftId | Ad-Enroll | adShiftID | Lookup | Application | cmc_application | Shift | cmc_shiftid | Number | |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| Enrollment | campusId | Ad-Enroll | SyCampus ID | Lookup | Application | cmc_application | Lifecycle | cmc_opportunityid | Number | |
| Enrollment | expectedStartDate | Ad-Enroll | ExpStart Date | DateTime | Application | cmc_application | Application Period | cmc_application periodid | Number | |
| Enrollment | gradeLevelId | Ad-Enroll | AdGrade LevelId | Number | Application | cmc_application | gradeLevelId | NA | Number | defaulted to 0 |
| Enrollment | enrollment Date | Ad-Enroll | EnrollDate | DateTime | Application | cmc_application | enrollment Date | NA | DateTime | CURRENT DATE,, yyyy-MM-ddTHH:mm:ss |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| Enrollment | studentId | Ad-Enroll | SyStudentId | Number | Application | cmc_application | studentId | NA | Number | @{variables('varStudentId')} StudentId is retrieved based on the PersonId maintained in the Contact's External Id |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| Enrollment | graduationDate | Ad_Enroll | Grad_Date | DateTime | Program Version Detail | cmc_application | Graduation Date | application.programversion.programversiondetail.ExpectedGraduationDate | DateTime | application.programversion.programversiondetail.expectedGraduationDate should be equal to appln.applnperiod.academicperiod.startdate |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| Enrollment | schoolStatusId | Ad-Enroll | SySchoolStatusID | Lookup | Application | cmc_application | schoolStatusId | NA | Number | defaulted |
| Enrollment | assignedAdmissionsRepId | Ad-Enroll | AmRepID | int | Contact | contact | owner | owner | | Fallback is provided in the mapping |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| Enrollment | startDateId | Ad-Enroll | AdStartDateID | int | Program Version Detail | cmc_application | mshied_ExternalIdentifier | application.programversion.programversiondetail.externalid | | application.programversion.programversiondetail.expectedGraduationDate should be equal to appln.applnperiod.academicperiod.startdate |

**Data Sent Bi-directionally between Anthology Reach and Anthology Student**

**Student > Contact (Operational Data)**

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Bi-Directional | Student | LastName | SyStudent | LastName | String | Contact | contact | Last Name | lastname | String | |
| One-way (Student to Reach) | Student | N/A | SyStudent | N/A | N/A | Contact | contact | Source Category | cmc_sourcecategoryid | Lookup | |

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|---|
| One-way (Student to Reach) | Student | N/A | SyStudent | N/A | N/A | Contact | contact | Source Method | cmc_sourcemethodid | Lookup | |
| Bi-Directional | Student | Work Phone Number | SyStudent | Work Phone | String | Contact | contact | Business Phone | telephone1 | String | |
| Bi-Directional | Student | Email Address | SyStudent | email | String | Contact | contact | Email | emailaddress1 | String | |

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Bi-Directional | Student | First Name | SyStudent | FirstName | String | Contact | contact | First Name | firstname | String | |
| Bi-Directional | Student | Mobile Phone Number | SyStudent | Mobile Phone | String | Contact | contact | Mobile Phone | mobilephone | String | |
| Bi-Directional | Student | Campus | SyStudent | SyCampus ID | Lookup | Contact | contact | Current Campus | parentcustomerid | Lookup | |

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Bi-Directional | Student | Phone Number | SyStudent | Phone | String | Contact | contact | Home Phone | telephone2 | String | |
| Bi-Directional | Student | Other Phone Number | SyStudent | Other Phone | String | Contact | contact | Address 1: Telephone 3 | address1_telephone3 | String | |
| Bi-Directional | Student | Date Of Birth | SyStudent | DOB | DateTime | Contact | contact | Birthday | birthdate | DateTime | |

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Bi-Directional | Student | Other Email Address | SyStudent | OtherEmail | String | Contact | contact | Email Address 2 | emailaddress2 | String | |
| Bi-Directional | Student | Person | SyStudent | SyPersonId | Lookup | Contact | contact | External Identifier | mshied_externalidentifier | String | |
| Bi-Directional | Student | DataBlock Indicator | SyStudent | DataBlockIndicator | Boolean | Contact | contact | FERPA Privacy | mshied_ferpaprivacy | Boolean | |

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Bi-Directional | Student | Gender | SyStudent | AmSexID | Optionset | Contact | contact | Gender | gendercode | Optionset | |
| Bi-Directional | Student | Disabled | SyStudent | Disabled | Optionset | Contact | contact | HIPAA Indicator | mshied_hipaaindicator | Boolean | |
| Bi-Directional | Student | Country | SyStudent | SyCountryID | Optionset | Contact | contact | Address 1: Country/Region | address1_country | String | |

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Bi-Directional | Student | State | SyStudent | State | Optionset | Contact | contact | Address 1: State/Province | address1_stateorprovince | String | |
| Bi-Directional | Student | Street Address | SyStudent | Addr1 | String | Contact | contact | Address 1: Street 1 | address1_line1 | String | |
| Bi-Directional | Student | City | SyStudent | City | String | Contact | contact | Address 1: City | address1_city | String | |

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Bi-Directional | Student | Postal Code | SyStudent | Zip | String | Contact | contact | Address 1: ZIP/-Postal Code | address1_postalcode | String | |
| Bi-Directional | Student | County | SyStudent | SyCountyID | Optionset | Contact | contact | Address 1: County | address1_county | String | |
| Bi-Directional | Student | Maiden Name | SyStudent | Maiden Name | String | Contact | contact | Maiden Name | mshied_maidenname | String | |

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Bi-Directional | Student | Marital Status | SyStudent | AmMaritalID | Optionset | Contact | contact | Marital Status | familystatuscode | Optionset | |
| Bi-Directional | Student | Title | SyStudent | AmTitleID | Optionset | Contact | contact | Title | cmc_title | Optionset | |
| Bi-Directional | Student | Middle Name | SyStudent | Middle Name | String | Contact | contact | Middle Name | middlename | String | |
| Bi-Directional | Student | Ssn | SyStudent | SSN | String | Contact | contact | National Identifier | mshied_nationalidentifier | String | |

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Bi-Directional | Student | Nationality | SyStudent | AmNationalityID | Optionset | Contact | contact | Nationality | mshied_nationality | Optionset | |
| Bi-Directional | Student | NickName | SyStudent | NickName | String | Contact | contact | Nickname | nickname | String | |
| One-way (Student to Reach) | Student | EthnicitiesList | SyStudent | AmRaceID | MultiOptionSet | Contact | contact | Race | mshied_race_ | Multiselect | |

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Bi-Directional | Student | StudentNumber | SyStudent | StuNum | String | Contact | contact | StudentNumber | cmc_studentnumber | String | |
| Bi-Directional | Student | Suffix | SyStudent | AmSuffixID | Optionset | Contact | contact | Suffix | suffix | Optionset | |
| Bi-Directional | Student | WorkPhoneNumberExtension | SyStudent | Workext | String | Contact | contact | WorkPhoneExtn. | cmc_phone2extension | String | |

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Bi-Directional | Student | School Status | SyStudent | SySchoolStatusID | Lookup | Contact | contact | Student Status | mshied_studentstatusid | Lookup | |
| Bi-Directional | Student | Citizen | SyStudent | AmCitizenID | Number | Contact | contact | Citizenship Status | mshied_citizenshipstatus | Optionset | |
| Bi-Directional | Student | Shift | SyStudent | AdShiftID | Lookup | Contact | contact | Shift | cmc_shiftid | Lookup | |

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|---|
| One-way (Student to Reach) | Student | N/A | SyStudent | N/A | N/A | Contact | contact | Contact Type | mshied_contacttype | Multiselect | mshied_contacttype = Student |
| One-way (Student to Reach) | Student | N/A | SyStudent | N/A | N/A | Contact | contact | External Source System | mshied_externalsourcesystem | Optionset | |
| Bi-Directional | Student | Hispanic Latino | SyStudent | IsHispanic | Optionset | Contact | contact | Ethnic Group | mshied_ethnicgroup | Optionset | |

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|---|
| One-way (Student to Reach) | Student | DateAdded | SyStudent | DateAdded | DateTime | Contact | contact | Record Created On | overriddencreatedon | DateTime | |
| One-way (Reach to Student) | Student | isActive | SyStudent | statecode | | Contact | contact | N/A | | | default value true |

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type | Notes |
|---|---|---|---|---|---|---|---|---|---|---|---|
| One-way (Reach to Student) | Student | leadDate | SyStudent | leadDate | DateTime | Contact | contact | N/A | | | CURRENT DATE,,,yyyy/MM/ddHH:mm:ss |

**StudentPreviousEducation <> Previous Education (Operational Data)**

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|---|
| Bi-Directional | Student Previous Education | Previous Education | AmProspect PrevEduc | AmProspect PrevEducID | Lookup | Previous Education | mshied_previous educations | Education Level | mshied_education levelid | Lookup |
| Bi-Directional | Student Previous Education | Grade Level | AmProspect PrevEduc | AmGradeLevelID | Lookup | Previous Education | mshied_previous educations | Education Level | mshied_education levelid | Lookup |
| Bi-Directional | Student Previous Education | High School | AmProspect PrevEduc | AmHighSchoolID | Lookup | Previous Education | mshied_previous educations | School Name | mshied_school nameid | Lookup |

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|---|
| Bi-Directional | StudentPreviousEducation | College | AmProspectPrevEduc | AmCollegeID | Lookup | PreviousEducation | msshied_previouseducations | School Name | msshied_schoolnameid | Lookup |
| One-way (Student to Reach) | StudentPreviousEducation | Person | AmProspectPrevEduc | SyStudentID | Lookup | PreviousEducation | msshied_previouseducations | Student | msshied_studentid | Lookup |
| Bi-Directional | StudentPreviousEducation | SchoolSize | AmProspectPrevEduc | SchoolSize | Integer | PreviousEducation | msshied_previouseducations | Class Size | msshied_classsize | Integer |

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|---|
| Bi-Directional | StudentPreviousEducation | EnrollmentDate | AmProspectPrevEduc | DateOfEnrollment | DateTime | PreviousEducation | mshied_previouseducations | Date of Enrollment | mshied_dateofenrollment | DateTime |
| Bi-Directional | StudentPreviousEducation | Gpa | AmProspectPrevEduc | GPA | Decimal | PreviousEducation | mshied_previouseducations | GPA | mshied_gpa | Decimal |
| Bi-Directional | StudentPreviousEducation | IsGraduated | AmProspectPrevEduc | Graduated | Boolean | PreviousEducation | mshied_previouseducations | Graduated | mshied_graduated | Boolean |

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|---|
| Bi-Directional | StudentPreviousEducation | GraduationDate | AmProspectPrevEduc | GraduationDate | DateTime | PreviousEducation | msshied_previouseducations | GraduationDate | msshied_graduationdate | DateTime |
| Bi-Directional | StudentPreviousEducation | LastAttendedDate | AmProspectPrevEduc | LastDateAttended | DateTime | PreviousEducation | msshied_previouseducations | Last Date of Attendance | msshied_lastdateofattendance | DateTime |
| Bi-Directional | StudentPreviousEducation | Major | AmProspectPrevEduc | Major | String | PreviousEducation | msshied_previouseducations | Major | msshied_major | String |

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|---|
| Bi-Directional | Student Previous Education | Student Rank | AmProspect PrevEduc | StudRank | Integer | Previous Education | mshied_previouseducations | Rank | mshied_rank | Integer |
| Bi-Directional | Student Previous Education | Degree | AmProspect PrevEduc | DegreeiD | Lookup | Previous Education | mshied_previouseducations | Degree (Program Level) | mshied_Degreeid | Lookup |
| Bi-Directional | Student Previous Education | Id | AmProspect PrevEduc | AmPrevEducID | String | Previous Education | mshied_previouseducations | External Identifier | cmc_externalidentifier | String |

| Integration Direction | Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|---|
| One-way (Student to Reach) | Student Previous Education | N/A | AmProspect PrevEduc | N/A | N/A | Previous Education | mshied_previous educations | External Source System | cmc_external source system | Optionset |
| One-way (Reach to Student) | Student Previous Education | previous Education | AmProspect PrevEduc | AmProspect PrevEduId | Number | Previous Education | mshied_previous educations | Education Level | mshied_education levelid | String |
| One-way (Reach to Student) | Student Previous Education | student | AmProspect PrevEduc | SyStudentID | Number | Previous Education | mshied_previous educations | N/A | | |

**Notes**:

Each entity which is integrated has two fields:

- External Identifier — This field had the record Id as maintained in Anthology Student.
- External Source System — This field indicates the source of the integration. For records integrated with Anthology Student, the value of this field is “CampusNexus Student".

These fields are used to identify any record which has previously been integrated.

For the Student entity, the PersonId field in Anthology Student is used as the External Identifier because the Contact entity in Anthology Reach will integrate with other entities like Staff, Instructor in the future.

The Contact entity has other duplicate criteria in addition to the External identifier:

- First Name, Last and Date of Birth
- Email Address
- Ssn

The Account entity has an additional duplicate check on the Name field.

The 'mshied_' entities represent higher education-specific extensions to the common data model (Microsoft Dynamics 365 Education Accelerator).

## Field Mappings for Service Bus Integrations

Subscriptions to business event integrations via the Azure Service Bus provide predefined field mappings as listed below. Clients can use the subscription based integrations for the entities listed below along side the flow-based integrations or migrate from the flow-based integration to use only the service bus integration.

### Data Sent from Anthology Student to Anthology Reach

### Student Status History > Student Status History (Special Handling)

A subscription to the "Student Course Status Has Changed" event provides the following predefined field mappings.

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Student Status History | NA | NA | NA | NA | Student Status History | cmc_student status history | External Identifier | cmc_external identifier | String |
| Student Status History | Id | SyStatChange | SyStat ChangeID | int | Student Status History | cmc_student status history | External Identifier Numeric | cmc_external identifier numeric | Number |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Student Status History | Student.PersonId | SySt-atCh-ange | SySt-udent-ID | int | Stu-den-t St-at-us Hi-st-or-y | c-m-c_ st-ud-en-t st-at-us hi-st-or-y | Con-tact | cm-c_ con-tacti-d | Loo-kup |
| Student Status History | Stu-dent Enrol-lmen-t Peri-odId | SySt-atCh-ange | aden-rollID | int | Stu-den-t St-at-us Hi-st-or-y | c-m-c_ st-ud-en-t st-at-us hi-st-or-y | Enr-oll-me-nt | cm-c_ enro-llme-ntid | Loo-kup |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Student Status History | NewSchoolStatusId | SySStatChange | NewSySchoolStatusID | int | Student Status History | cmc_student status history | New Status | cmc_newstatusid | Lookup |
| Student Status History | PreviousSchoolStatusId | SySStatChange | PrevSySchoolStatusID | int | Student Status History | cmc_student status history | Previous Status | cmc_previous statusid | Lookup |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| StudentStatusHistory | EffectiveDate | SyStatChange | EffectiveDate | DateTime | StudentStatusHistory | cmc_studentstatushistory | StartDate | cmc_startdate | Datetime |
| StudentStatusHistory | CreatedDateTime | SyStatChange | DateAdded | DateTime | StudentStatusHistory | cmc_studentstatushistory | DateAdded | cmc_dateadded | Datetime |

| Student Entity Name | Student Property Name | Student Database Table Name | Student Database Field Name | Data Type | Reach Entity Name | Reach Logical Entity Name | Reach Field Display Name | Reach Logical Field Name | Reach Data Type |
|---|---|---|---|---|---|---|---|---|---|
| Student Status History | Note | SySt-atCh-ange | Com-ment | stri-ng | Stu-den-t St-at-us Hi-st-or-y | c-m-c_ st-ud-en-t st-at-us hi-st-or-y | Co-mm-ents | cm-c_ com-men-ts | Stri-ng |
| Student Status History | NA | NA | NA | NA | Stu-den-t St-at-us Hi-st-or-y | c-m-c_ st-ud-en-t st-at-us hi-st-or-y | Ext-ern-al Sou-rce Sys-tem | cm-c_ exte-rnal sour-ce sys-tem | Opt-ion-set |

# Extensibility

Flow-based integrations and Service Bus integrations between Anthology Reach and Anthology Student can be extended and customized to suit the requirements of the client's environments.

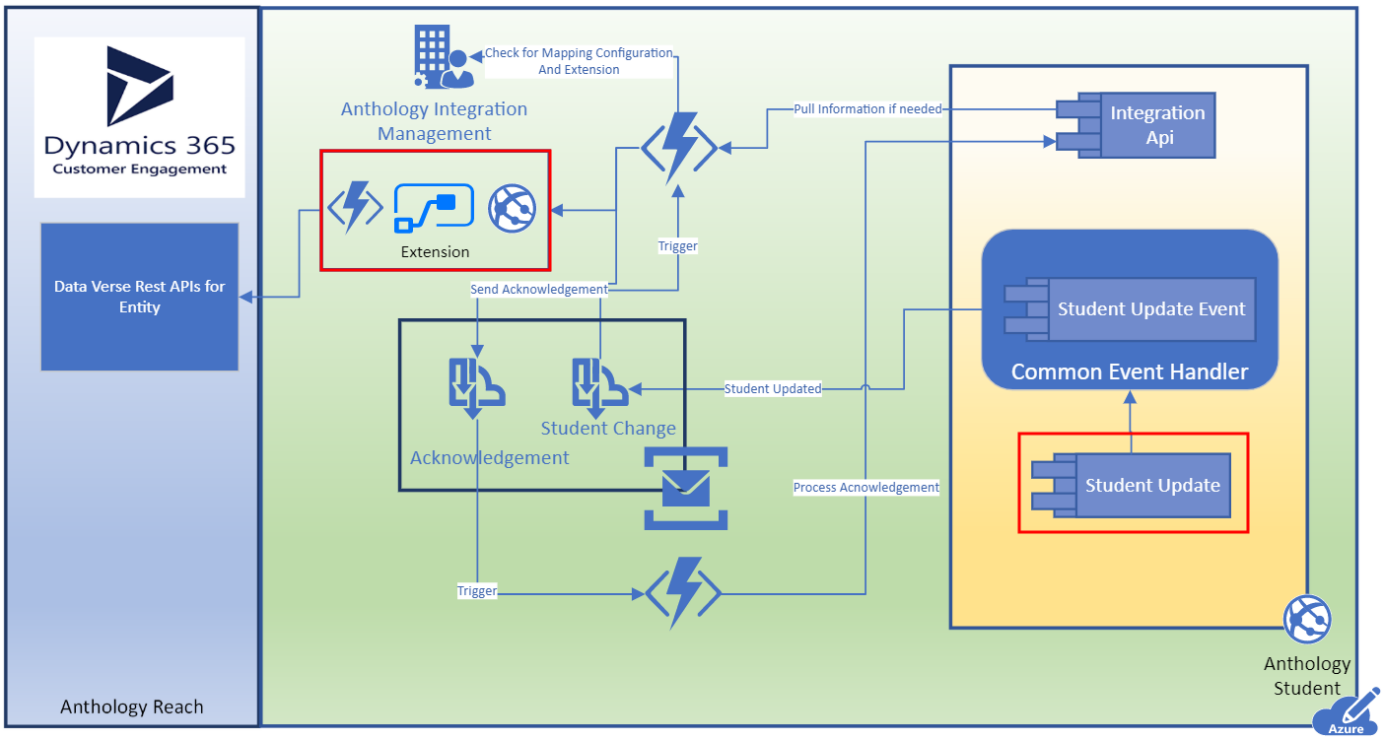
## Extensibility for Flow-based Integrations

The flow-based integrations between Anthology Reach and Anthology Student support three levels of data integration:

| Integration Level | Description | Reference |
|---|---|---|
| Out of the box integration | Predefined integration of reference and operational data with a fixed data model | Field Mappings for Flow-based Integrations |
| SDF integration and Extended Field Integration | Supports integration of School defined fields and/or extended fields as an extension to the existing framework. | From Anthology Reach to Anthology Student<br>From Anthology Student to Anthology Reach |
| Custom integration | Supports integration of objects/entities and data elements not delivered by product.<br>These customizations are made within user friendly interfaces of the Power Automate Flow, Workflow Composer, and Microsoft Dynamics 365 Forms.<br>The following customizations are available: | |
| | • Adding new options in Option Sets | Map Option Sets |
| | • Adding new Fields | From Anthology Student to Anthology Reach |
| | • Adding new Entities | |
| | • Adding conditions for integration (for example, restrict integration to a single campus, or integrate students with a specific school status | |

## Extensibility for Service Bus Integrations

Configurations in Anthology Reach are provided to configure extensions. Extensions are in the form of HTTP URLs. The URLs will be used to post any data from the Azure Function.



# Troubleshooting

This section provides tip on troubleshooting integrations between Anthology Reach and Anthology Student for flow-based integrations and Service Bus integrations.

## Troubleshooting Flow-based Integrations

### Symptoms

#### Flow is not triggered

Possible Causes

- The workflows for the corresponding entity and event are disabled.
- The Power Automate flow URL configured in the web.config is incorrect.
- For Contact flows, the Service Module Host is disabled.
- The flow has been disabled.
- The firewall setting for the flow URL is not accessible.

Actions to Take

- Check the workflow status using Workflow Composer.
- Check the flow URL in the web.config (see Configure the CNS-CNE Integration URL).
- Check the status of Service Module host (see Integration Prerequisites and ServiceModuleHost in Workflow Help).
- Enable Power Automate Flows.
- Check the firewall settings.

#### Field is not updated in Anthology Reach

Possible Causes

- The field mapping is not provided in the Integration Mapping record.
- The Power Automate flow has not been modified to include the update of the new field.
- The Integration user does not have enough privileges for performing the operation on the entity.

Actions to Take

- Check the Integration Mapping records in Anthology Reach (see Integration Prerequisites).
- Check the field references in the flow.

- Check the permissions for the Integration user in Anthology Reach.

### Flow error in the lookup data transformation steps

Possible Causes

- The corresponding lookup record of is not available in Anthology Reach.

  (Lookups for reference data have not been integrated as part of initial data migration.)

- In few cases the Lookup record might not have the correct external identifier.

Actions to Take

- If An Error Occurred in the Flow
- Check the Integration Mapping records in Anthology Reach (see Integration Prerequisites).

### Flow error in the Option Set data transformation steps

Possible Cause

- Option Set mappings are not created as records of the Integration Mapping entity.

Action to Take

- Create Integration Mapping records to add the new option (see Update Default Integration Mapping Records).

### Error while executing the CDS actions

Possible Cause

- The Integration user configured for the CDS Connection does not have sufficient privileges.

Action to Take

- Provide sufficient Security Roles for the Integration user to perform operations on the entity on which the flow is based.

### Records created from flows are not visible in Anthology Reach

Possible Cause

- The Business Unit configured for the default Integration user is accessible by the user.

Action to Take

- Modify the Business Unit for the Integration user to an appropriate Business Unit.

## Logging

See If An Error Occurred in the Flow.

### Monitoring Flows

See View the Application Insights for Student Integration Flows

# Integration between Anthology Student & Custom Payment Gateway Provider Technical Guide – Phase I

## Overview

The purpose of the document is to provide clients with technical details for a successful integration between Anthology Student and any payment gateway providers that operate under the following concepts:

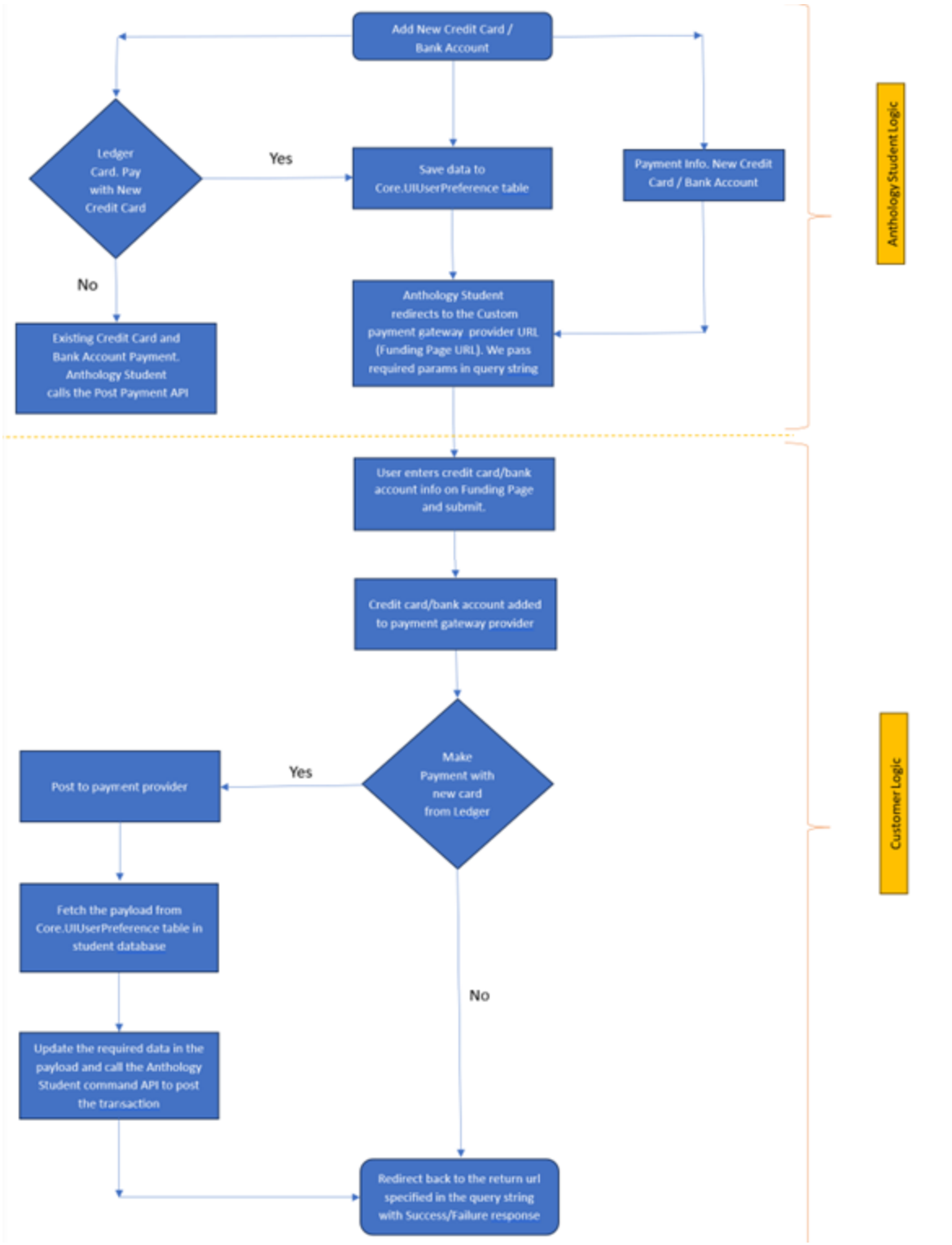
- Allow consumers to add a credit card or bank account on a dedicated funding page.
- The funding page returns a unique identifier for the host application to be stored.
- An endpoint (URL) exists to receive payment requests from the host application.

At a high level, this solution relies on existing payment services in Anthology Student to send transaction requests to the payment gateway providers to be processed. Once the requests are processed successfully, Anthology APIs can be used to post the processed transactions to Anthology Student to complete the process.

## Limitations & Practice Recommendations

- This is phase 1 and the scope is limited to:
    - Allow configuration of Credit Card and ACH electronic processors with “Custom” payment gateway provider
    - Allow URL configuration (funding URL and Post URL) for the “Custom” payment gateway provider
    - Add students’ credit cards and bank accounts to the Anthology Student web app
    - Make single credit card/ACH payments through the student’s ledger card.
- Clients must develop their own funding page that can communicate with Anthology Student and be able to use Anthology APIs to insert data into the Anthology Student database.

# Process Flow



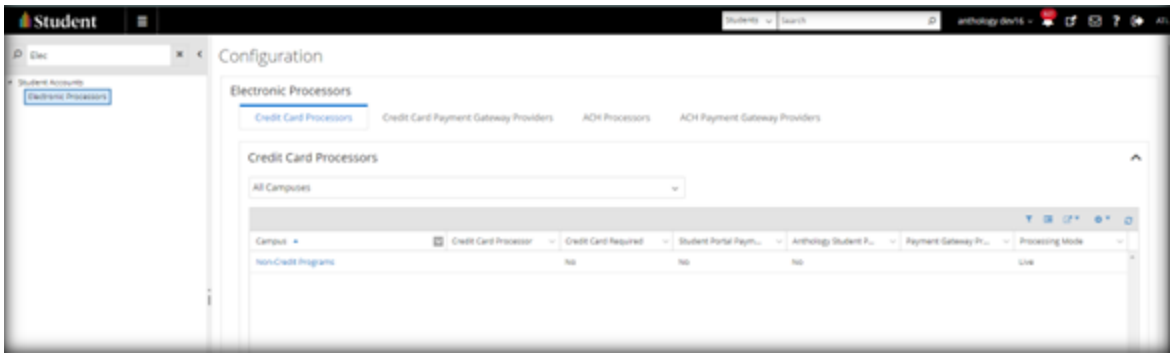
# Anthology Student Configurations

## Select Advanced Features Option

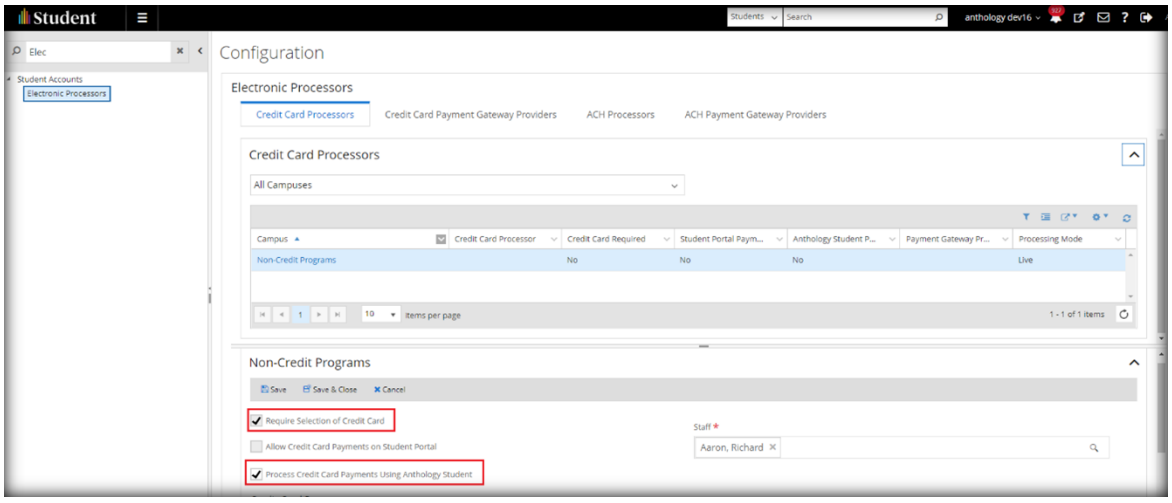
Under Settings > System > Advanced Features, the setting **Enable Additional Payment Gateway Provider for electronic processing?** should be **Yes**.

## Add Credit Card/ACH Processor

1. Navigate to Configuration > Student Accounts > **Electronic Processors** tab.

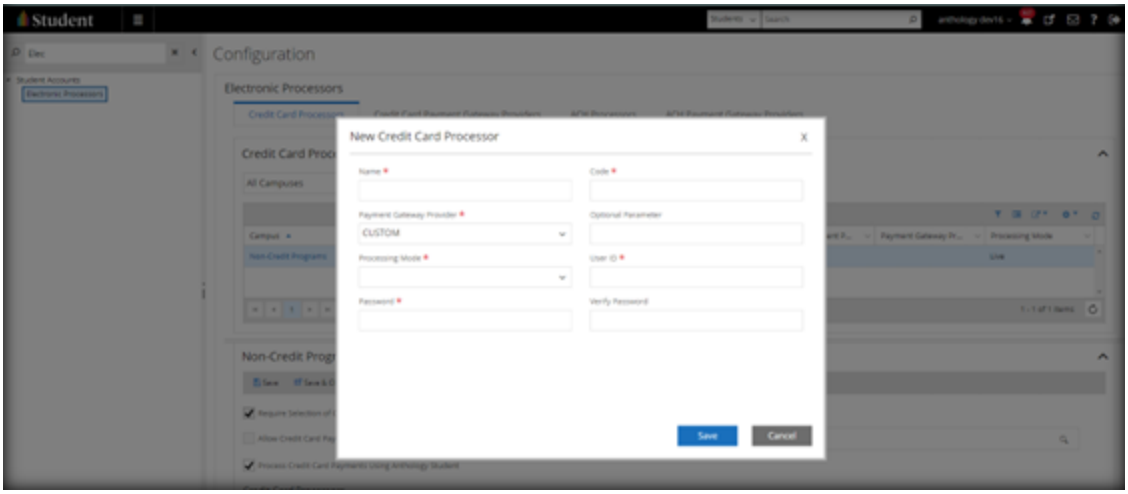


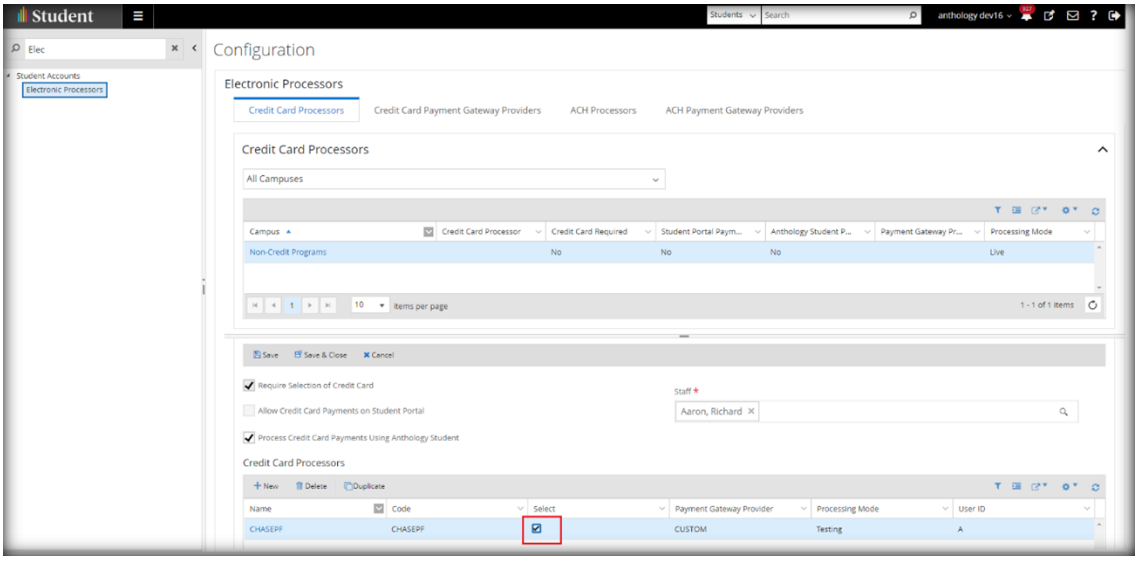

2. Click the **Campus** link to open the editable UI and select the check boxes **Require Selection of Credit Card** and **Process Credit Card Payments Using Anthology Student**.



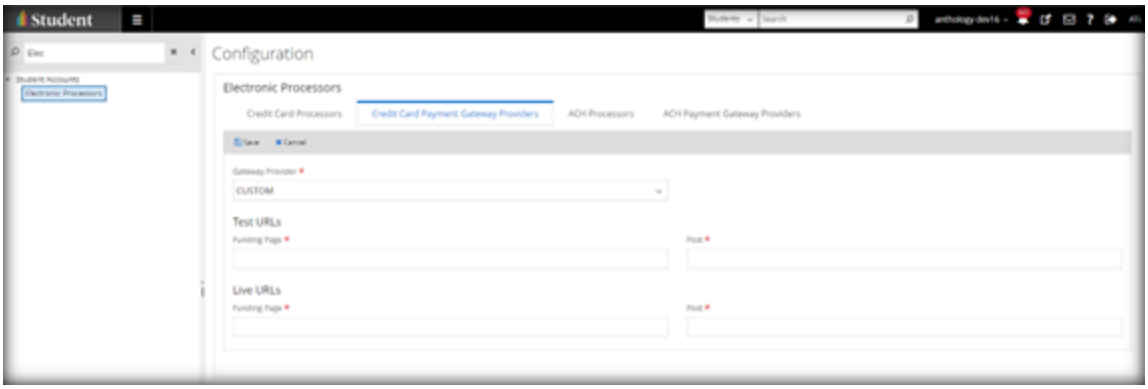

3. Click the **New** button to add the custom provider and select Payment Gateway Provider as **CUSTOM**.

   If you don't have a user ID and Password, enter dummy text in the appropriate fields and click Save. The User ID and Password are not currently being used to log in to the funding page. The funding page authentication is handled by the client. Anthology authenticates only the Post URL by providing the API token in the request header. We create the API token from the Azure AD service using the Client ID and ClientSecret, OAuthTokenUrl, and ResourceUrl. These values are provided by the client. More details are available in the API Authentication Method section.




4. Once the new payment gateway provider has been added to the grid, select the check box, and click the **Save** button.
5. Go the **Credit Card Payment Gateway Providers** tab and add valid URLs for the custom payment gateway provider. These URLs must be provided by the client.
    - **Funding Page URL** – This URL will redirect the user from Anthology Student to the client's hosted funding page to add credit cards.
    - **Post URL** – This URL will be used to post payment requests to the payment gateway provider.




Follow the steps above to add an ACH processor if needed.

## Test Tool

The test tool URLs for Anthology's internal 600045 environment are as follows:

- Credit Card/ACH Funding Page URL - https://600045appvm1.int.campusnexus.dev:888/PaymentProvider
- Credit Card/ACH Post URL - https://600045appvm1.int.campusnexus.dev:888/Payments/PostPayment

The test tool can be deployed into other internal environments upon request.

Once the deployment is done, change the URL for the **BaseServiceUrl** key on the web.config file to point to the same environment.

```
<configuration>
  <appSettings>
    <add key="webpages:Version" value="3.0.0.0" />
    <add key="webpages:Enabled" value="false" />
    <add key="PreserveLoginUrl" value="true" />
    <add key="ClientValidationEnabled" value="true" />
    <add key="UnobtrusiveJavaScriptEnabled" value="true" />
    <add key="BaseServiceUrl" value="https://600045appvm1.int.campusnexus.dev:9500/" />
  </appSettings>
```

## API Authentication Method

Anthology authenticates the custom payment gateway APIs by providing the authentication token. We use Client ID, Client Secret, Token URL, and Resource URL to generate the token. We added 4 keys in the syRegistry table for this purpose with empty values by default. These keys should be updated with valid values.

The key names are CustomPaymentProviderClientId, CustomPaymentProviderClientSecret, CustomPaymentProviderOAuthTokenUrl, and CustomPaymentProviderResourceUrl

(When a client registers its application with the authorization server, the authorization server will generate a unique Client ID and a Client Secret for that application. These credentials are typically provided by the OAuth provider (authorization server) as part of the registration process.)

**CustomPaymentProviderClientId** – This value will be added in the request header for “client_Id”.

**CustomPaymentProviderClientSecret** - This value will be added in the request header for “client_Secret”.

**CustomPaymentProviderResourceUrl** - This value will be added in the request header for “resource”.

**CustomPaymentProviderOAuthTokenUrl** – This is the endpoint that Anthology will call to generate the token.

```
<configuration>
  <appSettings>
    <add key="webpages:Version" value="3.0.0.0" />
    <add key="webpages:Enabled" value="false" />
    <add key="PreserveLoginUrl" value="true" />
    <add key="ClientValidationEnabled" value="true" />
    <add key="UnobtrusiveJavaScriptEnabled" value="true" />
    <add key="BaseServiceUrl" value="https://600045appvm1.int.campusnexus.dev:9500/" />
  </appSettings>
```

*Example*: Add a valid RegValue for all keys:

UPDATE syRegistry SET RegValue='xxxxxx' WHERE RegKey='CustomPaymentProviderClientId'

UPDATE syRegistry SET RegValue='xxxxxx' WHERE RegKey='CustomPaymentProviderClientSecret'

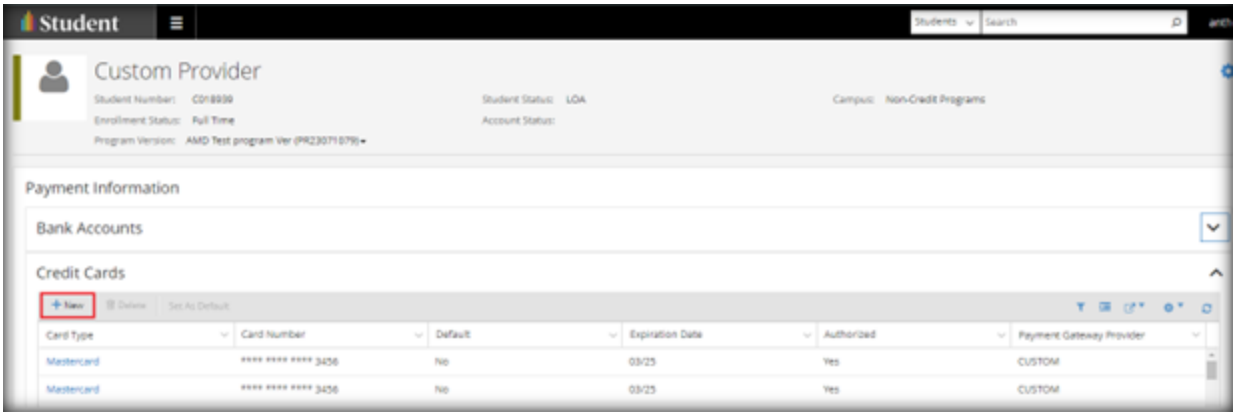
UPDATE syRegistry SET RegValue='xxxxxx' WHERE RegKey='CustomPaymentProviderOAuthTokenUrl'

UPDATE syRegistry SET RegValue='xxxxxx' WHERE RegKey='CustomPaymentProviderResourceUrl'

# Payment Info

## Add New Credit Card

Select the student > Student Accounts > Payment Information tile.




The New button in this UI will redirect to the specified custom payment provider Funding Page URL that is configured under Configuration > Student Accounts > Electronic Processors > Credit Card Payment Gateway Providers.

The userid, studentid, and returnurl will be passed as query strings. The passed URLs have been encoded.

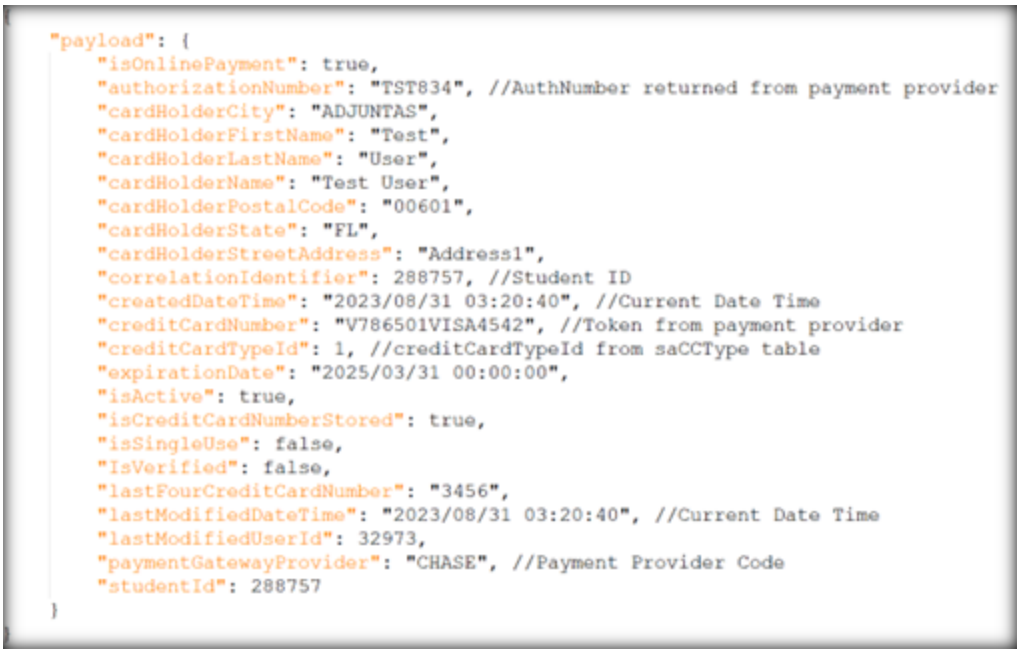
Sample URL:

https://[custompaymentproviderurl]?studentid=59655&userid=123&returnurl=http://siswebclienturl/#/students/59655/paymentInformation

Save data to the SaCC table once the credit card has been successfully added to the payment provider.

Below is the sample URL and payload to save a new credit card using the StudentCreditCard command API:

https://[siswebclienturl]/api/commands/StudentAccounts/StudentCreditCard/SaveNew



```
"payload": {
    "isOnlinePayment": true,
    "authorizationNumber": "TST834", //AuthNumber returned from payment provider
    "cardHolderCity": "ADJUNTAS",
    "cardHolderFirstName": "Test",
    "cardHolderLastName": "User",
    "cardHolderName": "Test User",
    "cardHolderPostalCode": "00601",
    "cardHolderState": "FL",
    "cardHolderStreetAddress": "Address1",
    "correlationIdentifier": 288757, //Student ID
    "createdDateTime": "2023/08/31 03:20:40", //Current Date Time
    "creditCardNumber": "V786501VISA4542", //Token from payment provider
    "creditCardTypeId": 1, //creditCardTypeId from saCCType table
    "expirationDate": "2025/03/31 00:00:00",
    "isActive": true,
    "isCreditCardNumberStored": true,
    "isSingleUse": false,
    "IsVerified": false,
    "lastFourCreditCardNumber": "3456",
    "lastModifiedDateTime": "2023/08/31 03:20:40", //Current Date Time
    "lastModifiedUserId": 32973,
    "paymentGatewayProvider": "CHASE", //Payment Provider Code
    "studentId": 288757
}
```

Using the following OData URL, the CreditCardTypeId can be fetched from the SaCCType table based on code.

https://[siswebclienturl]/ds/campusnexus/CreditCardTypes?$filter=Code eq 'V'

The sample credit card type codes are listed below.

| Credit Card Type | Code |
|---|---|
| Visa | V |
| MasterCard | M |
| Discover | D |
| Amex | A |
| American Express | A |
| Diner's Club | I |
| JCB | J |
| PayPath | P |
| Other | O |

If card is added successfully, redirect to the returnurl provided in the above query string with AddCard=true

*Example:*

http://[siswebclienturl]/#/students/59655/paymentInformation?AddCard=true

If any error, then pass the error message in query string as below.

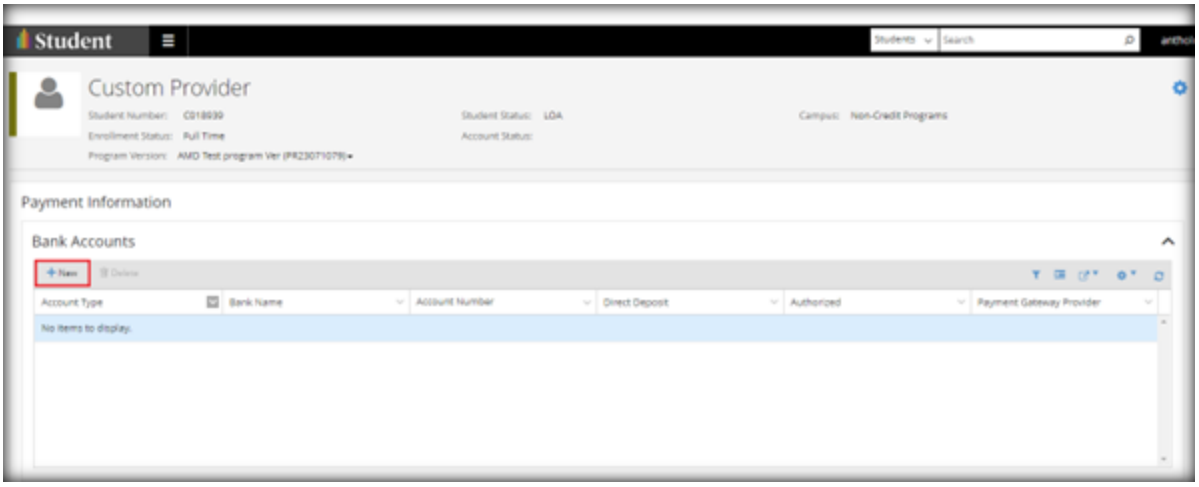
http://[siswebclienturl]/#/students/59655/paymentInformation?errormsg=error

Cancel URL.

http://[siswebclienturl]/#/students/59655/paymentInformation

## Add New Bank Account

Select the student > Student Accounts > Payment Information tile.




The New button in this UI will redirect to the specified custom payment provider Funding Page URL that is configured under Configuration > Student Accounts > Electronic Processors > ACH Payment Gateway Providers. The userid, studentid, and return URL will be passed as query strings. The passed URLs have been encoded.

Sample URL:

https://[custompaymentproviderurl]?studentid=59655&userid=123&returnurl=http://siswebclienturl/#/students/59655/paymentInformation

Save data to the SaStudentBankAccount table once the bank account has been successfully added to the payment provider.

Below is a sample URL and payload to save a bank account using the StudentBankAccount command API.

https://[siswebclienturl]/api/commands/StudentAccounts/StudentBankAccount/SaveNew

```
"payload": {
    "accountNumber": "12345678",//Token from payment provider
    "authorizationNumber": null,//AuthNumber returned from payment provider
    "createdDateTime": "2023/09/05 03:33:43", //Current Date Time
    "isDirectDeposit": false,
    "lastModifiedDateTime": "2023/09/05 03:33:43",//Current Date Time
    "lastModifiedUserId": 32973,
    "studentId": 288757,
    "paymentGatewayProvider": "CHASE",//payment gate way provider code. E.g ACI, PayPal, TouchNet, CHASE etc
    "isVerified": false,
    "bankName": "State Bank",
    "correlationIdentifier": 288757,//studentId
    "isActive": true,
    "isOnlinePayment": true,
    "LastFourNumber": "2345",//Last four account number
    "isPrimaryPayment": false,
    "isSingleUse": false,
    "routingNumber": "072403004",
    "accountType": "S"//"S" or "C"
}
```

The valid values for the account type will be one of codes from the list below.

| Account Type Name | Code |
|---|---|
| Savings | S |
| Personal Savings | S |
| Business Savings | S |
| | |
| Checking | C |
| Personal checking | C |
| Business checking | C |

If Bank Account is added successfully, redirect to the URL passed in query string with:

http://siswebclienturl/#/students/49633/paymentInformation?AddAccount=true

If any error occurs, pass the error message in query string

http://siswebclienturl/#/students/59655/paymentInformation?errormsg=error

Cancel URL:

http://siswebclienturl/#/students/59655/paymentInformation

Use the command below to read any previously saved bank accounts.

*Example*:

http://siswebclienturl/api/commands/StudentAccounts/StudentBankAccount/Get

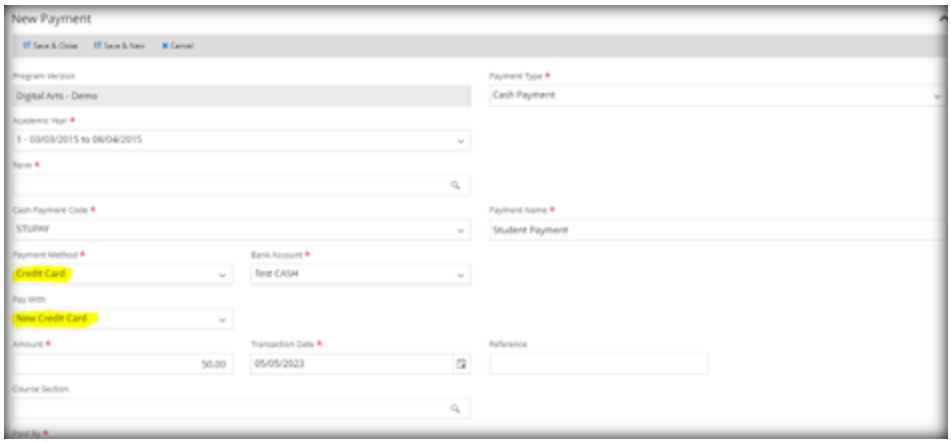
```
{
    "payload": {
        "id": "4"
    }
}
```

The decrypted bank account number will be included in the results with other details.

# Ledger Card

## Make Payment With New Credit Card

The Anthology Student web app will redirect to the configured custom payment gateway provider Funding URL when the user selects the Payment Method as “Credit Card” and Pay With as “New Credit Card” on this UI.




Sample URL:

https://[chaseurl]?studentid=59655&userid=2&userpreferenceguid=690746db-d1e3-4509-8cc9-e1e36837&returnurl=http://siswebclienturl/#/students/59655/ledgercard/transactions

The PostAccountTransactionPayment payload will be saved in the core.UIUserPreference table before redirecting for future use.

Once card is added and payment is processed successfully in the payment provider, you need to save the credit card with the details shown below.

http://localhost:90/api/commands/StudentAccounts/StudentCreditCard/SaveNew

```
{
  "payload": {
    "isOnlinePayment": true,
    "authorizationNumber": "TST834", //AuthNumber returned from payment provider
    "cardHolderCity": "ADJUNTAS",
    "cardHolderFirstName": "Test",
    "cardHolderLastName": "User",
    "cardHolderName": "Test User",
    "cardHolderPostalCode": "00601",
    "cardHolderState": "FL",
    "cardHolderStreetAddress": "Address1",
    "correlationIdentifier": 288757, //Student ID
    "createdDateTime": "2023/08/31 03:20:40", //Current Date Time
    "creditCardNumber": "V786501VISA4542", //Token from payment provider
    "creditCardTypeId": 1, //creditCardTypeId from saCCType table
    "expirationDate": "2025/03/31 00:00:00",
    "isActive": true,
    "isCreditCardNumberStored": true,
    "isSingleUse": false,
    "IsVerified": false,
    "lastFourCreditCardNumber": "3456",
    "lastModifiedDateTime": "2023/08/31 03:20:40", //Current Date Time
    "lastModifiedUserId": 32973,
    "paymentGatewayProvider": "CHASE", //Payment Provider Code
    "studentId": 288757
  }
}
```

After card is saved, post the transaction using the PostAccountTransactionPayment command API.

(api/commands/StudentAccounts/StudentAccountTransaction/PostAccountTransactionPayment)

The payload for this can be fetched using the userpreferenceguid passed in the query string.

This data is saved in the core.UIUserPreference table. The value column contains the payload.

*Example*

```
select [value] from core.UIUserPreference where [Key]='690746db-d1e3-4509-8cc9-e1e36837'
```

Sample OData to fetch records from UIUserPreference table would be as follows.

http://siswebclienturl/ds/views/UserPreferences/CampusNexus.GetUserPreferenceItemByKey(key='690746db-d1e3-4509-8cc9-e1e36837')

The value for the Key will be passed as query string parameter (userpreferenceguid)/

Once this value is fetched, modify the following property values in the request before posting.

*Example*:

```
var payload = value;

payload.StudentCreditCardId = [new credit card id added]

payload.IsNewPaymentSuccessful = true;

payload.IsStudentCreditCardAuthorizeRequired = true;

payload.AuthorizationNumber = [Auth Number from payment provider]
```

Once payment is posted, delete the record from the core.UIUserPreference table.

The Id for the Delete command can be fetched using the following OData query.

http://siswebclienturl/ds/views/UserPreferences/CampusNexus.GetUserPreferenceItemByKey(key='690746db-d1e3-4509-8cc9-e1e36837')

The following URL can be used to delete rows from the UIUserPreference table. For example, pass the Id (3695) that is returned from above URL to the Delete command and use the DELETE verb.

https://siswebclienturl/ds/views/UserPreferences(3695)

Success URL:

http://siswebclienturl/#/students/59655/ledgercard/transactions?paymentsuccess=true

Additionally, Anthology will pass a few more query string parameters based on business functionality and the same should be returned to us to with the return URL (isFromStudentAccountTransaction, isFromProcessPostPayment, IsSaveAndNew, IsApplyPayment,IsPrint, etc.).

For example, we will redirect to the end point with the following query string parameters.

https://[chaseurl]?studentid=59655&userid=2&userpreferenceguid=690746db-d1e3-4509-8cc9-e1e36837&isFromStudentAccountTransaction=true&isFromProcessPostPayment=false&IsSaveAndNew=false&IsApplyPayment=true&IsPrint=false&returnurl=http://siswebclienturl/#/students/59655/ledgercard/transactions

Anthologyexpects the return URL as: http://s-iswebclienturl/#/students/59655/ledgercard/transactions?paymentsuccess=true&isFromStudentAccountTransaction=true&isFromProcessPostPayment=false&IsSaveAndNew=false&IsApplyPayment=true&IsPrint=false

These are required in the UI to load the proper controls.

Cancel URL:

http://siswebclienturl/#/students/59655/ledgercard/transactions

Error URL

http://siswebclienturl/#/students/59655/ledgercard/transactions?errormsg=error

## Make Payment Using Saved Credit Card/Student Bank Account

On this screen, when the user selects the Payment Method as Credit Card/ACH and selects the existing Credit Card/Student Bank Account, the web client service will call the API endpoint (Post URL) that is configured under Configuration →Student Accounts →Electronic Processors (Credit Card Payment Gateway Providers / ACH Payment Gateway Providers). The Funding Page URL is used for redirection and the Post URL is used for posting the payment to the payment gateway provider.

Anthology Student sends a request to post a payment to the custom payment gateway provider. The following parameters are sent in the request.

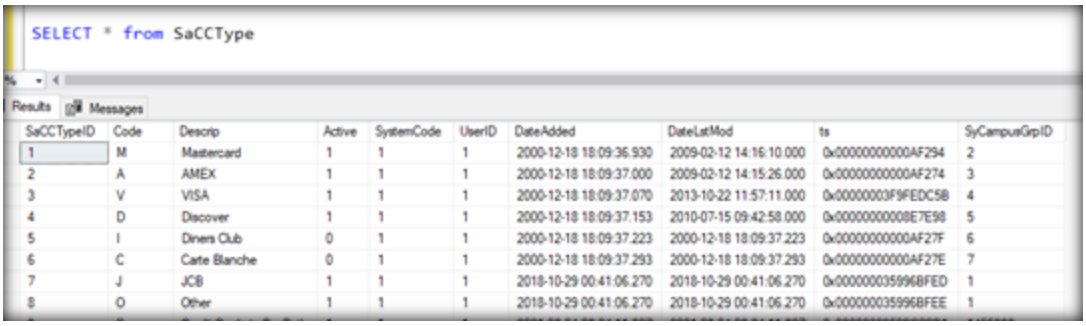
Credit Card Request

```
{
    "IsAutoPay": false,
    "CampusCode": "TRAD",
    "Amount": 100,
    "CreditCardType": "VISA",
    "VaultId": "DP3ECH367", //Reference number
    "CreatedBy": "test user",
    "Payor": {
        "StudentId": "137"
    }
}
```

Bank Account Request

```
{
    "IsAutoPay": false,
    "CampusCode": "TRAD",
    "Amount": 100,
    "AccountType": "Savings",
    "VaultId": "DP3ECH341", //Reference number
    "CreatedBy": "test user",
    "Payor": {
        "StudentID": "1133"
    }
}
```

**Note**: We might have to update the values in the Anthology Student database if the expected CreditCardType/AccountType in the request does not match the values in the Anthology Student database. For example, if the CreditCardType expected in the request is "Visa", but the value in the saCCType table is "V", we need to update the code in the saCCType table to match the request.



```
SELECT * from SaCCType
```

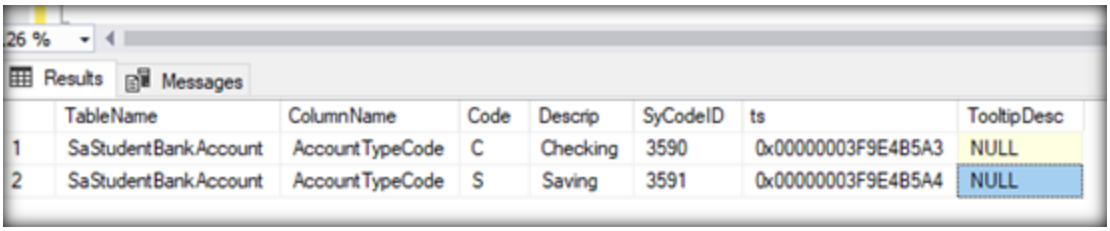
Results | Messages

| SaCCTypeID | Code | Descrip | Active | SystemCode | UserID | DateAdded | DateLstMod | ts | SyCampusGrpID |
|---|---|---|---|---|---|---|---|---|---|
| 1 | M | Mastercard | 1 | 1 | 1 | 2000-12-18 18:09:36.930 | 2009-02-12 14:16:10.000 | 0x00000000000AF294 | 2 |
| 2 | A | AMEX | 1 | 1 | 1 | 2000-12-18 18:09:37.000 | 2009-02-12 14:15:26.000 | 0x00000000000AF274 | 3 |
| 3 | V | VISA | 1 | 1 | 1 | 2000-12-18 18:09:37.070 | 2013-10-22 11:57:11.000 | 0x00000003F9FEDC5B | 4 |
| 4 | D | Discover | 1 | 1 | 1 | 2000-12-18 18:09:37.153 | 2010-07-15 09:42:58.000 | 0x00000000008E7E98 | 5 |
| 5 | I | Diners Club | 0 | 1 | 1 | 2000-12-18 18:09:37.223 | 2000-12-18 18:09:37.223 | 0x00000000000AF27F | 6 |
| 6 | C | Carte Blanche | 0 | 1 | 1 | 2000-12-18 18:09:37.293 | 2000-12-18 18:09:37.293 | 0x00000000000AF27E | 7 |
| 7 | J | JCB | 1 | 1 | 1 | 2018-10-29 00:41:06.270 | 2018-10-29 00:41:06.270 | 0x000000035996BFED | 1 |
| 8 | O | Other | 1 | 1 | 1 | 2018-10-29 00:41:06.270 | 2018-10-29 00:41:06.270 | 0x000000035996BFEE | 1 |

UPDATE SaCCType SET Code = 'VISA' WHERE Code ='V'

If the AccountType does not match, the Code in the SyCode table needs to be updated to match the expected values in the request.

SELECT * FROM SyCode WHERE TableName = 'saStudentBankAccount' and ColumnName = 'AccountTypeCode'



26 % | Results | Messages

| | TableName | ColumnName | Code | Descrip | SyCodeID | ts | TooltipDesc |
|---|---|---|---|---|---|---|---|
| 1 | SaStudentBankAccount | AccountTypeCode | C | Checking | 3590 | 0x00000003F9E4B5A3 | NULL |
| 2 | SaStudentBankAccount | AccountTypeCode | S | Saving | 3591 | 0x00000003F9E4B5A4 | NULL |

UPDATE syCode SET Code = 'Checking' WHERE Code = 'C'

AND TableName = 'saStudentBankAccount' AND ColumnName = 'AccountTypeCode'

In addition, make sure to update the AccountTypeCode column with the same value when the records are inserted into the saStudentBankAccount table.

Anthology expects the following properties in the response when calling the API to post payment.

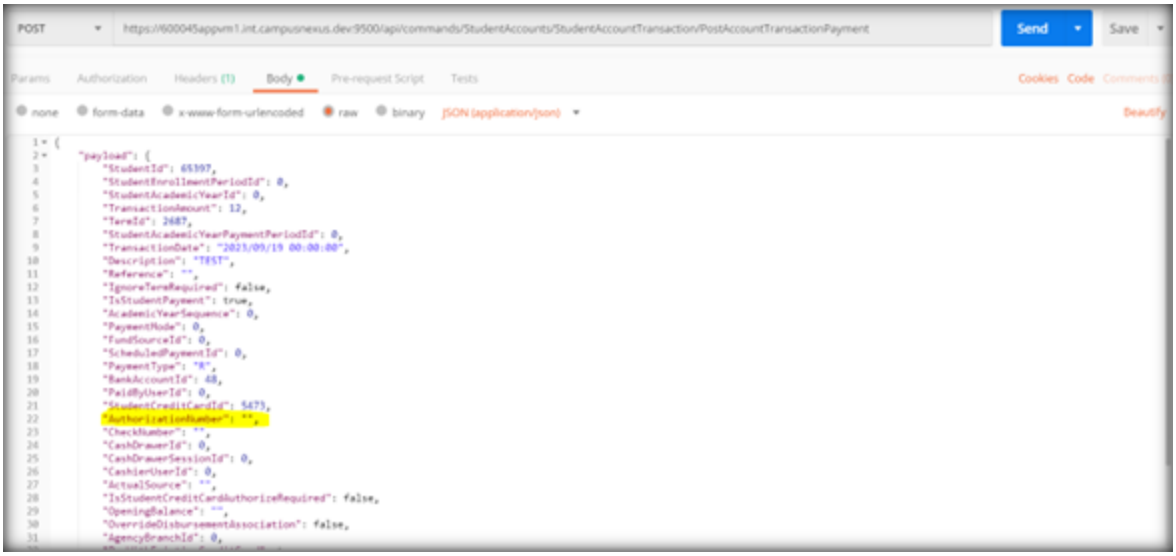
- Success – true/false. Indicates whether the transaction is failure or success.
- AuthCode – [Authorization number returned from payment gateway provider]
- Token – [Token returned from payment gateway provider]
- Messages – If failure, then this should contain the error message.
- Error Type – Provide any other information related to the error, if available, or Empty.

```
{
    "success": true,
    "errorType": "",
    "authCode": "TEST574",
    "token": "TESTVISA4242",
    "messages": [
        "Approved",
    ]
}
```

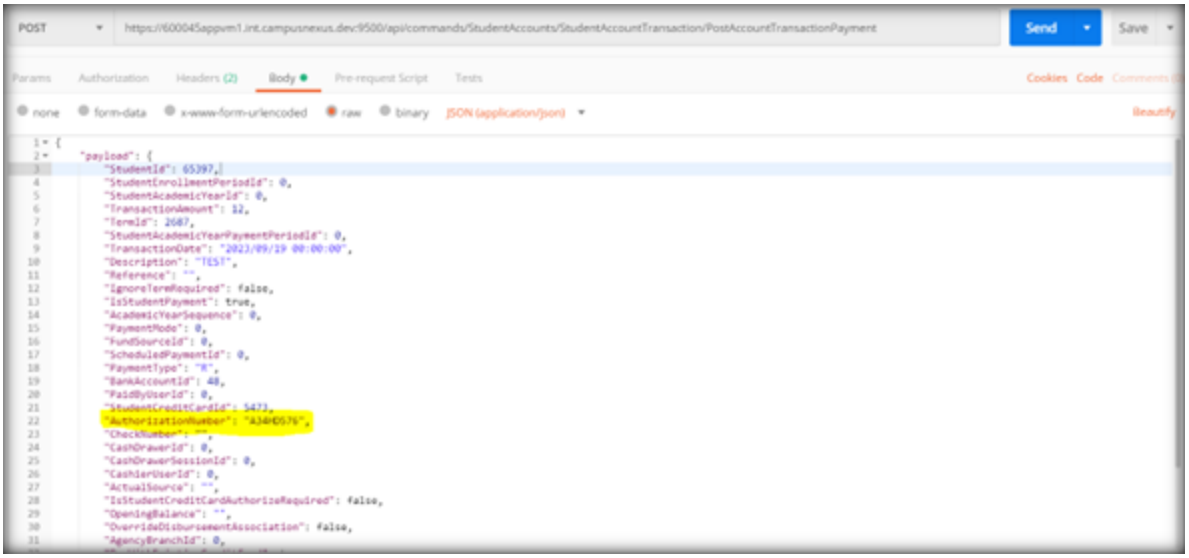
## Post Credit Card/ACH Payments for Reference Only

Clients can use the command API (api/-commands/StudentAccounts/StudentAccountTransaction/PostAccountTransactionPayment) to post Credit Card/ACH transactions directly to the ledger card without sending them to payment gateway provider to be processed. This can be done by populating the AuthorizationNumber in the payload. So, if the Authorization Number in the payload is blank, the transaction will be sent to the custom payment gateway provider; but if it has a value, the request will not be sent to the custom payment gateway provider and will be posted to the ledger card only.

For example, the following request will be sent to custom payment gateway provider.




The following request will not be sent to the custom payment gateway provider because the AuthorizationNumber is available.



# Resources

## All Products

Anthology Developer Documentation:

- https://docs.anthology.com/

Anthology Developer Portal

- https://developer.anthology.com/

## Anthology Student

Anthology Student APIs and Entities: (MyCampusInsight login required)

- Service Catalog
- Object Library
- REST APIs

## Anthology Reach

- Anthology Reach Help

Refer to the Product Information site for additional documentation and training resources.