

Lab 10. Create a Custom Connector

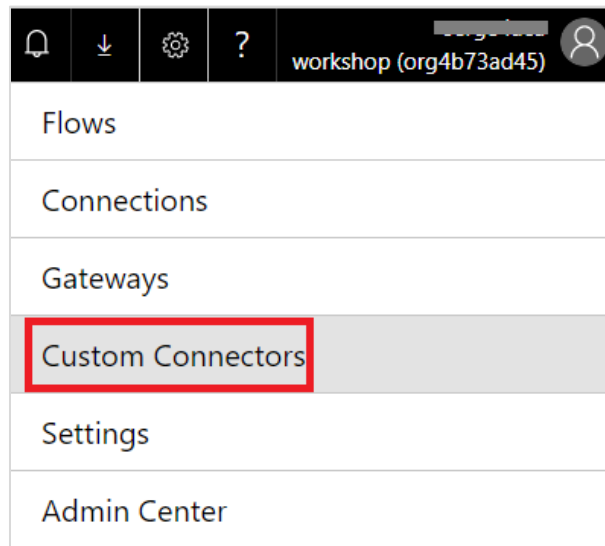
Lab 11. Create a Custom Connector

Author: Serge Luca aka "Doctor Flow"

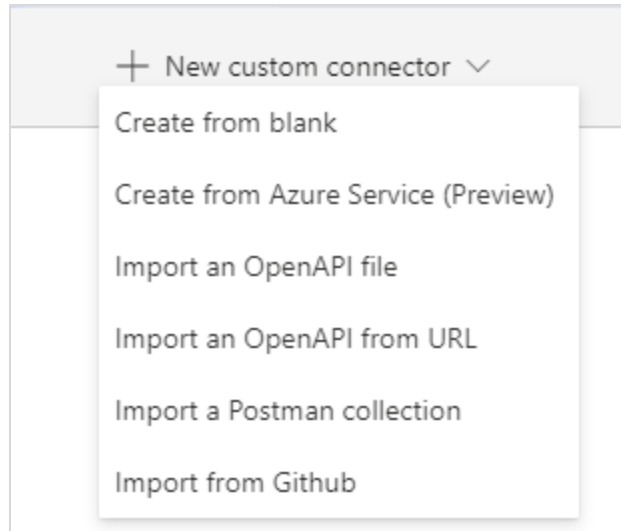
Prerequisites: lab **call an external API with ower Automate with the HTTP action.**

Tasks:

1. Before starting this lab, make sure your API key has been generated. Then, you can reuse the key from the previous lab.
2. Go to the **Connector** menu and select **Custom Connectors**:



3. Click on **New custom connector**:



4. Select **Create from blank**.
5. Name the connector "PPA Training - AccuWeather" and click **Continue**

Create from blank

Connector name


PPA Training - AccuWeather

Continue

Cancel

6. Upload a picture to be associated with your connector, provide the **host** (**dataservice.accuweather.com**) and a short description of what your connector does

General information



Upload connector icon
Supported file formats are PNG and JPG. (< 1MB)

Icon background color

A color to show behind the icon (e.g., '#007ee5')

Description

Provide the weather in a city

☐ Connect via on-premises data gateway [Learn more](#)

Scheme *

☒ HTTPS ☐ HTTP

Host *

dataservice.accuweather.com

Base URL

/

7. Click **Security** to move to the next screen.
8. The authentication type should be **API key**.
9. Since we want the Key parameter to be provided in the query string, create an API key with **API key** as Parameter label and **apikey** as a Parameter name; switch the parameter location to Query as illustrated in the following picture:

Authentication type

Choose what authentication is implemented by your API *

API Key

 Edit

API Key

Users will be required to provide the API Key when creating a connection

Parameter label *

API Key

Parameter name *

apikey

Parameter location *

Query

 Edit

10. Click **Definition**:

1. General > 2. Security > **3. Definition** > 4. Test

Actions (0)

Actions determine the operations that users can perform. Actions can be used to read, create, update or delete resources in the underlying connector.

+ New action

Triggers (0)

Triggers read data in from your connector. A trigger focuses on a particular event that happens, say a new Contact or Order being created and provides the relevant data so that users can take action on that event.

+ New trigger

References (0)

References are reusable parameters used by both actions and triggers.

11. Click on **New action**:

12. Define the Action as follow:

1. General > 2. Security > **3. Definition** > 4. Code (Preview) > 5. Test ☐ Swagger Editor ☒ Update connector

Actions (1)

Actions determine the operations that users can perform. Actions can be used to read, create, update or delete resources in the underlying connector.

☐ **GetWeather** ...

+ New action

Triggers (0)

Triggers read data in from your connector. A trigger focuses on a particular event that happens, say a new Contact or Order being created and provides the relevant data so that users can take action on that event.

General

Summary [Learn more](#)

Get current weather

Description [Learn more](#)

This action returns the current weather on a specific location

Operation ID *

This is the unique string used to identify the operation.

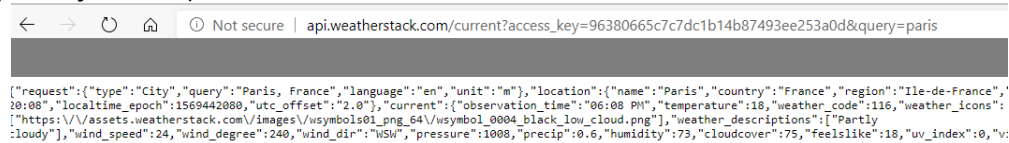
GetWeather

Visibility [Learn more](#)

☒ none ☐ advanced ☐ internal ☐ important

13. In a new browser tab, type a weather request to make sure it works fine and also to generate sample data that we will reuse in our connector; replace the location key with {location} because this is a value that will be passed dynamically

<https://dataservice.accuweather.com/currentconditions/v1/{location}?apikey=wAcxfiZGgPCAYvPC5pEITDojGhUHQqbV&details=true>



14. Keep this tab open and go back to the connector definition.
15. Down below in **Request**, click **Import from sample**

A screenshot of a web interface for defining a connector. The 'General' tab is active. It contains sections for 'Summary' (with a 'Learn more' link and a text input 'Get weather'), 'Description' (with a 'Learn more' link and a text input 'Get weather in a specific city'), 'Operation ID' (with a text input 'GetWeather' and a description: 'This is the unique string used to identify the operation.'), and 'Visibility' (with radio buttons for 'none', 'advanced', 'internal', and 'important', where 'none' is selected). Below these is the 'Request' section, which includes a description: 'It defines the pre-requirements needed in order to make a request. Describes a single operation parameter. A unique parameter is defined by a combination of a name and location.' At the bottom of the 'Request' section, there is a button labeled '+ Import from sample' which is highlighted with a red rectangle.

16. Pass your query string, set the verb to **Get** and click **import**

Import from sample



Verb *

☒ GET ☐ DELETE ☐ POST ☐ PUT ☐ HEAD ☐ OPTIONS
☐ PATCH

URL *

`https://dataservice.accuweather.com/currentconditions/v1/{location}?ap`

This is the request URL.

Headers

Headers separated by a new line, e.g.:
Content-Type application/json
Accept application/json

These are custom headers that are part of the request.

Import

Close

17. After importing, this is what you will get:

Request

It defines the pre-requirements needed in order to make a request. Describes a single operation parameter. A unique parameter is defined by a combination of a name and location.

Request

+ Import from sample

Verb *

The verb describes the operations available on a single path.

GET

URL *

This is the request URL.

`https://dataservice.accuweather.com/currentconditions/v1/{location}`

Path

Path is used together with Path Templating, where the parameter value is actually part of the operation's URL.

* location

Query

Query parameters are appended to the URL. For example, in `/items?id=###`, the query parameter is `id`.

apikey

details

Headers

These are custom headers that are part of the request.

Body

The body is the payload that's appended to the HTTP request. There can only be one body parameter.

18. **apikey** will be registered in the connector in such a way that there is no need to pass the key for each query; therefore, we can delete it:

Request

It defines the pre-requirements needed in order to make a request. Describes a single operation parameter. A unique parameter is defined by a combination of a name and location.

Request

+ Import from sample

Verb *

The verb describes the operations available on a single path.

GET

URL *

This is the request URL.

https://dataservice.accuweather.com/currentconditions/v1/{location}

Path

Path is used together with Path Templating, where the parameter value is actually part of the operation's URL.

* location

▼

Query

Query parameters are appended to the URL. For example, in /items?id=####, the query parameter is id.

details

▼

Headers

These are custom headers that are part of the request.

Body

The body is the payload that's appended to the HTTP request. There can only be one body parameter.

19. Edit the **details** parameter and fill in the **Description** , the **Summary**, the **default value** and the visibility as follows:

Parameter

Name *

details

Description [Learn more](#)

Provide more details on the weather

Summary [Learn more](#)

more details

Default value

false

Is required?

☐ Yes ☒ No

Visibility [Learn more](#)

☐ none ☒ advanced ☐ internal ☐ important

Location *

☐ Path ☒ Query ☐ Header ☐ Body

Type

boolean

Format

Dropdown type [Learn more](#)

☒ Disabled ☐ Static ☐ Dynamic

20. Click on **Back**, scroll to **Response**, click on **default** to import another sample; the scroll bar is in the middle of the screen as illustrated in the picture:

Request

+ Import from sample

Verb *

The verb describes the operations available on a single path.

GET

URL *

This is the request URL.

http://api.weatherstack.com/current

Path

Path is used together with Path Templating, where the parameter value is actually part of the operation's URL.

Query

Query parameters are appended to the URL. For example, in /items?id=####, the query parameter is id.

* City ...

Headers

These are custom headers that are part of the request.

Body

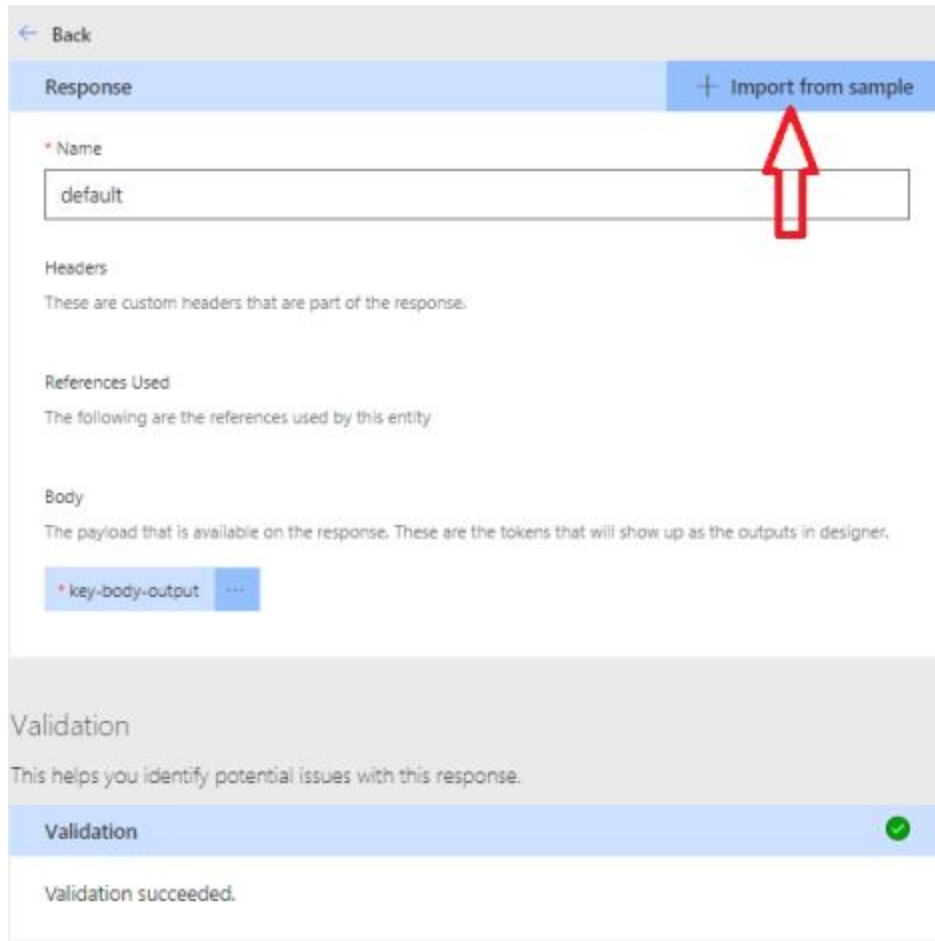
The body is the payload that's appended to the HTTP request. There can only be one body parameter.

Response

It defines the shape of response returned by the underlying connector when making the request.

default default

21. In the next window, click on **Import from sample**:



← Back

Response + Import from sample

* Name
default

Headers
These are custom headers that are part of the response.

References Used
The following are the references used by this entity.

Body
The payload that is available on the response. These are the tokens that will show up as the outputs in designer.

* key-body-output ...

Validation
This helps you identify potential issues with this response.

Validation ✓

Validation succeeded.

22. ...and in the next window, paste the JSON result grabbed from the browser (with &details=false):

```
[{"LocalObservationDateTime": "2023-02-20T05:32:00+01:00", "EpochTime": 1676867520, "WeatherText": "Mostly cloudy", "WeatherIcon": 38, "HasPrecipitation": false, "PrecipitationType": null, "IsDayTime": false, "Temperature": {"Metric": {"Value": 6.2, "Unit": "C", "UnitType": 17}, "Imperial": {"Value": 43.0, "Unit": "F", "UnitType": 18}}, "MobileLink": "http://www.accuweather.com/en/be/brussels/27581/current-weather/27581?lang=en-us", "Link": "http://www.accuweather.com/en/be/brussels/27581/current-weather/27581?lang=en-us"}]
```

23. Click **Import**.

24. Several new properties have been created in the body area:

Response

+ Import from sample

Name *

default

Headers

These are custom headers that are part of the response.

References Used

The following are the references used by this entity

Body

The payload that is available on the response. These are the tokens that will show up as the outputs in designer.

EpochTime

HasPrecipitation

IsDayTime

Link

LocalObservationDat...

MobileLink

PrecipitationType

Unit

UnitType

Value

Unit

UnitType

Value

Weathericon

WeatherText

- Edit each Value property (there is two "Value") and make sure their type is **number** and the format is **float**:

← Back

Schema Property

Title

Description [Learn more](#)

Value

Visibility [Learn more](#)

☒ none
☐ advanced
☐ internal
☐ important

Type

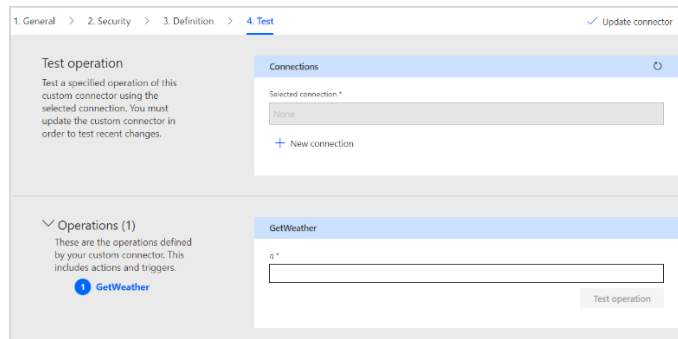
number

Format

float

- Click **Create Connector**

27. You can now test the connector (click **Test**):



28. However, you have to create a New connection; click on **New connection**, provide your key, and click again on **Create connection**:

The screenshot shows a dialog box titled 'Add Demo Weather connection'. It has a back arrow icon on the left. The main content area has a label 'key *' above a text input field. The input field contains a series of dots, indicating a masked password. At the bottom of the dialog, there are two buttons: a grey 'Cancel' button and a blue 'Create connection' button.

29. You will be redirected to the previous page, where you can click the refresh icon to display the new connection:

Connections

Selected connection *

PPA Training - AccuWeather (Created at 2023-02-19T14:59:37.8177145Z)

+ New connection

GetWeather

location *

details

Test operation

30. Provide the **city location** like 27581 for Brussels, type false in details, click **Test Operation**, and you should get the corresponding weather:

GetWeather

location *

27581

details

false

Test operation

Request

Response

Status

(200)


Headers

```
{
  "accept": "application/json,text/plain,*/*",
  "accept-encoding": "gzip,deflate",
  "accept-language": "en-US,en;q=0.9,fr;q=0.8",
  "akamai-nrm": "0 099eca17 16768666772 479hd8r3"
}
```

Body

```
[
  {
    "LocalObservationDateTime": "2023-02-20T05:17:00+01:00",
    "EpochTime": 1676866620,
    "WeatherText": "Mostly cloudy",
    "WeatherIcon": 38,
    "HasPrecipitation": false,
    "PrecipitationType": null,
    "IsDayTime": false,
    "Temperature": {
      "Metric": {
        "Value": 6.2,
        "Unit": "C",
        "UnitType": 17
      }
    }
  }
]
```

31. Click Create connector to deploy it
32. Click **close**, and you will notice that your connector has been generated:

Custom connectors		+ New custom connector ▾	
Icon	Name	Actions	
	PPA Training - AccuWeather serge Luca	<div>+</div> <div>↓</div> <div>✎</div> <div>⋮</div>	

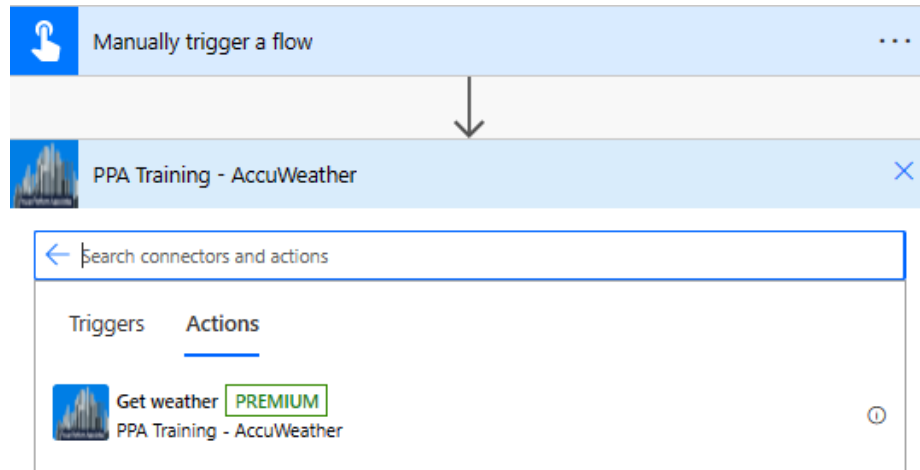
33. You can now create an instant Flow from blank to use this custom connector

The screenshot shows the 'Build an instant cloud flow' dialog box. On the left, there is an illustration of a laptop with a play button icon and a list of tasks. Below it, text states: 'Triggered manually from any device, easy-to-share instant flows automate tasks so you don't have to repeat yourself.' Examples listed are: 'Get an automatic mobile alert whenever a VIP client emails you' and 'Save all your email attachments to a folder automatically'. On the right, the 'Flow name' field contains 'weather flow'. Below it, the 'Choose how to trigger this flow' section lists several options: 'Manually trigger a flow' (selected with a checkmark), 'PowerApps', 'When Power Virtual Agents calls a fl...', 'When a flow step is executed', 'For a selected message', 'For a selected file', and 'Power BI button clicked'. At the bottom right, there are 'Skip', 'Create', and 'Cancel' buttons.

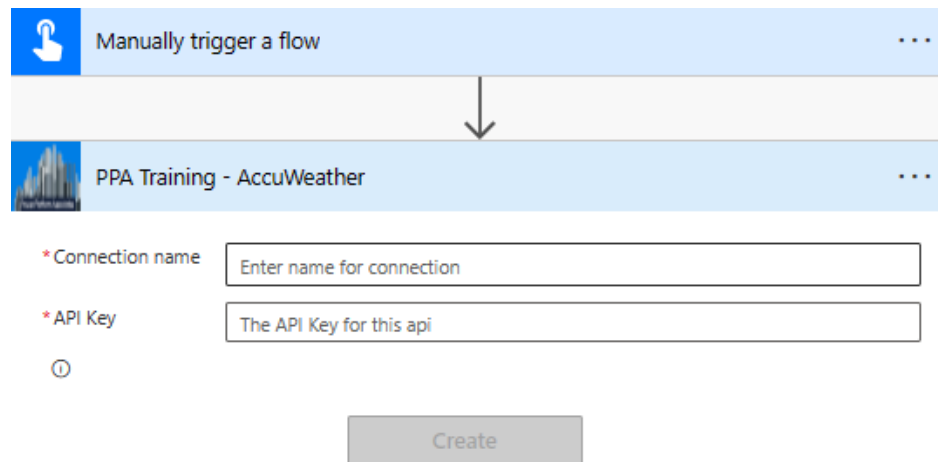
34. Add an "action" from the **Custom** category; you should find you Demo Weather custom connector:

The screenshot shows the 'Choose an operation' dialog box. At the top, there is a blue header bar with a hand icon and the text 'Manually trigger a flow'. Below it, a grey bar contains a search icon and the text 'Search connectors and actions'. A list of categories is shown: 'All', 'Built-in', 'Standard', 'Premium', 'Custom' (highlighted with a red box), and 'My clipboard'. Below the categories, there is a grid of connector icons. One icon is labeled 'PPA Training'.

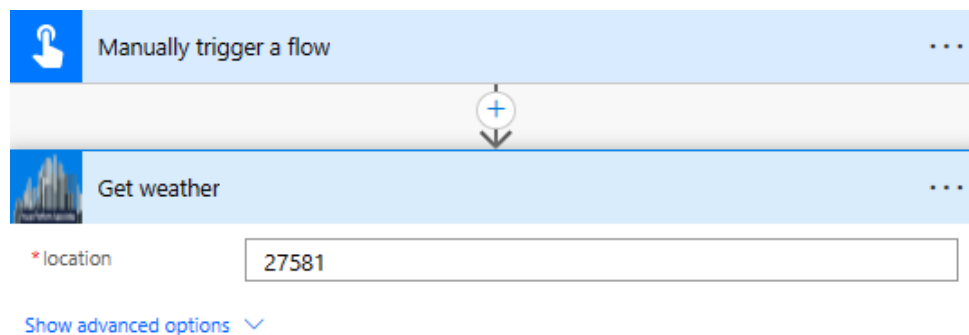
35. Select the **your connector**



36. You will be requested to provide the **API key** and a **name** for the new connection and click **create**:



37. Provide 27581 (or another location code) and save your Flow:



38. Run the Flow and check the output of "Get Weather":

The screenshot displays a Power Automate flow with two steps:

- Manually trigger a flow**: The first step, which triggers the flow.
- Get weather**: The second step, which retrieves weather data. It has a duration of 0s and a green checkmark indicating success.

The **Get weather** step details are as follows:

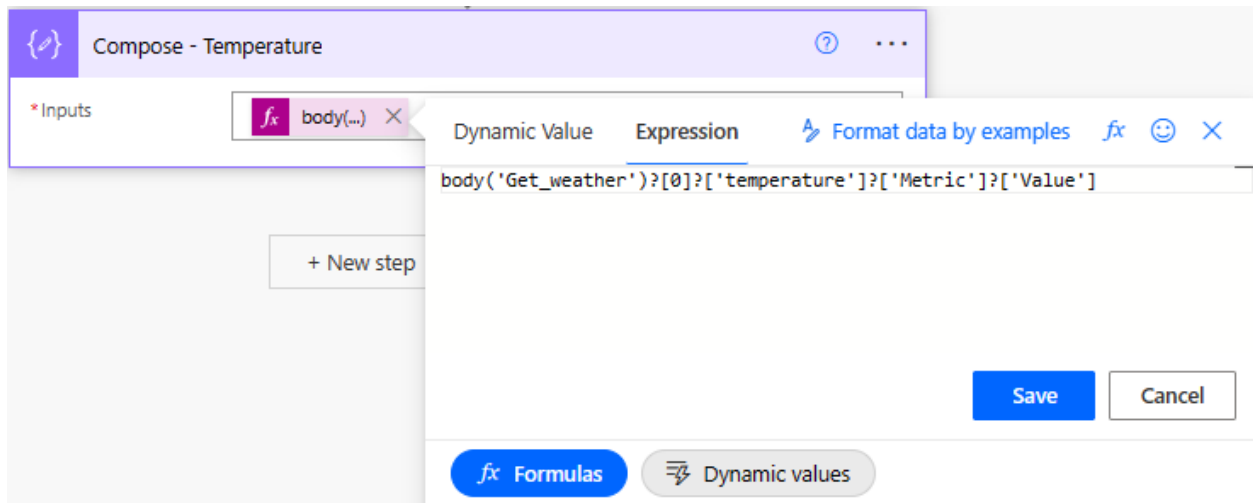
- INPUTS**:
 - location**: 27581
 - Show more**: A link to expand the input fields.
- OUTPUTS**:
 - body**: A JSON object containing weather data:

```
[
  {
    "LocalObservationDateTime": "2023-02-22T05:42:00+01:00",
    "EpochTime": 1677040920,
    "WeatherText": "Mostly cloudy",
    "WeatherIcon": 38,
    "HasPrecipitation": false,
    "Temperature": 5.1
  }
]
```
 - Show raw outputs**: A link to view the raw output data.

At the bottom, the connection is identified as **AccuWeather** with a green checkmark.

In the following steps, we will get the temperature in Celcius

39. Add a Compose and store the temperature value:



40. Test your flow.

We need your feedback

Do you want to report an issue or suggest something? We need your feedback:

<https://github.com/Power-Automate-in-a-day/Training-by-the-community/issues>