Wellington Global Power Platform Bootcamp 2023

# End to end scenario

We are implementing an end-to-end service that takes a conversation recording in WAV passes it to a Azure service to transcribe it and store the returned sentences in Dataverse.

For each of the sentences, we will run an AI Builder model to detect the sentiment and for the general conversation, we will run ChatGPT to detect sentiment and reason for the sentiment.

Finally, we will display the sentiment results in a Canvas app.

## What is out of scope

Capturing the recording and formatting correctly to work with Azure cognitive services.

Creating an Azure function that accepts a link to the recording and transcribes it.

The resources above will be provided.

# Instance details

## App URL

<https://gppb23.crm6.dynamics.com/main.aspx?appid=9798ad91-d2ad-ed11-83ff-0022489804cd>

## Usernames

[abhay@SamplePages.onmicrosoft.com](mailto:abhay@SamplePages.onmicrosoft.com)

[ak@SamplePages.onmicrosoft.com](mailto:ak@SamplePages.onmicrosoft.com)

[linn@samplepages.onmicrosoft.com](mailto:linn@samplepages.onmicrosoft.com)

[mac@SamplePages.onmicrosoft.com](mailto:mac@SamplePages.onmicrosoft.com)

[user1@samplepages.onmicrosoft.com](mailto:user1@samplepages.onmicrosoft.com)

…

[user18@samplepages.onmicrosoft.com](mailto:user18@samplepages.onmicrosoft.com)

**password** Gppbnz23

# Session 1 – Create a data model and a model driven app in Dataverse

**Rami**

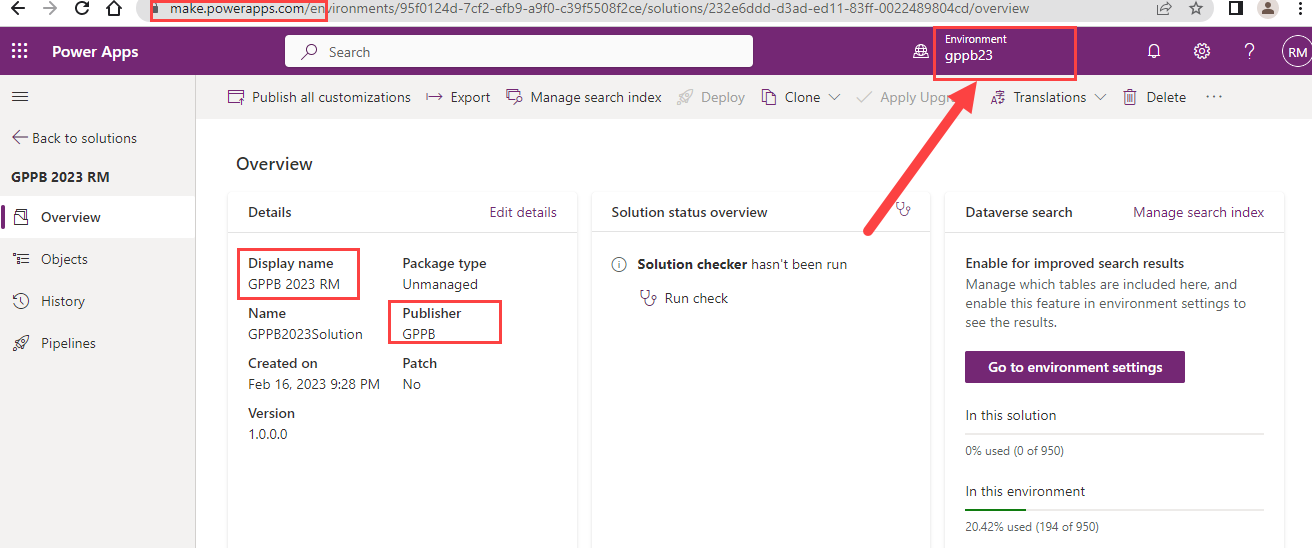
## Goals

1. Create a new solution
2. Create a data structure
3. Create a model-driven app to display the results



## Solution

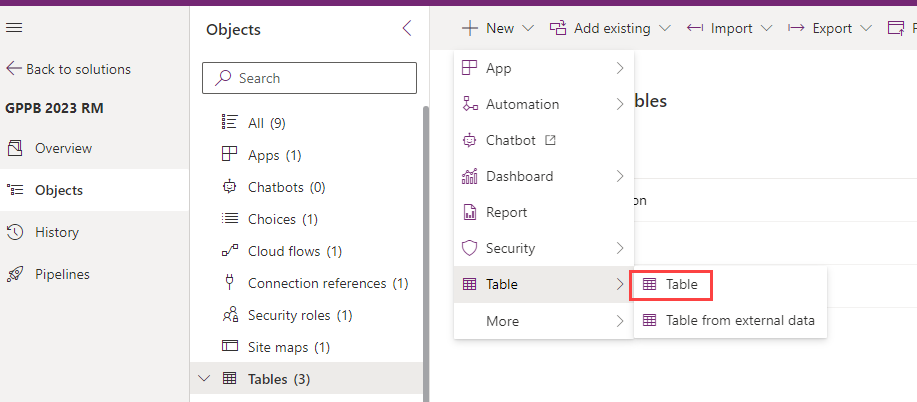
* Navigate to make.powerapps.com
* Login with one of the user accounts
* Ensure you are using gppb23 not the default app
* Create a new solution with your initials (e.g. gppb23\_rm) select gppb as the publisher



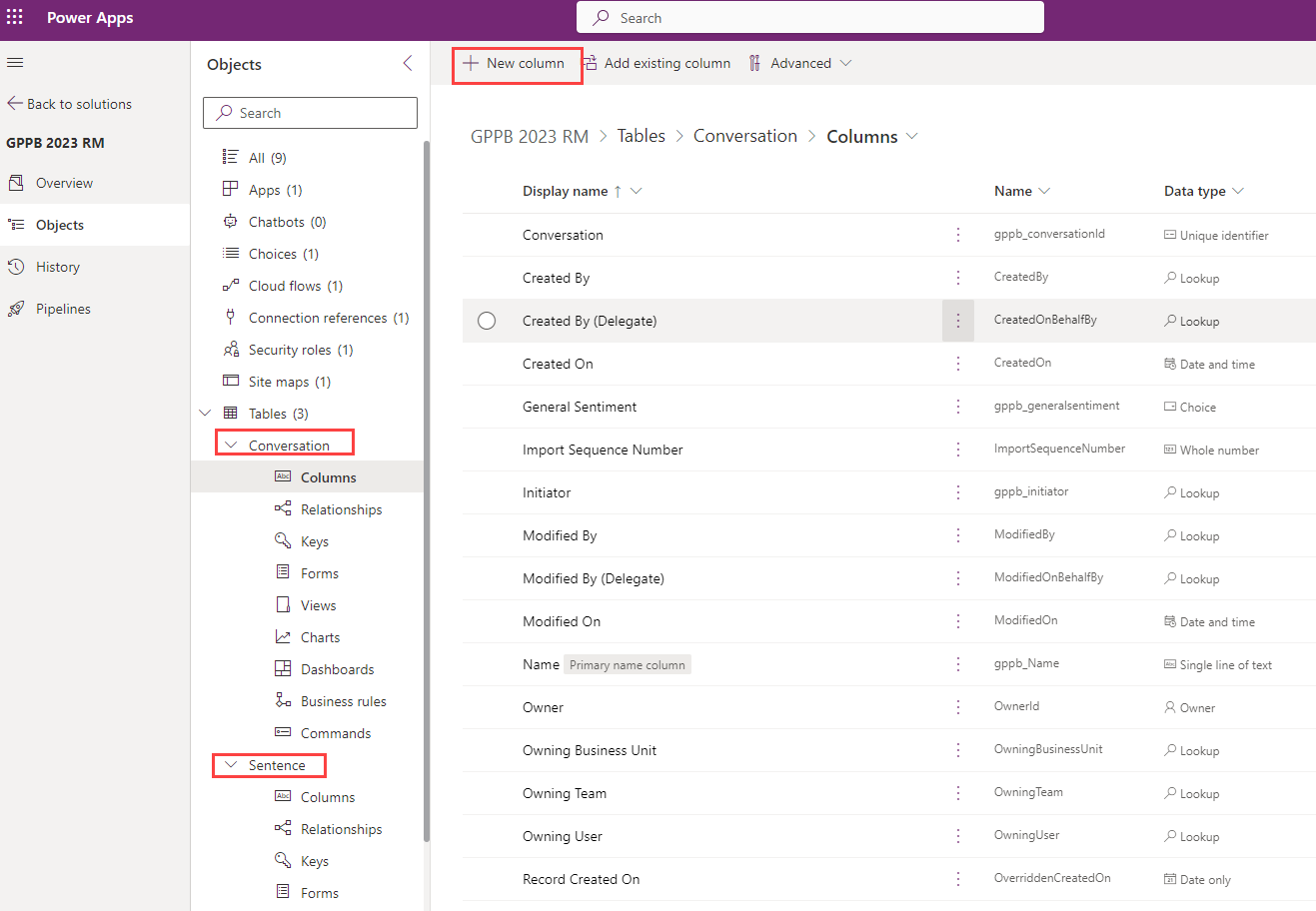
## Data Model

Build the schema for the following tables

* Conversation
* Sentence with a lookup to conversations



### Create new columns



### Conversation

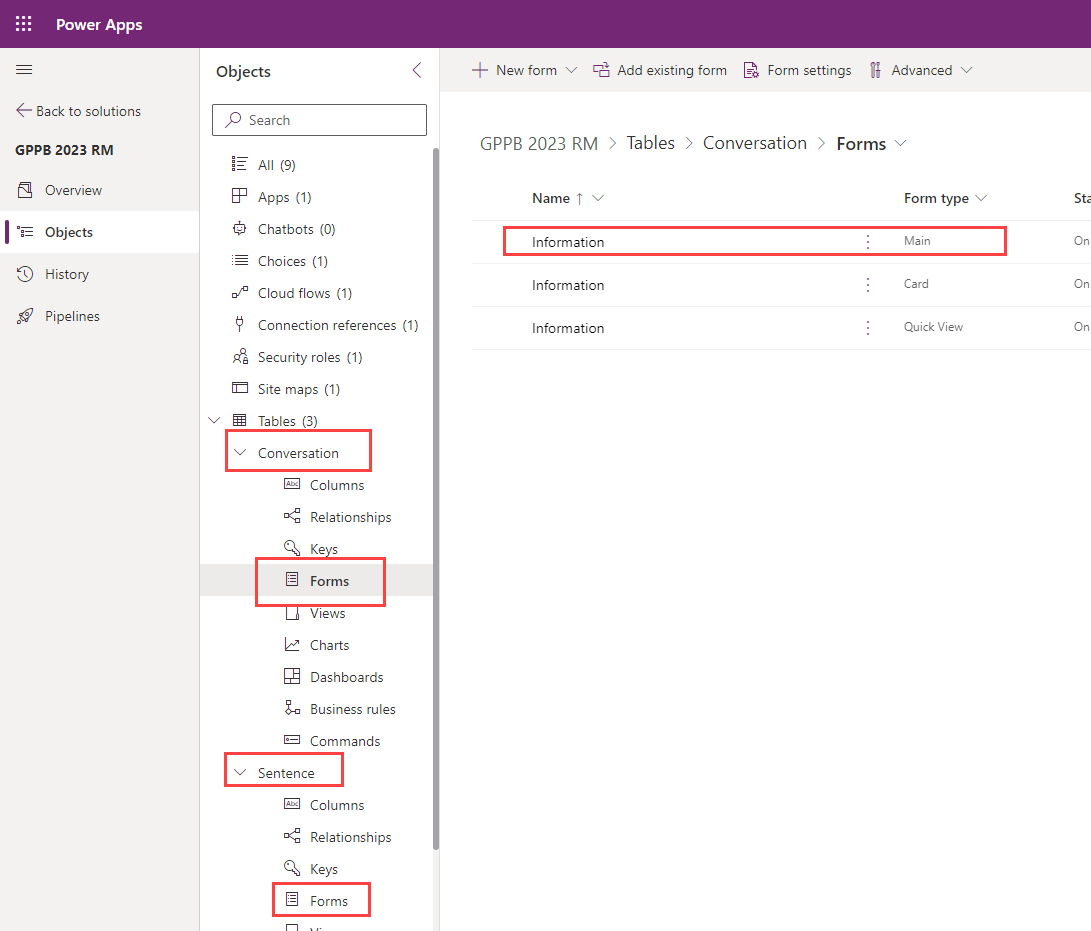
|  |  |
| --- | --- |
| Field Name | Data Type |
| Start Time | Date Time |
| Recording | File (32MB) |
| Name | Text |
| Sentiment | Global Drop Down (positive, negative, neutral) |
| Sentiment Text | Text 100 |
| Sentiment Analysis | Text 4000 |

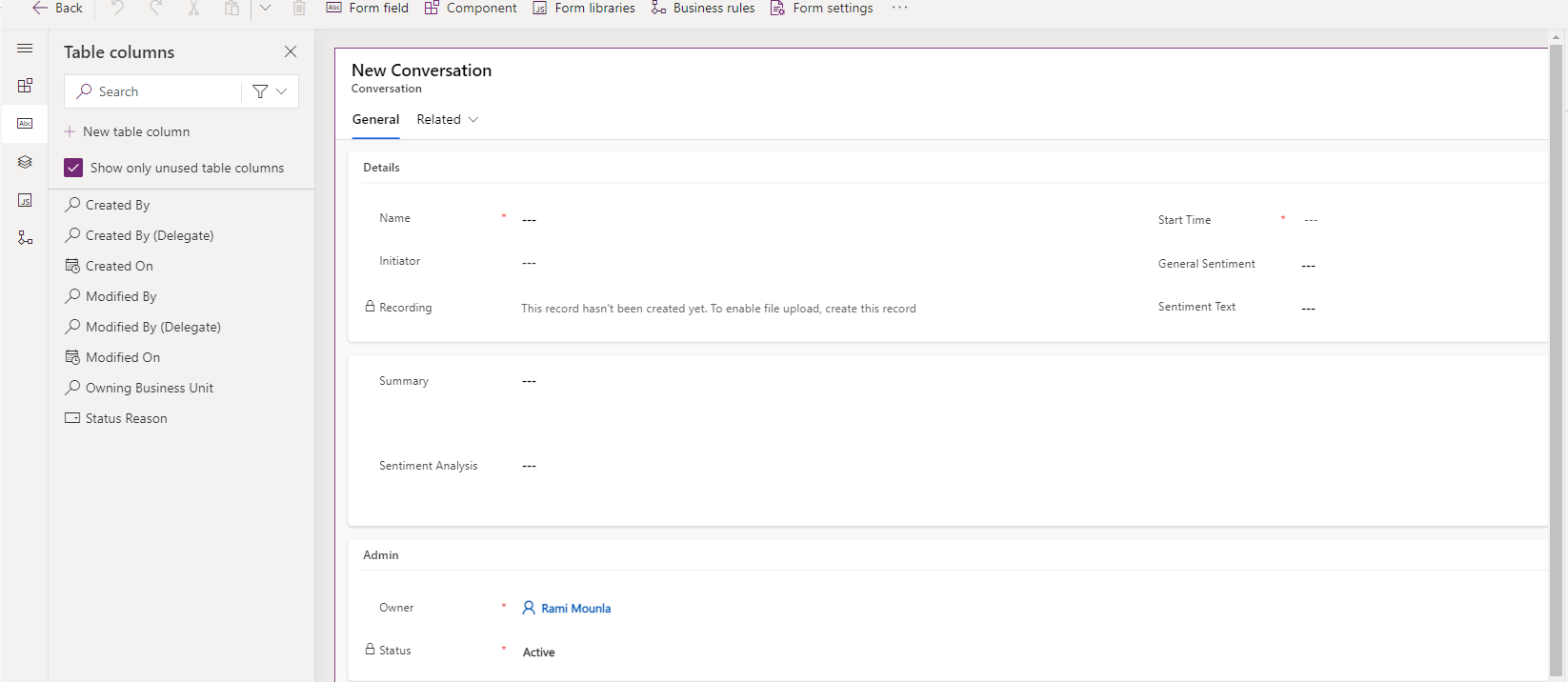
### Sentence

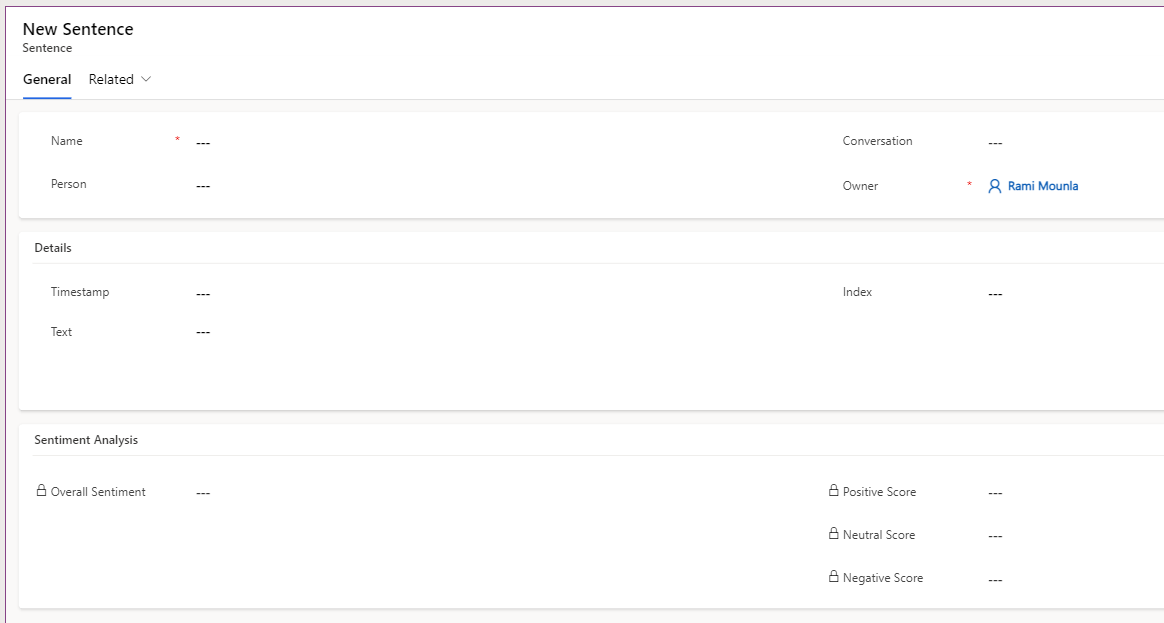
|  |  |
| --- | --- |
| Field Name | Data Type |
| Name | Text 100 |
| Text | Text 4000 |
| Sentiment | Global Drop Down (positive, negative, neutral) |
| Conversation | Lookup |
| Person | Text |
| StartTime | Number Int |
| Negative Score | Decimal |
| Neutral Score | Decimal |
| Positive Score | Decimal |
| Overall Sentiment | Text 100 |

### Configure forms and views

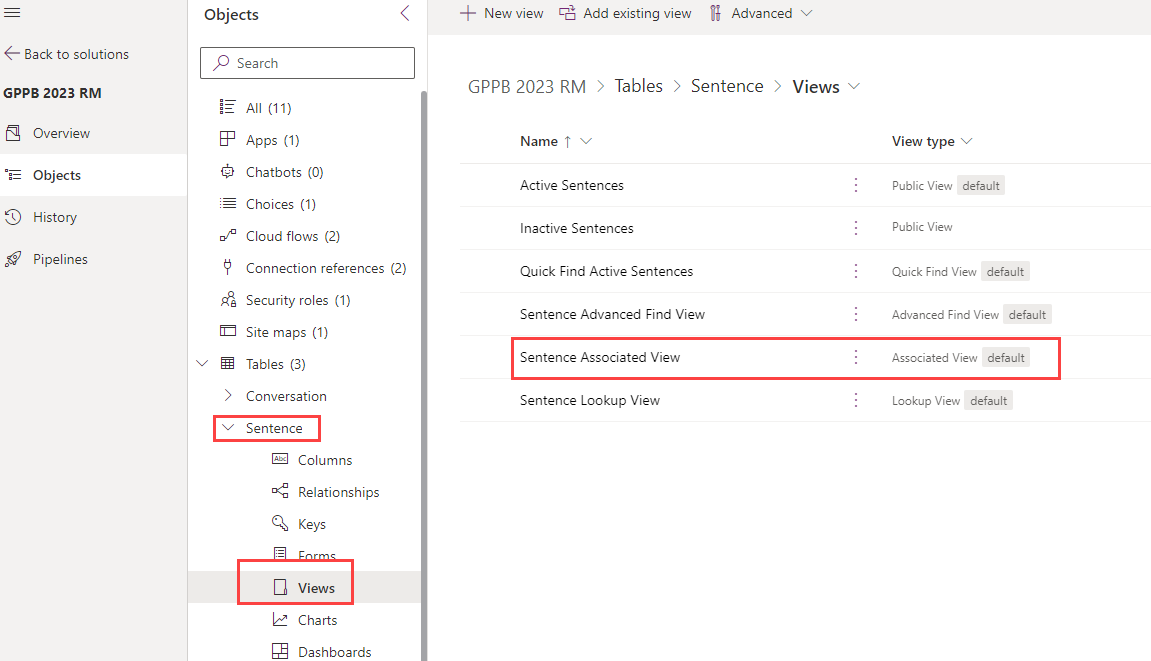
* Configure conversation and sentences forms

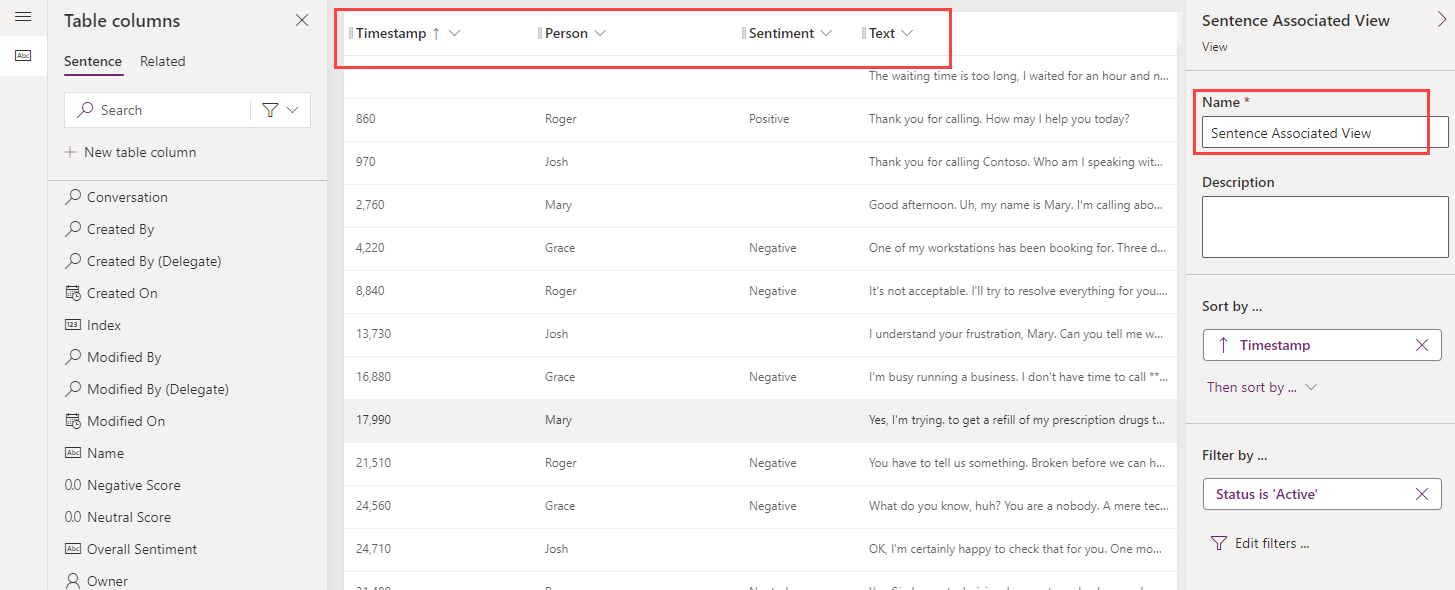






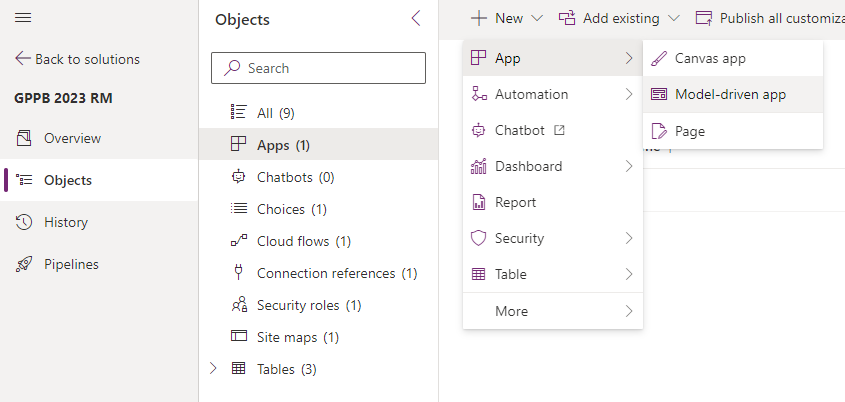
* Configure the sentences related view

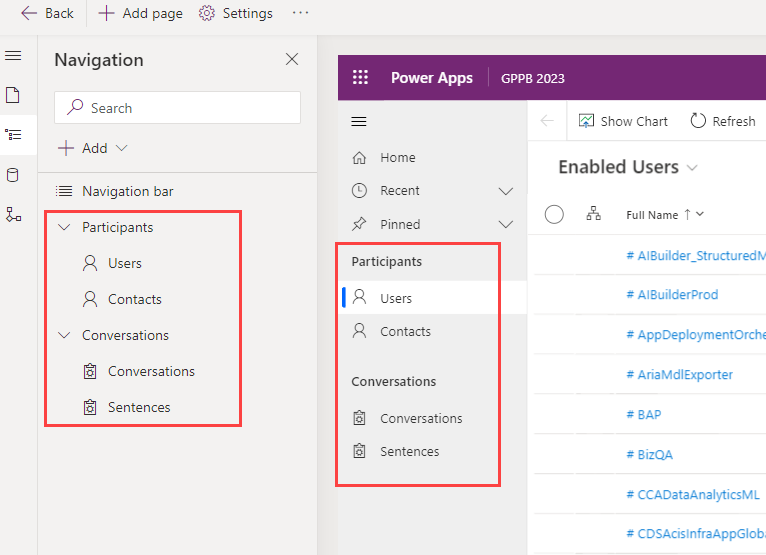




## Model Driven App

* Create the app with conversation and sentences
* Give the app a name with your initials





# Session 2 – Create an AI Builder model to detect sentiment

**Mary Ann Castro**

## AI Builder

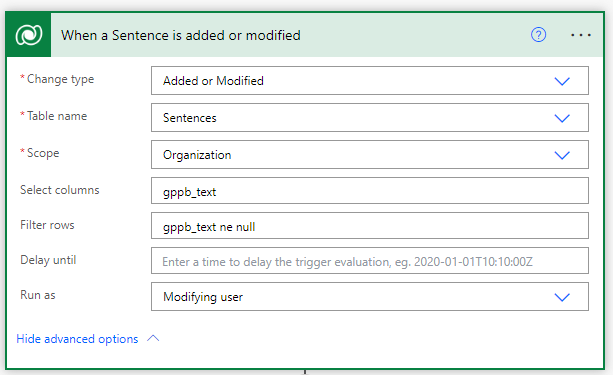
Go through the basics, create AI model that trigger on create of a sentence, passes the text and return the sentiment.

## Use the sentiment analysis prebuilt model in Power Automate

1. Login to <https://make.powerapps.com/> and click Solutions
2. Open the solution you are working on and click Cloud Flows
3. Select New 🡪 Automation 🡪 Cloud Flow 🡪 Automated

**Graphical user interface, text, application

Description automatically generated**

1. Provide a flow name (i.e. Analyze sentiment of a sentence using AI Builder) and choose the Dataverse trigger “When a row is added, modified or deleted”.
2. Add the details as per screenshot below  
   
3. Click +New Step and look for AI Builder. Select “Analyze positive or negative sentiment in text” and add the “Text” column from the trigger output.  
     
   Graphical user interface, text, application, email

   Description automatically generated  
   Graphical user interface, application

   Description automatically generated
4. Click +New Step and click Dataverse. Select “Update a row” and add the following output from “Analyze positive or negative sentiment” step as per screenshot below.
   * Negative Score = Probability overall text is negative
   * Neutral Score = Probability overall text is neutral
   * Positive Score = Probability overall text is positive
   * Overall Sentiment = Overall text sentiment

Graphical user interface, text, application, email

Description automatically generated

1. Save the flow and test.

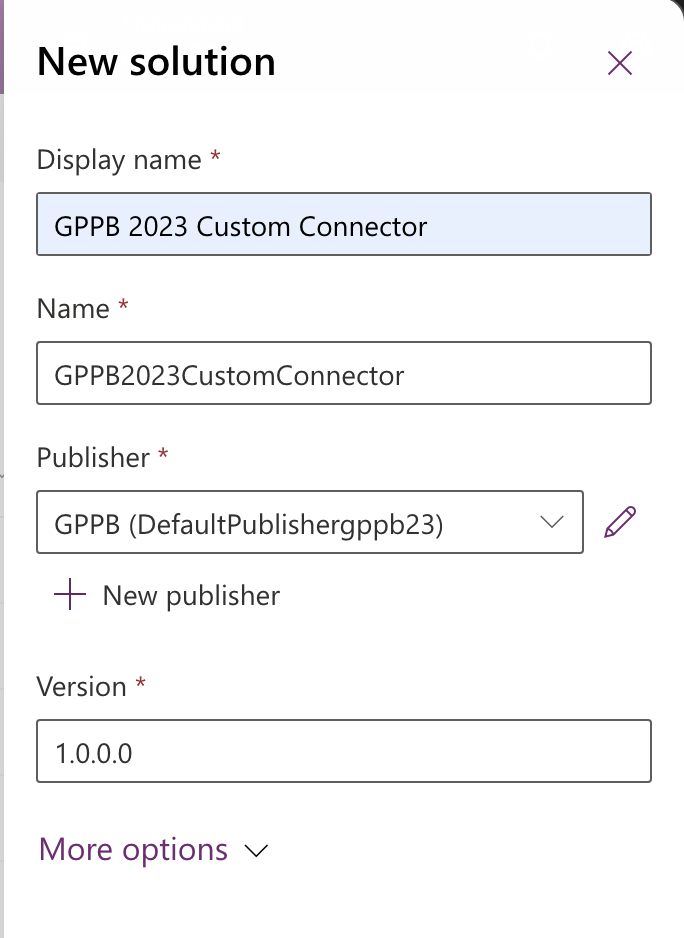
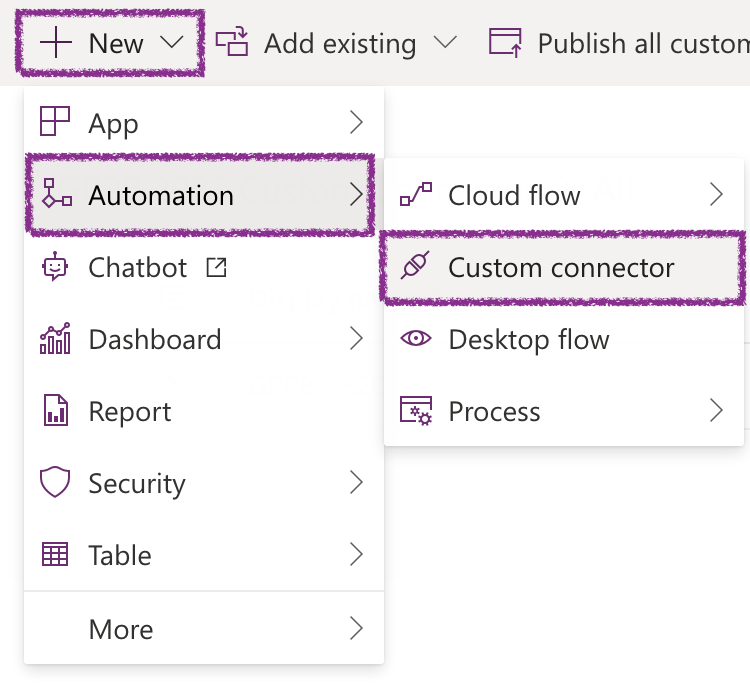
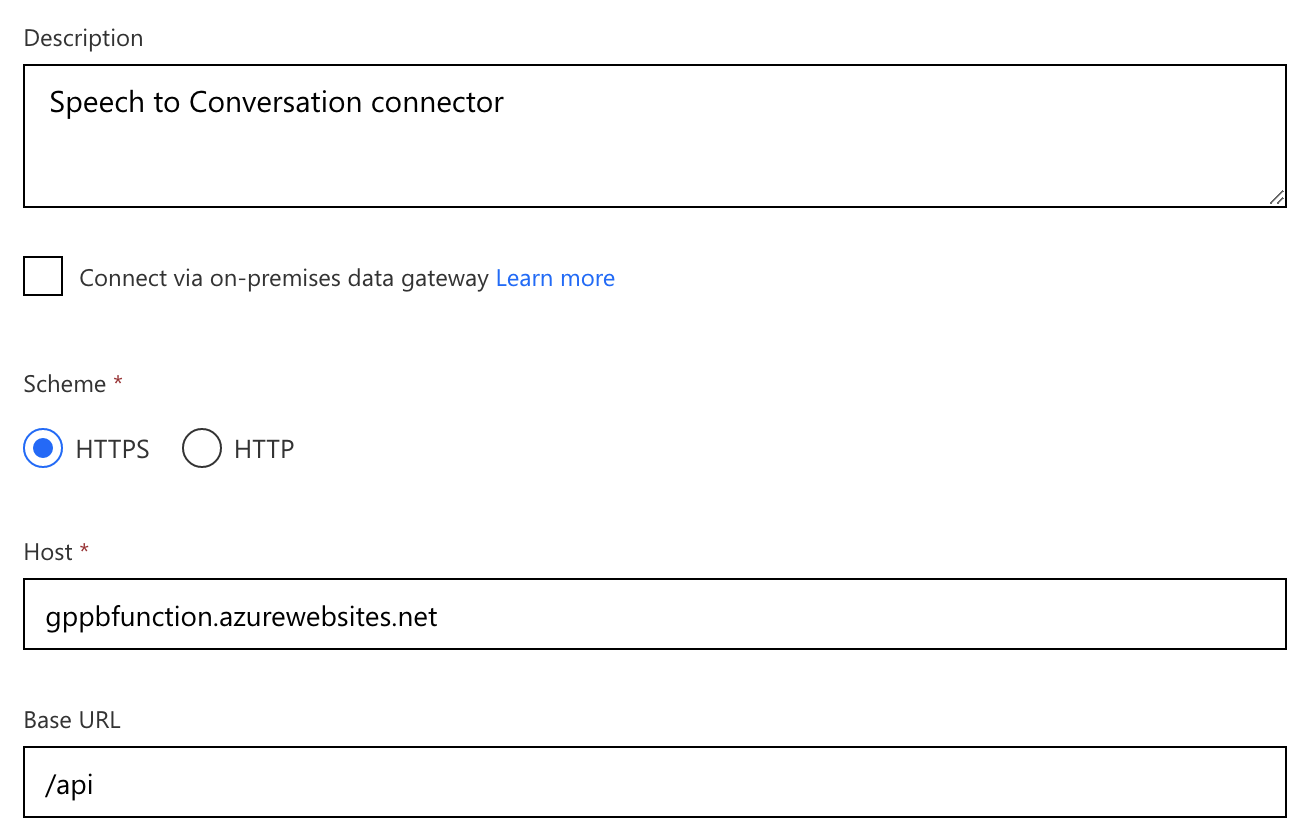
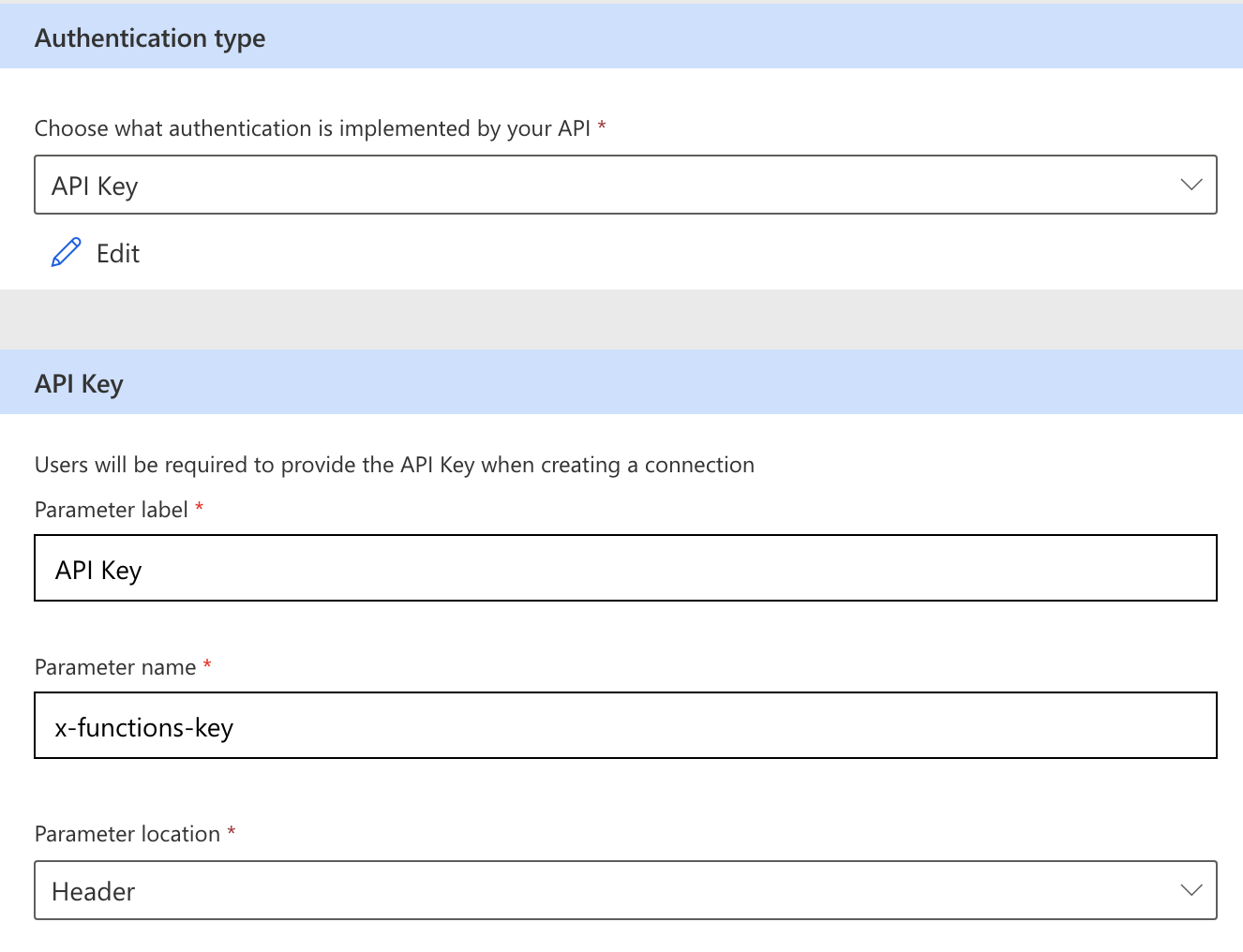
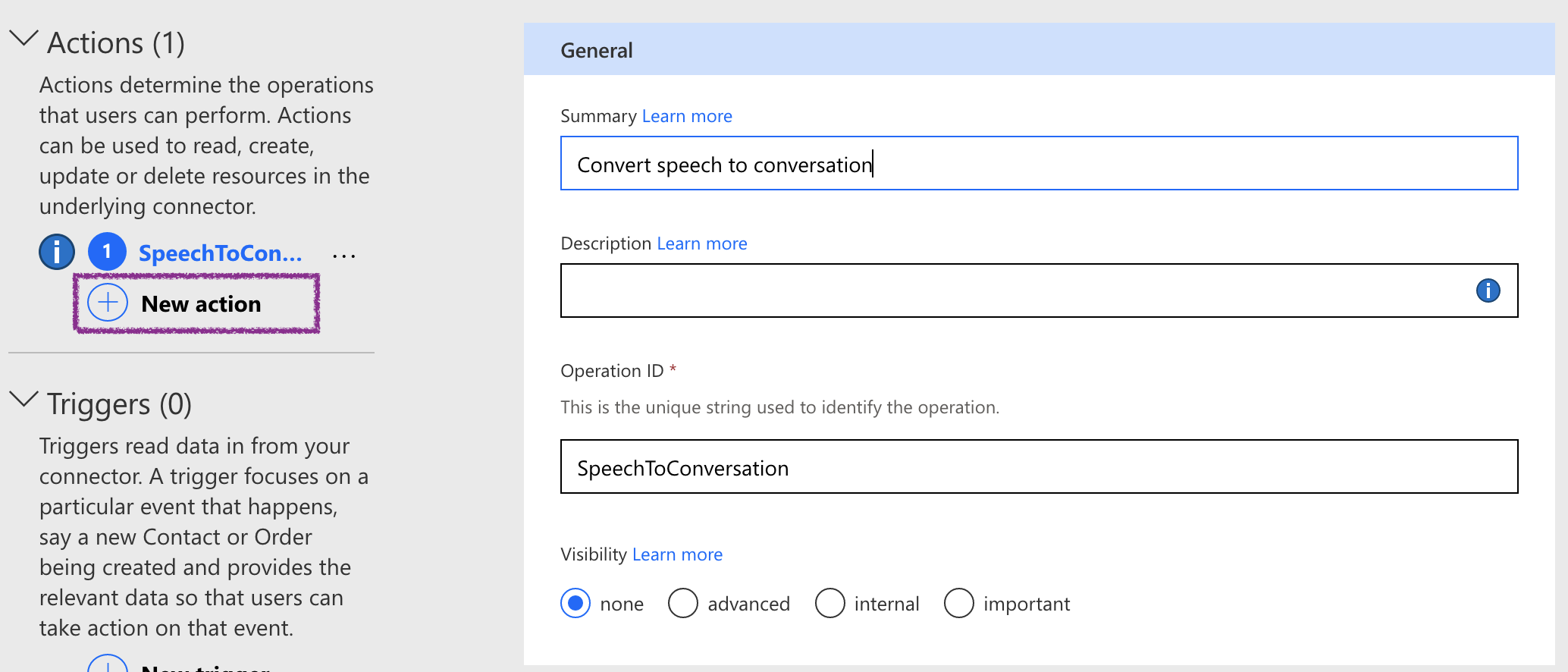
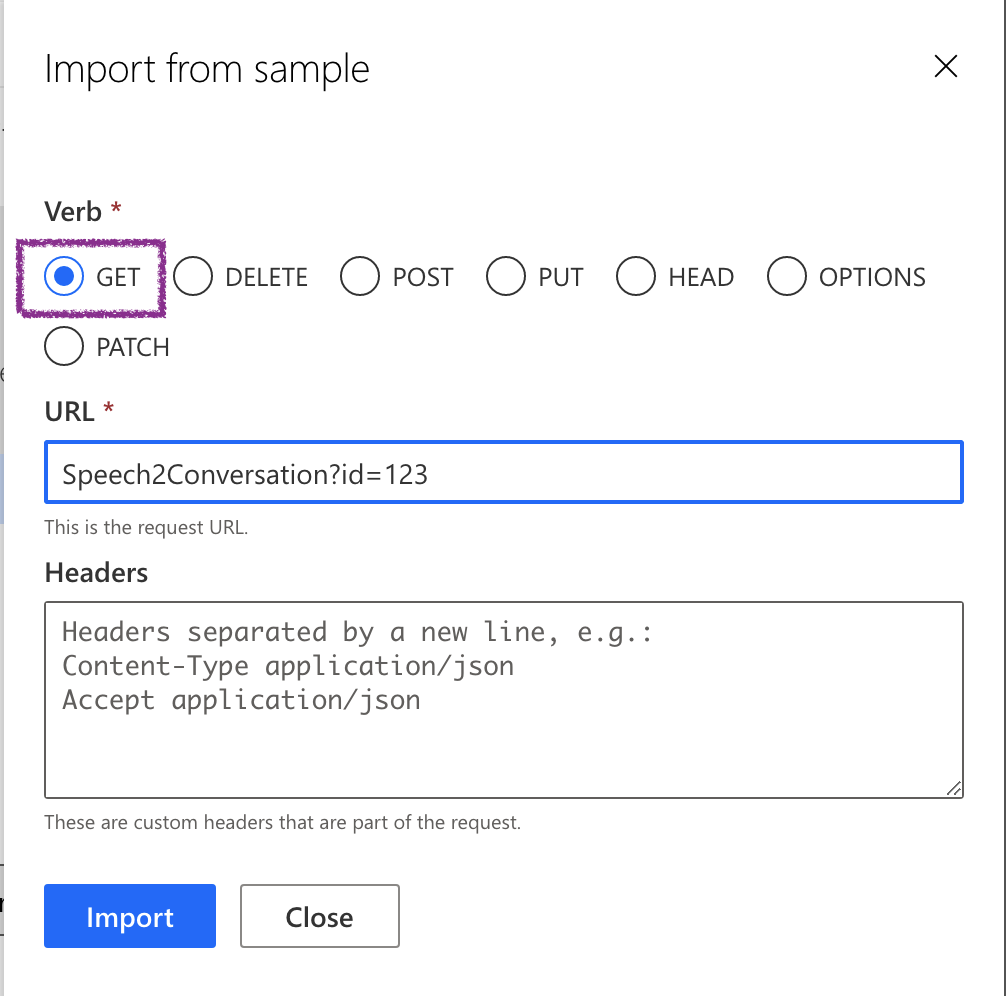
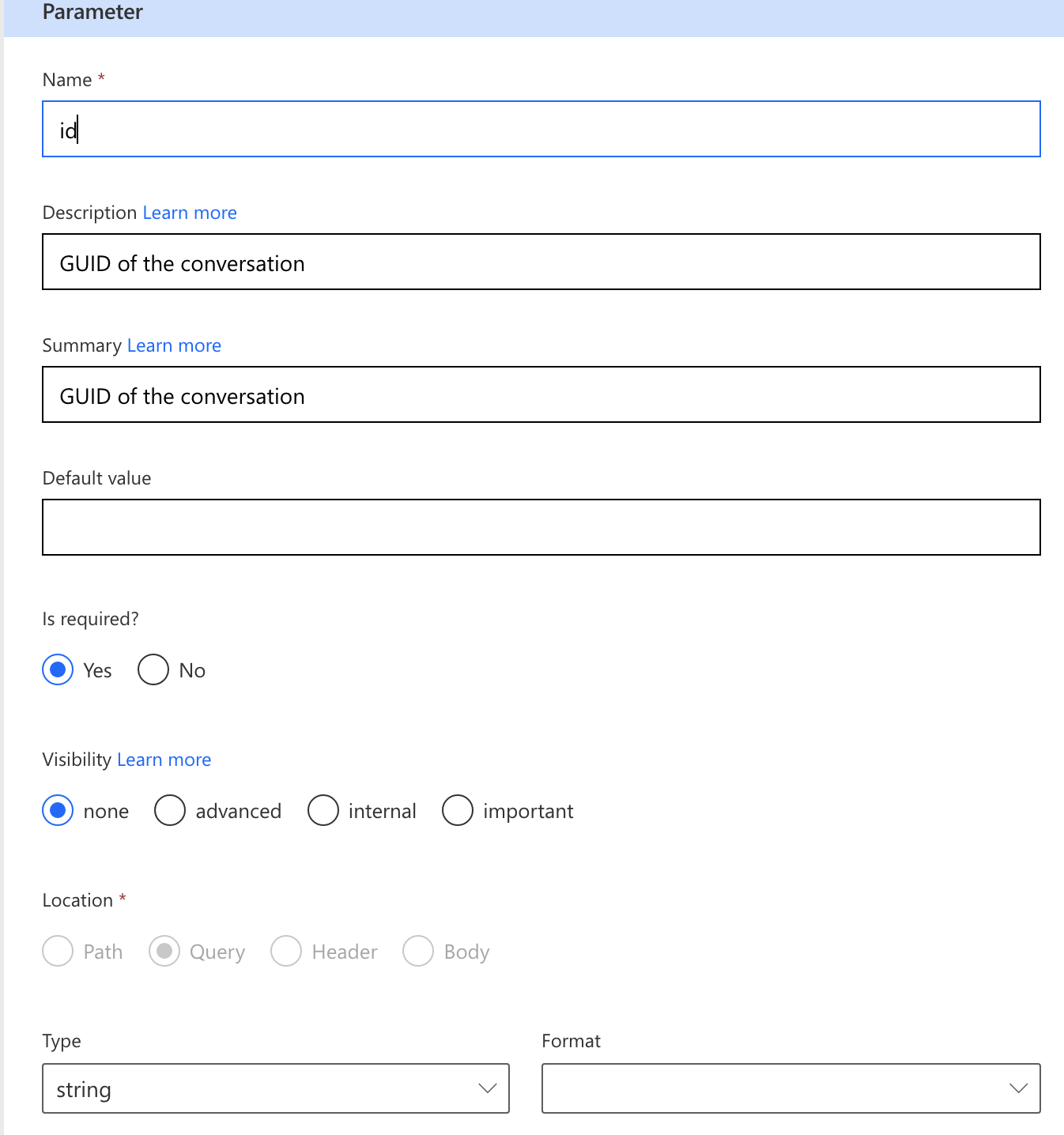
# Session 3 - Create a custom connector

**Aung Khaing**

Connector to Azure function (pre built) using authentication type (?)

Send binary (or GUID and pull the binary if it’s easier) return the sentiment.

## A basic custom connector: A step by step manual process

1. Go to <https://make.powerapps.com>. Ensure you are in the correct environment.
2. Create a new solution if there is no pre-existed solution for a custom connector.  
   
3. Create a new custom connector.  
   
4. Enter the description, host and base urls:  
   
5. The external API hosted in Azure Function Apps uses authentication key. The API key needs to be in the header.  
   Select **API Key** as authentication type. Parameter name should be same as the header which is accepted by the API.  
   
6. Add a new action. Enter summary and operation id. Note: if you have multiple actions, operation id must be unique.  
   
7. Add a request sample. Url is Speech2Conversation?id=GUID  
   Note: the external API currently accepts GET and POST. We need to accept **id** as a query string which is the GUID of a conversation record in Dataverse.  
   
8. You can fine tune the query string **id** properties such as required, visibility and type.   
   
9. Add a default response to the operation. This makes a custom connector more user friendly.   
   Note: Please use the following as body:  
   [

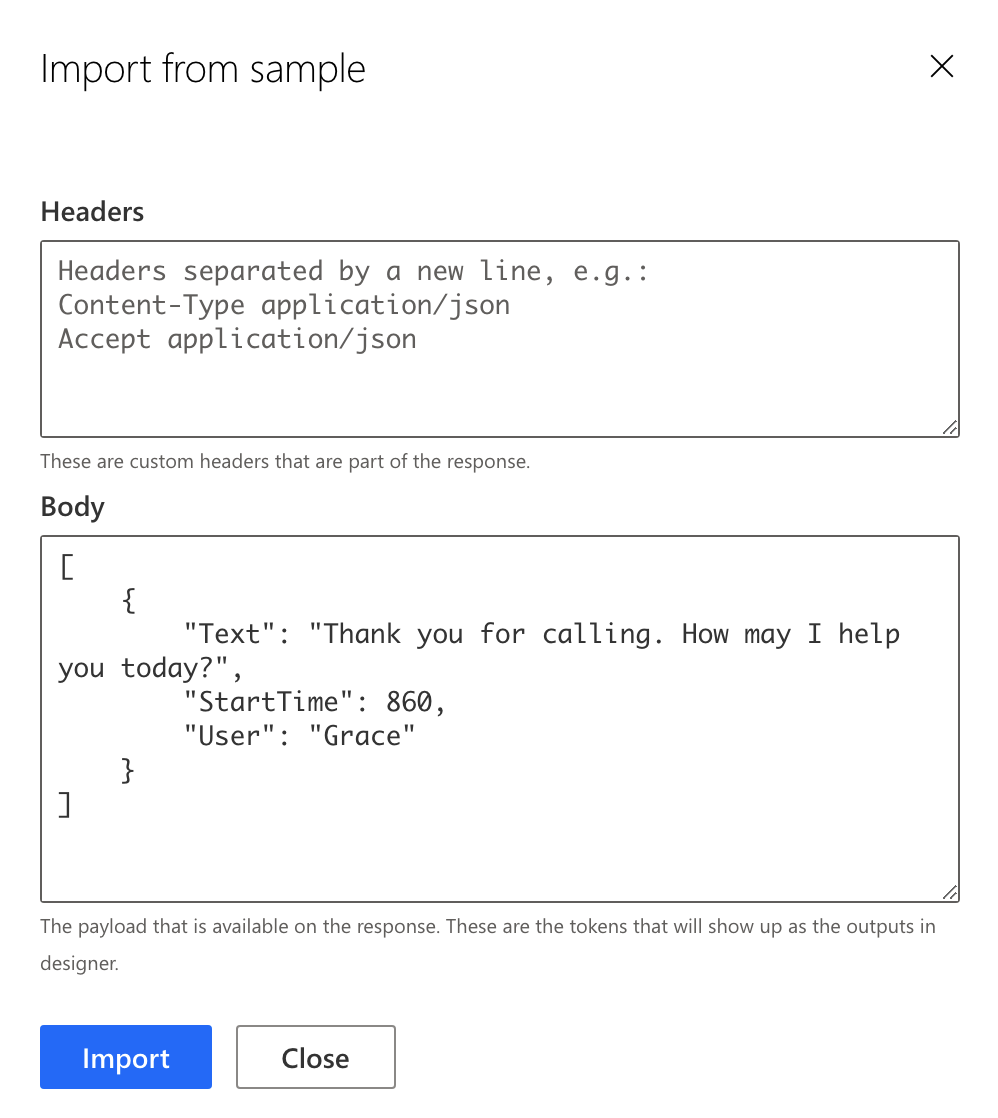
{

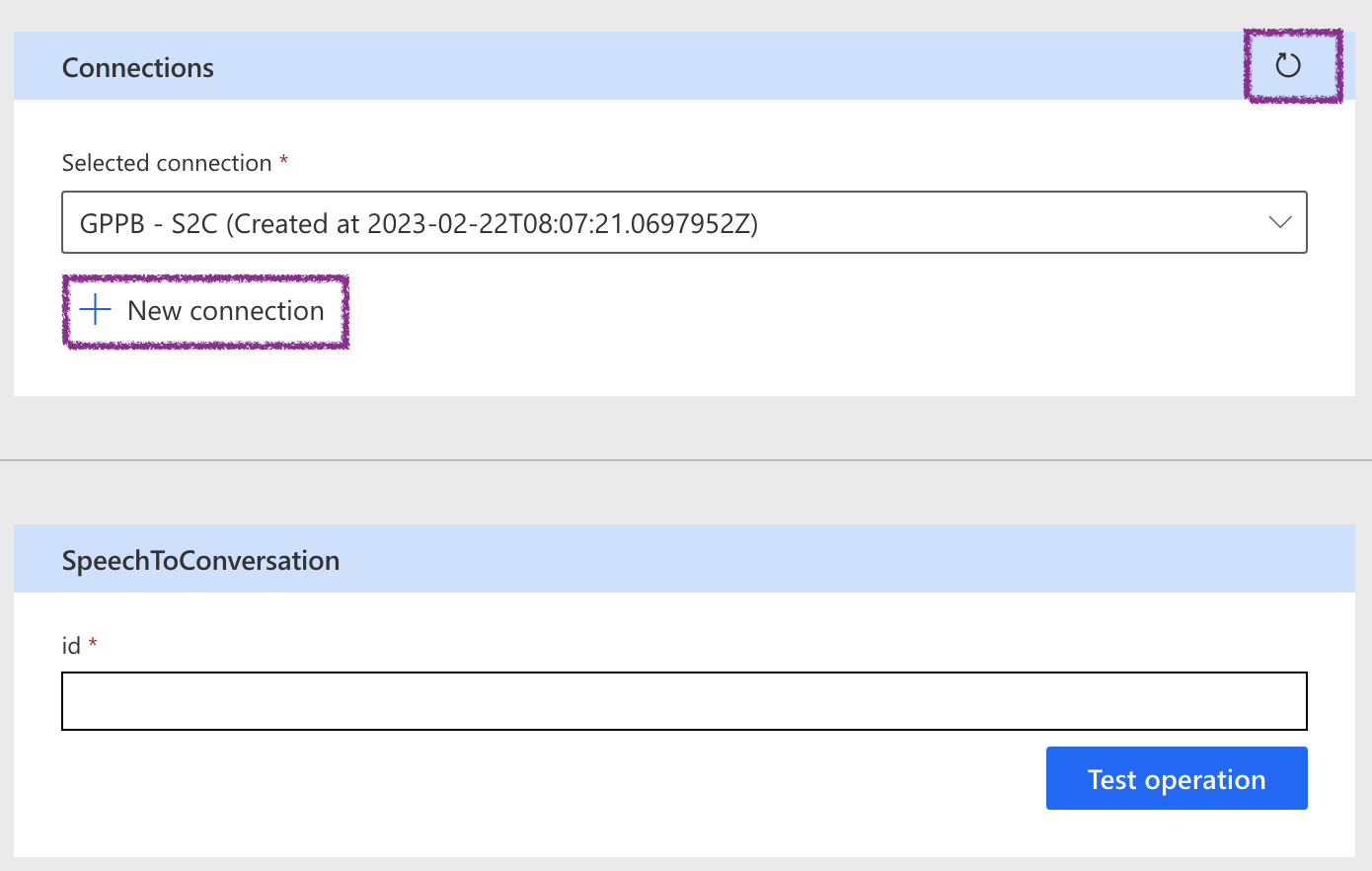
"Text": "Thank you for calling. How may I help you today?",

"StartTime": 860,

"User": "Grace"

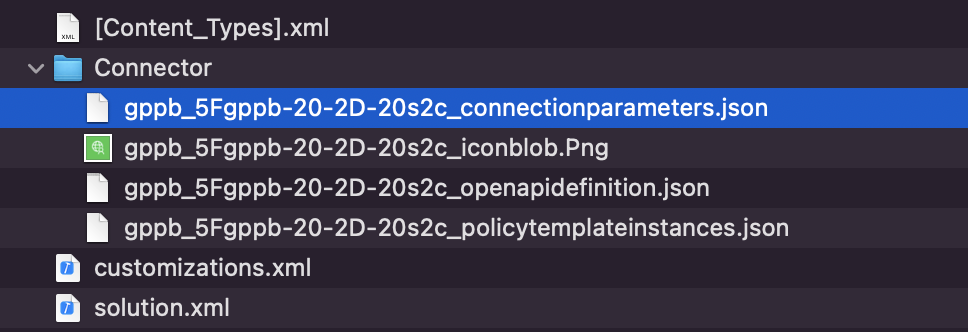
}

]  


1. Now, you can create a custom connector before testing.
2. Create a new connection to the external API. It will ask you the API key to connect to API. Enter the following API key: uZRlZNQafWOXIJYCprrdCvlWOaUWQWVvsv58M2Dl3rVGW0cTui3caA==  
   After that, you need to refresh the connection to see a newly created connection.  
   
3. Enter a valid conversation id from Dataverse to test operation.
4. Bravo! You have now successfully created a custom connector.

## A custom connector: Extended steps

The section focuses on creating a reusable custom connector which can connect to different endpoint.

1. Export the custom connector solution as an un-managed solution.
2. Un-zip the solution and open the connectionparameters.json  
   
3. Add the following text before api\_key.  
   This will be used in a custom policy to over-write the API url.

"api\_url": {

"type": "string",

"uiDefinition": {

"displayName": "API Url",

"description": "The API Url for this api",

"tooltip": "Provide your API Url",

"constraints": {

"tabIndex": 1,

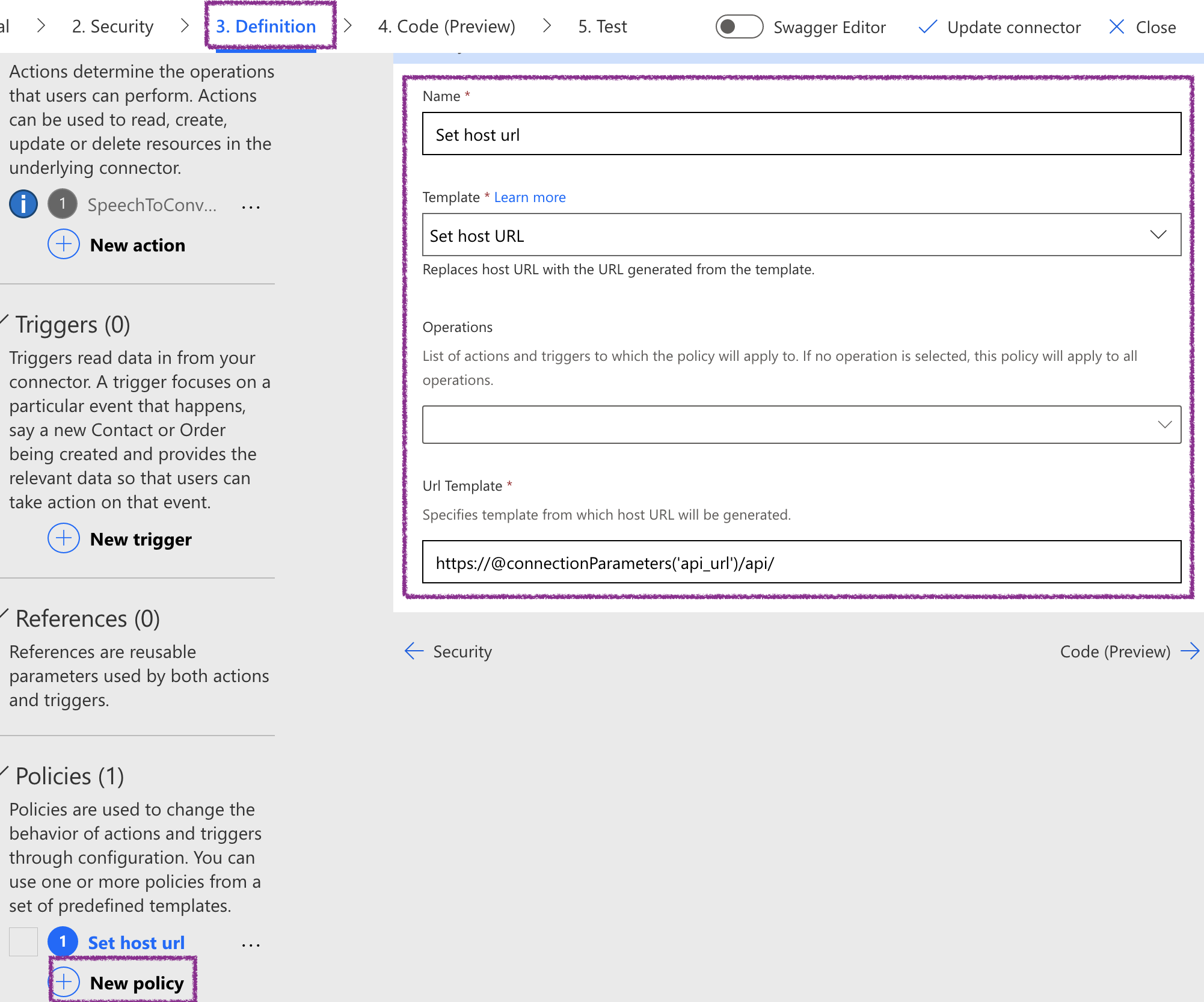
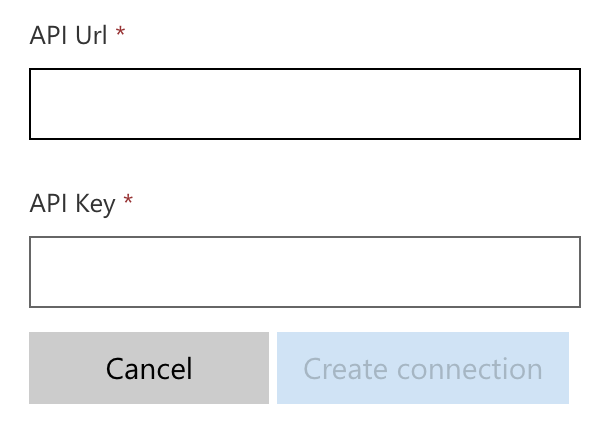
"clearText": true,

"required": "true"

}

}

},

1. Zip all files and import it to Dataverse.
2. Edit the custom connector. Add a new policy under Definition.  
   
3. Save and test the custom connector again. You will need to create a new connection.
4. You will see the connection is now asking the API URL.  
   Enter gppbfunction.azurewebsites.net in API Url, and uZRlZNQafWOXIJYCprrdCvlWOaUWQWVvsv58M2Dl3rVGW0cTui3caA== in API key.  
   
5. This method is useful if you have a same API in different environment. You don’t need to create multiple custom connectors. Instead, values can be parameterised.

# Session 4 - Power Automate

**Linn Zaw Win**

On create of conversation, pass the recording to Custom Connector and create sentences as per the returned JSON (no sentiment)

On create of sentences trigger the AI model and get the sentiment.

# Session 5 – Building a Power App (canvas)

**Abhay Mishra**

## Goals

1. Create a new solution (You can use your existing solution if already have created one)
2. Create a data structure (Your can use the existing schema and following tables: 1. Conversation (gppb\_conversation) 2. Sentence (gppb\_sentence)
3. Create a model-driven app to display the results

Create a gallery that includes a person image what they said and sentiment (try to alternate sides)

Build it as a custom page and embed in model-driven.

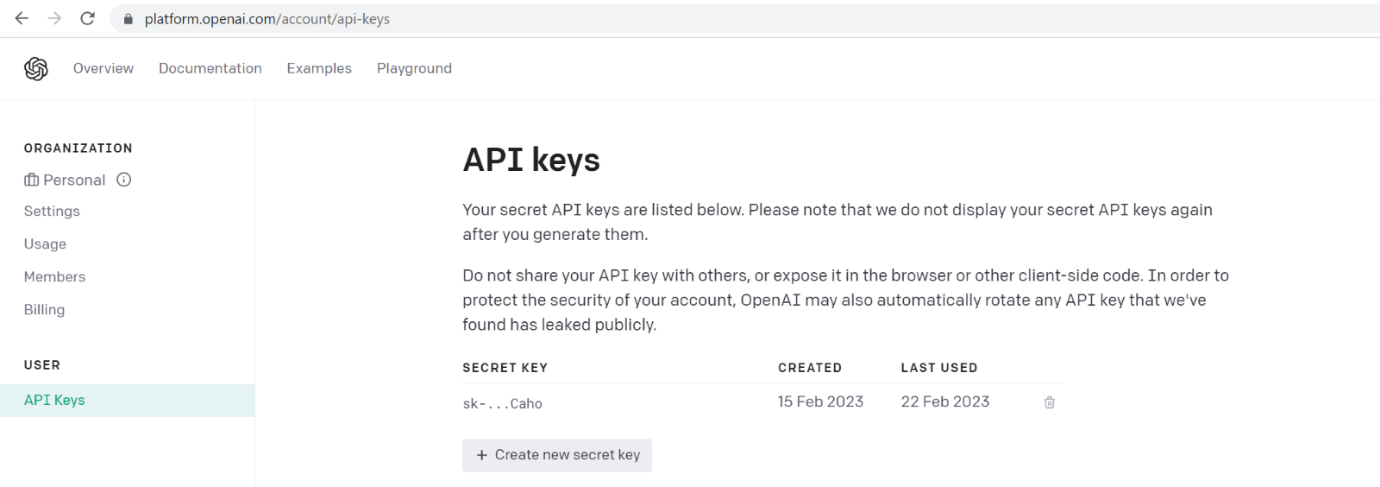
# Session 6 – Connect with ChatGPT

**Shobhit Bhatnagar**

Connect to Power Automate (desktop) to chatGPT and get the sentiment for the overall summary.

## **Pre-requisites:**

1. Create an account on ChatGPT: <https://chat.openai.com/>
2. Use your personal or work email to create an account.
3. Generate an API Key for connecting to ChatGPT via API
4. Navigate to <https://platform.openai.com/account/api-keys>
5. Create an API Key and copy and store it on your local machine in One Note.



For our session we will invoke API: <https://api.openai.com/v1/completions>

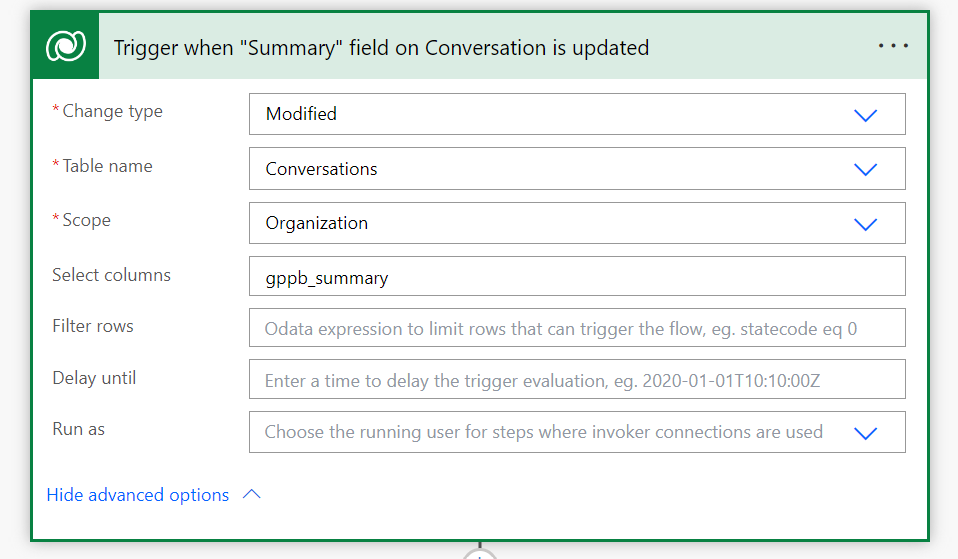
Documentation here: <https://platform.openai.com/docs/api-reference/completions>

## Steps

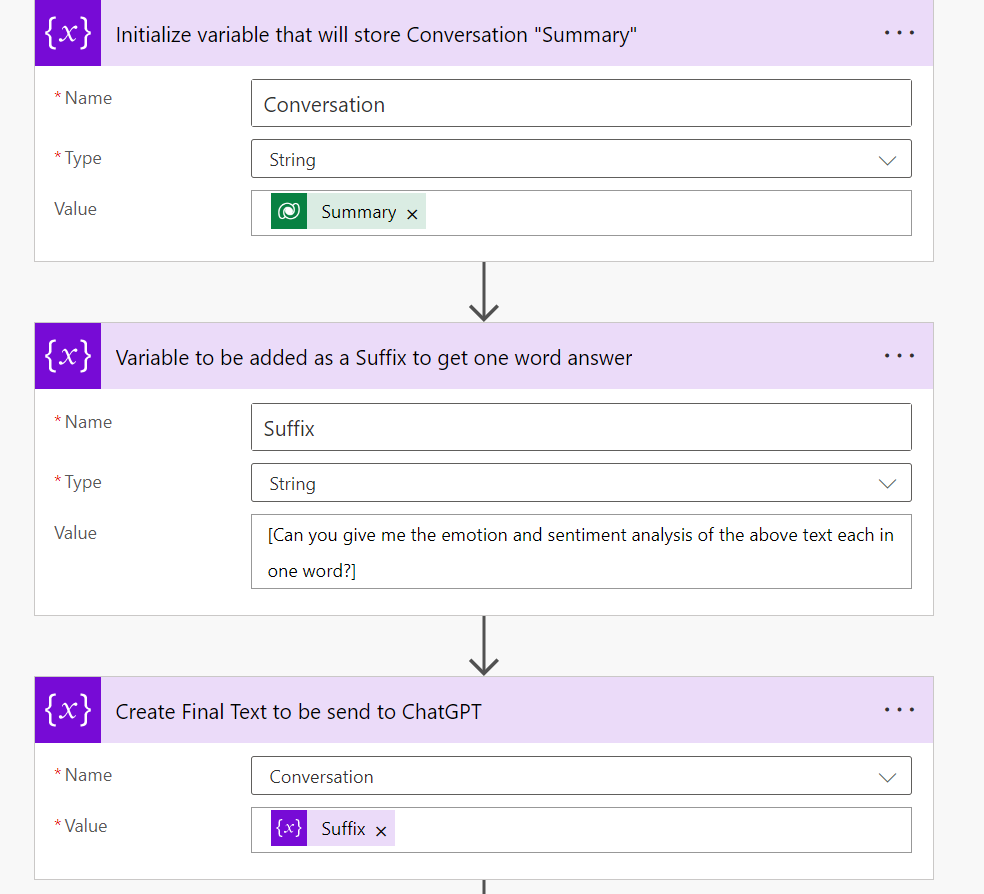
On update of Summary field of Conversation (once transcription of recording is done), trigger a Power Automate to invoke ChatGPT API using a HTTP connector and return the Emotion and Sentiment from the API. Use Parse Json to read and update the Emotion and Sentiment back on the Conversation record in Dataverse.

Steps

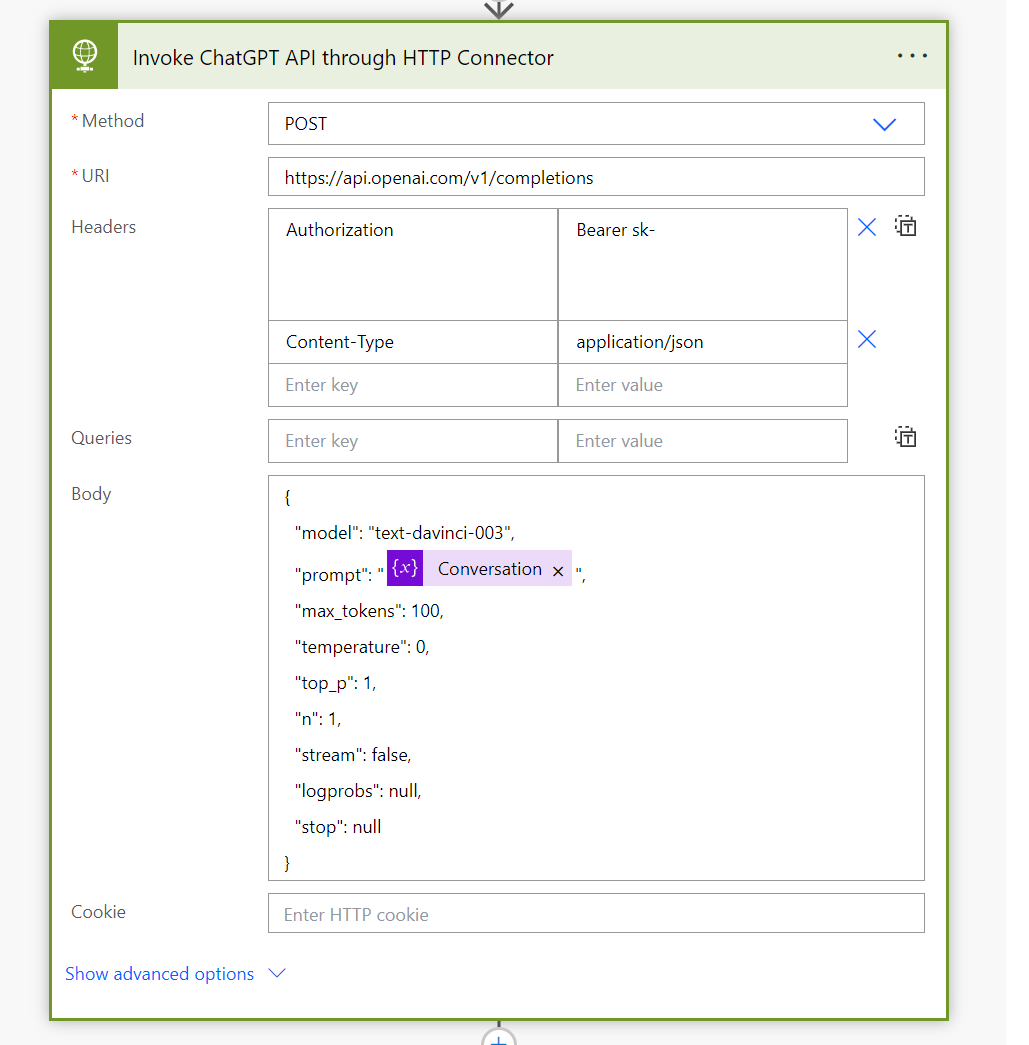
1. Navigate to <https://make.powerapps.com/> and open Solution.
2. Click
3. Start by adding a new Trigger as follows:
   1. Trigger type: Dataverse
   2. Trigger when “Record is modified”
   3. Table: Conversations
   4. Select column: Summary



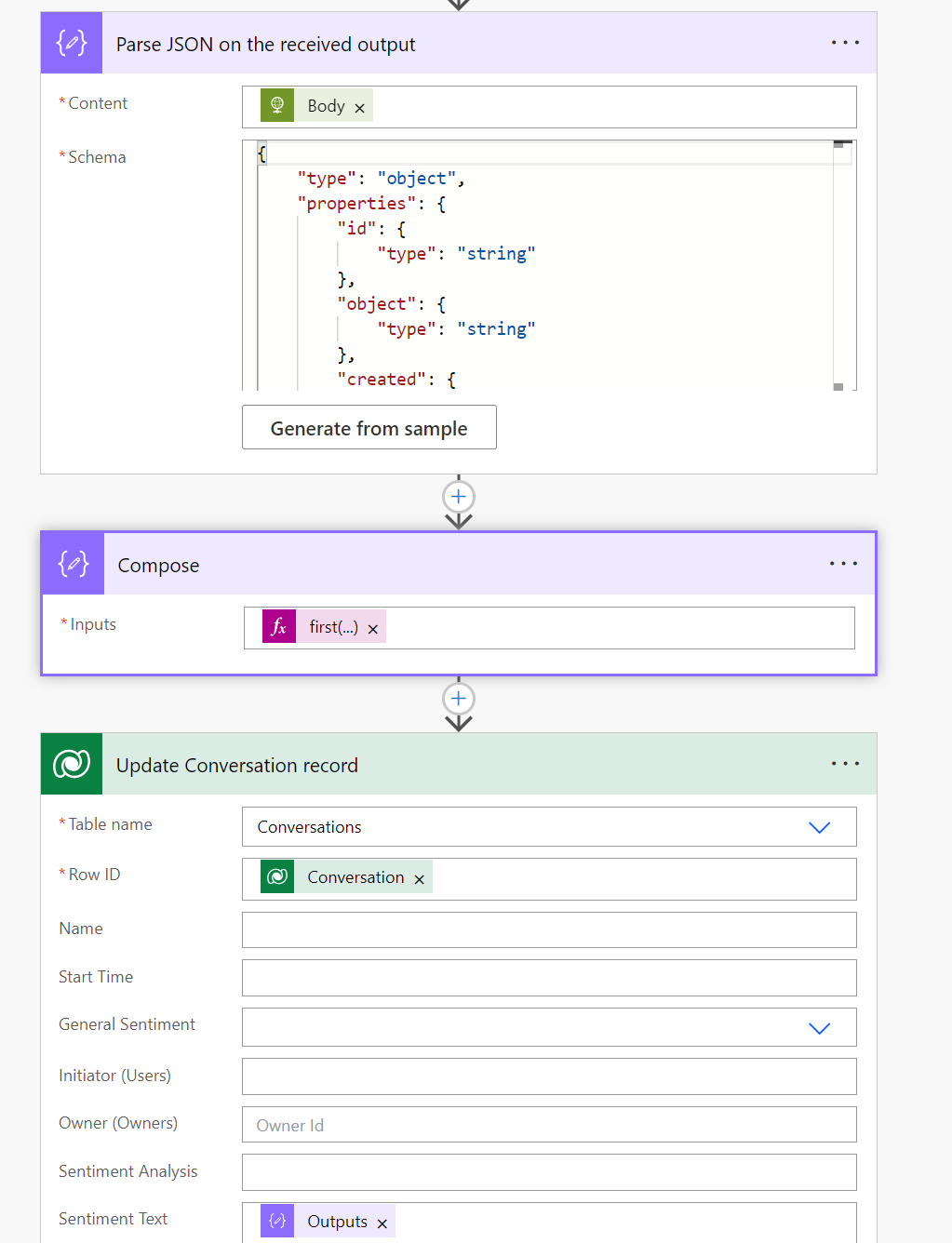
1. Set up a local Flow variable to store the “Summary” text and append specific “ChatGPT” formatted text at the end.



1. Use HTTP Connector to invoke ChatGPT API as follows:



1. Read the result and update Dataverse record
   1. We can use the following expression to get the desired text from the result set: first(body('Parse\_JSON')?['choices'])['text']



# Appendix

## Azure function app

**URL** <https://gppbfunction.azurewebsites.net/api/Speech2Conversation>

### Security

Function key is uZRlZNQafWOXIJYCprrdCvlWOaUWQWVvsv58M2Dl3rVGW0cTui3caA==

When connecting add the following header

|  |  |
| --- | --- |
| Key | Value |
| x-functions-key | uZRlZNQafWOXIJYCprrdCvlWOaUWQWVvsv58M2Dl3rVGW0cTui3caA== |

### Parameters

Pass query parameter id=123

e.g. <https://gppbfunction.azurewebsites.net/api/Speech2Conversation?id=123>

### Request Type

Accepts both get and post

### Response

JSON array containing a sentence’s text, start time, and user. Right now it is static, eventually it will pick the file associated with the record id passed and convert speech to text (wip).

[

    {

        "Text": "Hello",

        "StartTimeInMilliseconds": 0,

        "User": "1"

    },

    {

        "Text": "Welcome",

        "StartTimeInMilliseconds": 1500,

        "User": "3"

    }

]

