

Prerequisites

- An [API key](#) for the Cognitive Services Text Analytics API
- One of the following subscriptions:
 - [Power Automate](#)
 - [Power Apps](#)

Create the Text Analytics Resource

Step 1: Sign in to Azure Portal

1. Go to the [Azure Portal](#).
2. Sign in with your Azure account credentials.

Step 2: Create a New Resource

1. Once signed in, click on the **"Create a resource"** button found in the left-hand menu or on the homepage.
2. In the search box, type **"Text Analytics"** and select **"Text Analytics"** from the list of services.

Step 3: Configure the Text Analytics Resource

1. **Subscription:** Select the Azure subscription you want to use.
2. **Resource Group:** Choose an existing resource group or create a new one by clicking **"Create new"** and providing a name.
3. **Region:** Select the "West US" region.
4. **Name:** Enter a unique name for your Text Analytics resource. This name will be used to identify the resource in your Azure account.
5. **Pricing Tier:** Choose a pricing tier that suits your needs. The default is typically S0 (Standard).
6. **Resource Mode:** For a general-purpose Text Analytics resource, leave this as **"Single Service"**. If you need other Cognitive Services, you might consider the **"Multi-service resource"** option.

Step 4: Review and Create

1. After filling out all required fields, click on **"Review + create"**.
2. The portal will validate your settings. If everything is correct, click **"Create"**.
3. Azure will start deploying your resource. This may take a few moments.

Step 5: Access the Resource

1. Once the deployment is complete, click on **"Go to resource"**.
2. In the resource's overview page, you will find the **Endpoint** and **Key(s)** that you will need to connect your applications to the Text Analytics service.

Start the custom connector wizard

1. Sign in to [Power Apps](#) or [Power Automate](#).
2. On the left pane of power automate, select **More > Discover..**
3. Select **Custom connector** from data.
4. Select **New custom connector > Create from blank.**
5. Enter a name for the custom connector, and then select **Continue.**

Parameter	Value
Connector Name	SentimentDemo

Step 1: Update general details

From this point, we'll show the Power Automate UI, but the steps are largely the same across the technologies. We'll point out any differences.

On the **General** tab, do the following:

1. In the **Description** field, enter a meaningful value. This description will appear in the custom connector's details, and it can help others decide whether the connector might be useful to them.
2. Update the **Host** field to the address for the Text Analytics API. The connector uses the API host and the base URL to determine how to call the API.
- 2.

Parameter	Value
Description	Uses the Cognitive Services Text Analytics Sentiment API to determine whether text is positive or negative
Host	westus.api.cognitive.microsoft.com

Step 2: Specify authentication type

There are several options available for authentication in custom connectors. The Cognitive Services APIs use API key authentication, so that's what you specify for this tutorial.

1. On the **Security** tab, under **Authentication type**, select **API Key**.
2. Under **API Key**, specify a parameter label, name, and location. Specify a meaningful label, because this is displayed when someone first makes a connection with the custom connector. The parameter name and location must match what the API expects. Select **Connect**.
- 2.

Parameter	Value
Parameter label	API key
Parameter name	Ocp-Apim-Subscription-Key
Parameter location	Header

- At the top of the wizard, make sure the name is set to **SentimentDemo**, and then select **Create connector**.

Step 3: Create the connector definition

The custom connector wizard gives you many options for defining how your connector functions, and how it's exposed in logic apps, flows, and apps. We'll explain the UI and cover a few options in this section, but we also encourage you to explore on your own.

Create an action

The first thing to do is create an action that calls the Text Analytics API sentiment operation.

- On the **Definition** tab, the left pane displays any actions, triggers (for Logic Apps and Power Automate), and references that are defined for the connector. Select **New action**.

There are no triggers in this connector. To learn about triggers for custom connectors.

- The **General** area displays information about the action or trigger that's currently selected. Add a summary, description, and operation ID for this action.

Parameter	Value
Summary	Returns a numeric score representing the sentiment detected
Description	The API returns a numeric score between 0 and 1. Scores close to 1 indicate positive sentiment, while scores close to 0 indicate negative sentiment.
Operation ID	DetectSentiment

Leave the **Visibility** property set to **none**. This property for operations and parameters in a logic app or flow has the following options:

- none**: displayed normally in the logic app or flow
 - advanced**: hidden under another menu
 - internal**: hidden from the user
 - important**: always shown to the user first
- The **Request** area displays information based on the HTTP request for the action. Select **Import from sample**.
 - Specify the information necessary to connect to the API, specify the request body (provided after the following image), and then select **Import**. We provide this information

for you, but for a public API, you typically get this information from documentation such as [Text Analytics API \(v2.0\)](#).

Parameter	Value
Verb	POST
URL	<https://westus.api.cognitive.microsoft.com/text/analytics/v2.0/sentiment>
Body	Use the following JSON code

Example:

JSON

```
• {
  "documents": [
    {
      "language": "string",
      "id": "string",
      "text": "string"
    }
  ]
}
```

- The **Response** area displays information based on the HTTP response for the action. Select **Add default response**.
- Specify the response body, and then select **Import**. As we did for the request body, we provide this information for you following the image, but it's typically provided in the API documentation.

Example:

JSON

```
{
  "documents": [
    {
      "score": 0.0,
      "id": "string"
    }
  ],
  "errors": [
    {
      "id": "string",
      "message": "string"
    }
  ]
}
```

The **Validation** area displays any issues that are detected in the API definition. Check the status, and then in the upper-right corner of wizard, select **Update connector**.

Update the definition

Now let's change a few things so that the connector is more friendly when someone uses it in a logic app, flow, or app.

1. In the **Request** area, select **body**, and then select **Edit**.
2. In the **Parameter** area, you now see the three parameters that the API expects: `id`, `language`, and `text`. Select **id**, and then select **Edit**.
3. In the **Schema Property** area, update values for the parameter, and then select **Back**.

• Parameter	Value
Title	ID
Description	An identifier for each document that you submit
Default value	1
Is required	Yes

- In the **Parameter** area, select **language** > **Edit**, and then repeat the process that you used for `id` in steps 2 and 3 of this procedure, with the following values.

• Parameter	Value
Title	Language
Description	The two or four character language code for the text
Default value	en
Is required	Yes

- In the **Parameter** area, select **text** > **Edit**, and then repeat the process that you used for `id` in steps 2 and 3 of this procedure, with the following values.

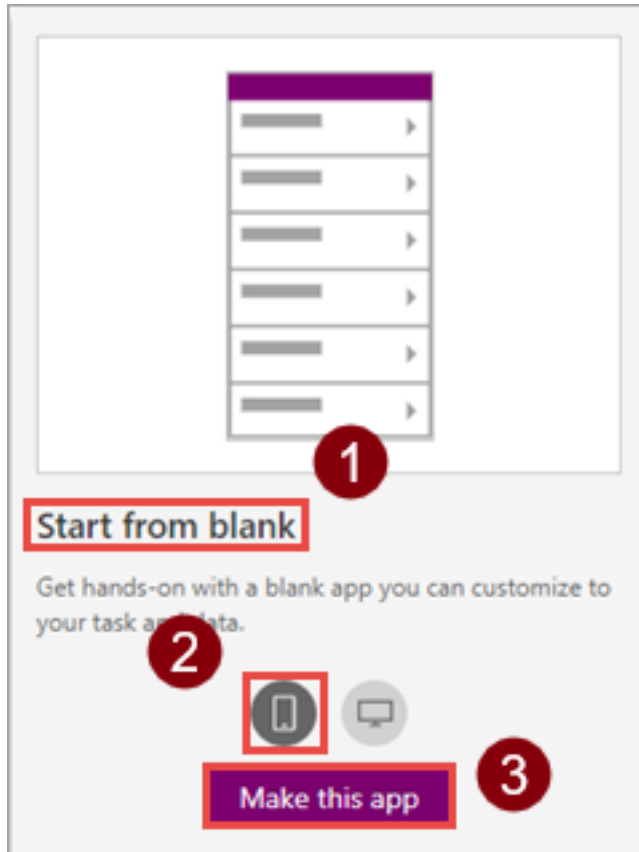
• Parameter	Value
Title	Text
Description	The text to analyze for sentiment
Default value	None
Is required	Yes

- In the **Parameter** area, select **Back** to take you back to the main **Definition** tab.
- In the upper-right corner of the wizard, select **Update connector**.

Create the app and add the custom connector

The first thing you do is create an app from blank, then connect to the custom connector that you created in a previous topic.

1. In make.powerapps.com, In the left side menu select Create. Then select **Blank App** > **Create button below Blank Canvas App** >
 1. Give it a good name.
 2. Select phone and click on Create.



2. On the app canvas left side menu, choose **data**.
3. On the **Data** panel, choose the connection you created in a previous topic (such as "SentimentDemo").
4. Save the app with the name `Sentiment Analysis`.

Add controls to the app

You now build out the UI for the app, so that you can enter text, submit that text to the API, and get a response.

1. Add a rectangle icon as a title bar, then add the label "Sentiment Analysis".

Sentiment Analysis

2. Add the label "Enter your text, then click Get score", then add a text input control.

Sentiment Analysis

Enter your text, then click Get score

3. Add a button with the text "Get score".

Sentiment Analysis

Enter your text, then click Get score

Text input

Get score

4. Add the label "The sentiment score is". In the next section, you add a formula to complete this label.

Sentiment Analysis

Enter your text, then click Get score

Text input

Get score

The sentiment score is

Add formulas to drive behavior

With the data connection and UI in place, now you add Power Apps formulas that drive the behavior of the app. The formulas call the API through the custom connector, store the result in a *collection* (a tabular variable), then display the formatted result in the app.

1. Choose the button you created, then set the **OnSelect** property of the button to the name of the connector (including the period).

- `SentimentDemo.`

Power Apps gives you an auto-complete option of `DetectSentiment` because the custom connector makes this available.

- Now set the **OnSelect** property of the button to the following formula.

```
ClearCollect(sentimentCollection, SentimentDemo.DetectSentiment(
    {id:"1", language:"en", text:TextInput1.Text}).documents.score)
```

This formula gets the sentiment score from the API, and stores it in a collection:


1. The formula calls the `DetectSentiment` function with the three parameters exposed by the custom connector: `id`, `language`, and `text`. We specify values for the first two right in the formula, and get the value for `Text` from the text input control (you could also pull the first two values from somewhere else in an app).
2. The function returns a `score` for each document that you send; in our examples, we send one document at a time. The score ranges from 0 (negative) to 1 (positive).
3. The formula then calls the `ClearCollect` function to remove any existing values from the `sentimentCollection` and add the value from `score`.

- Choose the label that you created, then set the **Text** property of the label to the following formula.

3. `"The sentiment score is " & Round(First(sentimentCollection).score, 3) * 100 & "%"`
4. This formula gets the sentiment score from the collection, and formats and displays it:
 1. The `First()` function returns the first (and in this case only) record in `sentimentCollect`, and displays the `score` field (the only field) associated with that record.
 2. The `Round()` function rounds the score to 3 places; the rest of the formula formats the result as a percentage and adds some information for context.

Test the app

Now run the completed app to make sure it works as expected.

1. Choose  in the top right to run the app.
2. Enter a phrase in the text input control, and choose **Get score**. The sentiment score should be displayed within a few seconds.

The finished app looks like the following image:

Sentiment Analysis

Enter your text, then click Get score

I enjoyed the new movie after a long day

Get score

The sentiment score is 97.1% positive.

It's a simple app, but it gains powerful functionality by being able to call Cognitive Services through a custom connector.