Integration Flow

# Flow Breakdown

## Step Here’s the integration flow documentation for the AIS project based on the translations and functions detailed previously. This documentation will cover data flow, system interactions, transformations, and external calls. --- ## AIS Integration Flow Documentation ### Overview This document outlines the integration flows designed to interact with Salesforce for group management and Chatter feed operations. The flows include retrieving group information, posting to Chatter feeds, updating feed items, and generating access tokens for authentication. ### Flow 1: Group-GetByName \*\*Purpose:\*\* Retrieve the Salesforce group ID based on the group's name. #### Data Flow: 1. \*\*Trigger:\*\* The flow starts when an HTTP GET request is made with a query parameter `groupName`. 2. \*\*Input Schema:\*\* ```json { "groupName": "Marketing" } ``` 3. \*\*External Call:\*\* - \*\*Service:\*\* Salesforce SOQL Query - \*\*Endpoint:\*\* `/services/data/vXX.X/query` - \*\*Query:\*\* ```sql SELECT ID FROM CollaborationGroup WHERE Name = ':groupName' ``` 4. \*\*Output Schema:\*\* - On Success: ```json { "id": "0F9XXXXXXXXXXXXXXX" } ``` - On Not Found: ```http HTTP Status: 404 Not Found ``` --- ### Flow 2: Chatter-FeedItem-Post \*\*Purpose:\*\* Post a new Chatter message to a specified record. #### Data Flow: 1. \*\*Trigger:\*\* The flow begins when an HTTP POST request containing the Chatter message is received. 2. \*\*Input Schema:\*\* ```json { "messagePieces": [ {"type": "Text", "text": "Hello, World!"} ], "recordIDToAddFeedItemTo": "001XXXXXXXXXXXXXXX" } ``` 3. \*\*Transformations:\*\* - Prepare the message for the Chatter API format. - Check if a similar post exists by querying existing feed items. 4. \*\*External Call:\*\* - \*\*Service:\*\* Salesforce Chatter API - \*\*Endpoint:\*\* `POST /services/data/v47.0/chatter/feed-elements` - \*\*Request Body:\*\* ```json { "body": { "messageSegments": [] }, "feedElementType": "FeedItem", "subjectId": "001XXXXXXXXXXXXXXX" } ``` 5. \*\*Output Schema:\*\* - On Success (Created): ```http HTTP Status: 201 Created ``` - On Conflict (Already Exists): ```http HTTP Status: 409 Conflict ``` --- ### Flow 3: Chatter-FeedItem-Put \*\*Purpose:\*\* Update an existing Chatter message or handle existing posts. #### Data Flow: 1. \*\*Trigger:\*\* The flow starts with an HTTP PUT request aimed at potentially updating an existing Chatter post. 2. \*\*Input Schema:\*\* ```json { "messagePieces": [ {"type": "Text", "text": "Updated Message!"} ], "recordIDToAddFeedItemTo": "001XXXXXXXXXXXXXXX" } ``` 3. \*\*Transformations:\*\* - Check if a similar post already exists by querying the FeedItem. 4. \*\*External Call:\*\* - If no existing post is found, proceed to post a new message. - \*\*Service:\*\* Salesforce Chatter API - \*\*Endpoint:\*\* `POST /services/data/v47.0/chatter/feed-elements` - \*\*Request Body:\*\* ```json { "body": { "messageSegments": [] }, "feedElementType": "FeedItem", "subjectId": "001XXXXXXXXXXXXXXX" } ``` 5. \*\*Output Schema:\*\* - On New Creation: ```http HTTP Status: 201 Created ``` - If Already Exists: ```http HTTP Status: 200 OK ``` --- ### Flow 4: GenerateChatterAccessToken \*\*Purpose:\*\* Obtain an OAuth access token for Salesforce Chatter API. #### Data Flow: 1. \*\*Trigger:\*\* Initiated by an HTTP POST request to generate an access token. 2. \*\*Input Schema:\*\* ```json { "client\_id": "your\_client\_id", "client\_secret": "your\_client\_secret", "username": "your\_username", "password": "your\_password" } ``` 3. \*\*External Call:\*\* - \*\*Service:\*\* Salesforce OAuth Endpoint - \*\*Endpoint:\*\* `POST /services/oauth2/token` - \*\*Request Body (formatted as multipart/form-data):\*\* ```plaintext grant\_type=password client\_id=<client\_id> client\_secret=<client\_secret> username=<username> password=<password> ``` 4. \*\*Output Schema:\*\* - Successful Token Retrieval: ```json { "access\_token": "00DXXXXXXXXXXXX!AQoXXXXXXXXXXXXXXX" } ``` --- ### Summary The integration flows are designed with clear separations of concerns, enabling easy maintenance and scalability. Each flow is self-contained, specifies input and output formats, and documents interactions with Salesforce APIs. This architecture provides a robust solution for managing group information and Chatter interactions while adhering to best practices for Azure Integration Services. If you need further assistance or modifications to this documentation, feel free to ask!

Group-GetByName  
  
\*\*Purpose:\*\* Retrieve the Salesforce group ID based on the group's name.  
  
#### Data Flow:  
1. \*\*Trigger:\*\* The flow starts when an HTTP GET request is made with a query parameter `groupName`.  
2. \*\*Input Schema:\*\*  
 ```json  
 {  
 "groupName": "Marketing"  
 }  
 ```  
  
3. \*\*External Call:\*\*  
 - \*\*Service:\*\* Salesforce SOQL Query  
 - \*\*Endpoint:\*\* `/services/data/vXX.X/query`  
 - \*\*Query:\*\*  
 ```sql  
 SELECT ID FROM CollaborationGroup WHERE Name = ':groupName'  
 ```  
  
4. \*\*Output Schema:\*\*  
 - On Success:  
 ```json  
 {  
 "id": "0F9XXXXXXXXXXXXXXX"  
 }  
 ```  
 - On Not Found:  
 ```http  
 HTTP Status: 404 Not Found  
 ```  
  
---  
  
### Flow 2: Chatter-FeedItem-Post  
  
\*\*Purpose:\*\* Post a new Chatter message to a specified record.  
  
#### Data Flow:  
1. \*\*Trigger:\*\* The flow begins when an HTTP POST request containing the Chatter message is received.  
  
2. \*\*Input Schema:\*\*  
 ```json  
 {  
 "messagePieces": [  
 {"type": "Text", "text": "Hello, World!"}  
 ],  
 "recordIDToAddFeedItemTo": "001XXXXXXXXXXXXXXX"  
 }  
 ```  
  
3. \*\*Transformations:\*\*  
 - Prepare the message for the Chatter API format.  
 - Check if a similar post exists by querying existing feed items.  
  
4. \*\*External Call:\*\*  
 - \*\*Service:\*\* Salesforce Chatter API  
 - \*\*Endpoint:\*\* `POST /services/data/v47.0/chatter/feed-elements`  
 - \*\*Request Body:\*\*  
 ```json  
 {  
 "body": {  
 "messageSegments": []  
 },  
 "feedElementType": "FeedItem",  
 "subjectId": "001XXXXXXXXXXXXXXX"  
 }  
 ```  
  
5. \*\*Output Schema:\*\*  
 - On Success (Created):  
 ```http  
 HTTP Status: 201 Created  
 ```  
 - On Conflict (Already Exists):  
 ```http  
 HTTP Status: 409 Conflict  
 ```  
  
---  
  
### Flow 3: Chatter-FeedItem-Put  
  
\*\*Purpose:\*\* Update an existing Chatter message or handle existing posts.  
  
#### Data Flow:  
1. \*\*Trigger:\*\* The flow starts with an HTTP PUT request aimed at potentially updating an existing Chatter post.  
  
2. \*\*Input Schema:\*\*  
 ```json  
 {  
 "messagePieces": [  
 {"type": "Text", "text": "Updated Message!"}  
 ],  
 "recordIDToAddFeedItemTo": "001XXXXXXXXXXXXXXX"  
 }  
 ```  
  
3. \*\*Transformations:\*\*  
 - Check if a similar post already exists by querying the FeedItem.  
  
4. \*\*External Call:\*\*  
 - If no existing post is found, proceed to post a new message.  
 - \*\*Service:\*\* Salesforce Chatter API  
 - \*\*Endpoint:\*\* `POST /services/data/v47.0/chatter/feed-elements`  
 - \*\*Request Body:\*\*  
 ```json  
 {  
 "body": {  
 "messageSegments": []  
 },  
 "feedElementType": "FeedItem",  
 "subjectId": "001XXXXXXXXXXXXXXX"  
 }  
 ```  
  
5. \*\*Output Schema:\*\*  
 - On New Creation:  
 ```http  
 HTTP Status: 201 Created  
 ```  
 - If Already Exists:  
 ```http  
 HTTP Status: 200 OK  
 ```  
  
---  
  
### Flow 4: GenerateChatterAccessToken  
  
\*\*Purpose:\*\* Obtain an OAuth access token for Salesforce Chatter API.  
  
#### Data Flow:  
1. \*\*Trigger:\*\* Initiated by an HTTP POST request to generate an access token.  
  
2. \*\*Input Schema:\*\*  
 ```json  
 {  
 "client\_id": "your\_client\_id",  
 "client\_secret": "your\_client\_secret",  
 "username": "your\_username",  
 "password": "your\_password"  
 }  
 ```  
  
3. \*\*External Call:\*\*  
 - \*\*Service:\*\* Salesforce OAuth Endpoint  
 - \*\*Endpoint:\*\* `POST /services/oauth2/token`  
 - \*\*Request Body (formatted as multipart/form-data):\*\*  
 ```plaintext  
 grant\_type=password  
 client\_id=<client\_id>  
 client\_secret=<client\_secret>  
 username=<username>  
 password=<password>  
 ```  
  
4. \*\*Output Schema:\*\*  
 - Successful Token Retrieval:  
 ```json  
 {  
 "access\_token": "00DXXXXXXXXXXXX!AQoXXXXXXXXXXXXXXX"  
 }  
 ```  
  
---  
  
### Summary  
The integration flows are designed with clear separations of concerns, enabling easy maintenance and scalability. Each flow is self-contained, specifies input and output formats, and documents interactions with Salesforce APIs.  
  
This architecture provides a robust solution for managing group information and Chatter interactions while adhering to best practices for Azure Integration Services.  
  
If you need further assistance or modifications to this documentation, feel free to ask!