**Watson Assistant: learn to access data from third party applications and APIs**

In this article, you are provided with the information needed to set up a **Watson Assistant** instance to augment a conversation with live data from API calls. APIs can be set up to fetch near real time data from databases and/or to fetch public information on the internet.

Chatbots can be defined as “A computer program designed to simulate conversation with human users, especially over the Internet.”

Chatbots are gaining more and more popularity across all knowledge and business domains, especially in healthcare, banking, and insurance industries. Reasons driving the popularity of chatbots are high efficiency, swift response, cost savings, better customer experience, and actionable insights to name a few.

=

One such capable solution by IBM is **Watson Assistant**, an AI capable chatbot which can allow you to harness the power of AI with minimal technical knowledge across any messaging platform, application, device or channel. This self-help Watson Assistant service provides users with swift and accurate means of communication along-with an ability to improvise and learn from user conversation.

=

Out of the stack, we will provide the tools to accelerate use of Web functions to perform the API calls, return data to the chatbot, and then display the data to augment conversations.

Technically speaking Web Functions, also known as FaaS (Function-as-a-Service), are a type of cloud-computing service that allows you to execute code in response to events without the complex infrastructure typically associated with building and launching micro-services applications.

This approach has several benefits, some of them are:

* Pay only for the resources you use, when you use them
* Scale up or down automatically
* Focus more on code, not infrastructure

=

The real question is how can this add value?

This functionality can add tremendous value to the chatbot by allowing it to collect, analyse, aggregate, and search data through the third-party APIs. Imagine having machine learning models in place and being able to get insight directly from your chatbot in response to a question — wouldn’t that be fantastic?

=

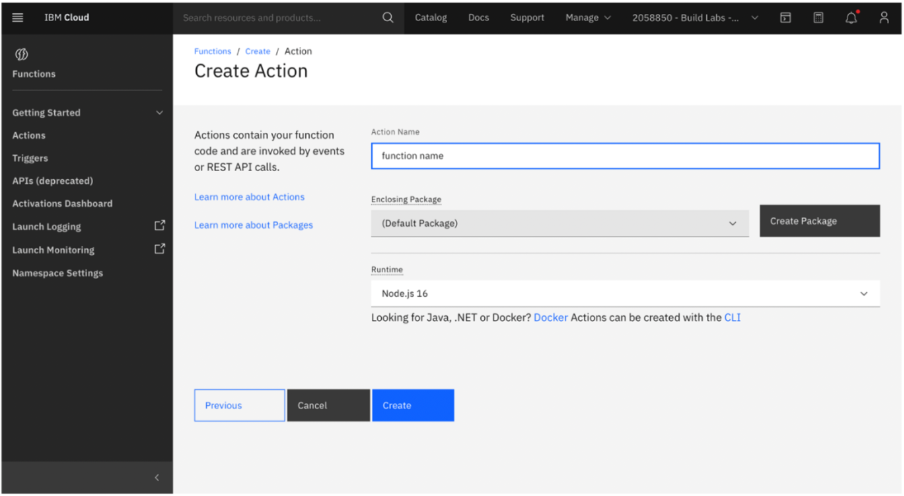
Web functions are low-code solutions that can be implemented by your API developer using predefined Python or Node.js templates.

Here is a step by step guide to implement web function:

Web functions are available on IBM cloud > Functions > Actions

Tip: Make sure your Function namespace and your Watson Assistant both are on same region.

Step 1: Create an Action by providing a name and selecting a runtime environment (which could be python or node.js) and click on Create:



Step 2: One can use the predefined API templates and modify them to create a custom program that fits a particular need. [Here](https://github.com/ibm-build-lab/Watson-Assistant/tree/main/external-api-web-functions) are templates for NodeJS and Python languages. Simply, cut and paste the code into your Web Function action.

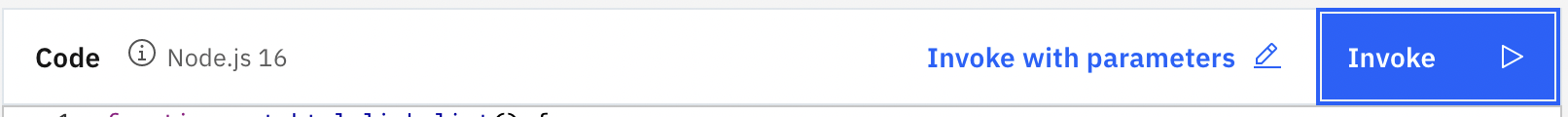




The above code displays multiple conditions and when the parsed value matches the parameter value the code gets executed.

* Here, the “url: and raw\_url:” are API endpoint URLs.
* API key can be setup, if needed otherwise could be removed.
* Request “method” could be “get”, “put” or “post.
* One can manipulate the return values in the *return* block.

Once done, *Save* the function. You will see *Invoke* and *Invoke with Parameters* as below:



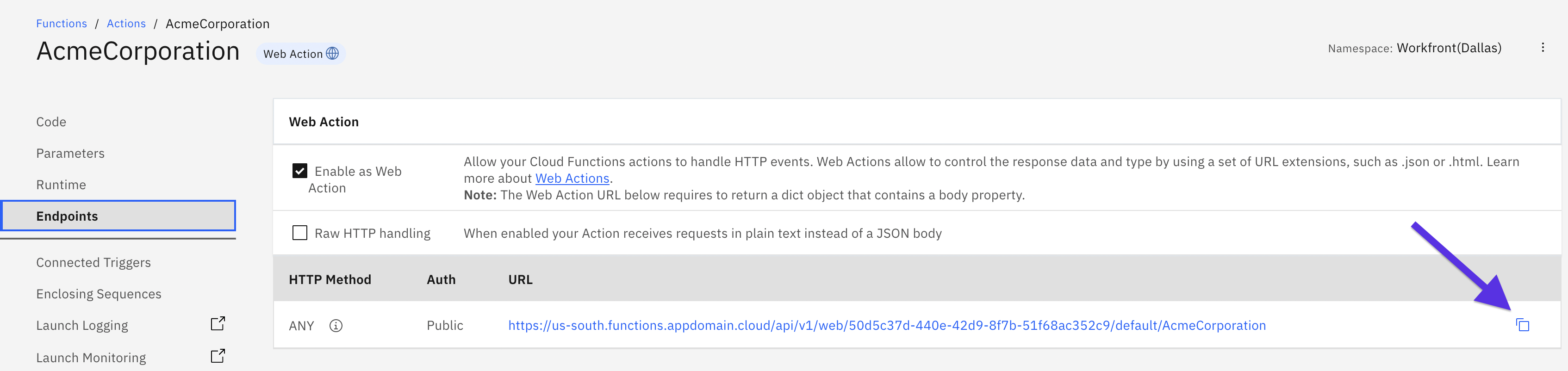
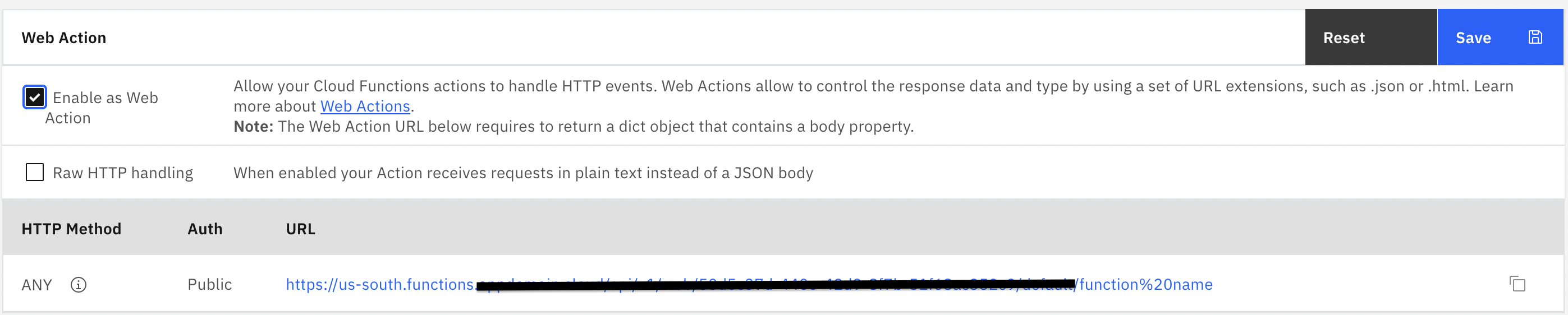
Click on the *Invoke with Parameters* and add *key, value* pair to call a webhook as shown below. Apply the change and click on *Invoke* to validate it.

Graphical user interface, application

Description automatically generated

Step 3: Now, select Endpoints from the left-hand side of the page and check Enable as Web Action and Save your changes.

Copy down the URL that’s created under Web Actions, you’ll need this later.

=

Now that you have the program/function ready, you would like to know how to integrate it or link it with your Watson Assistant. This is where the WebHooks come in to play.

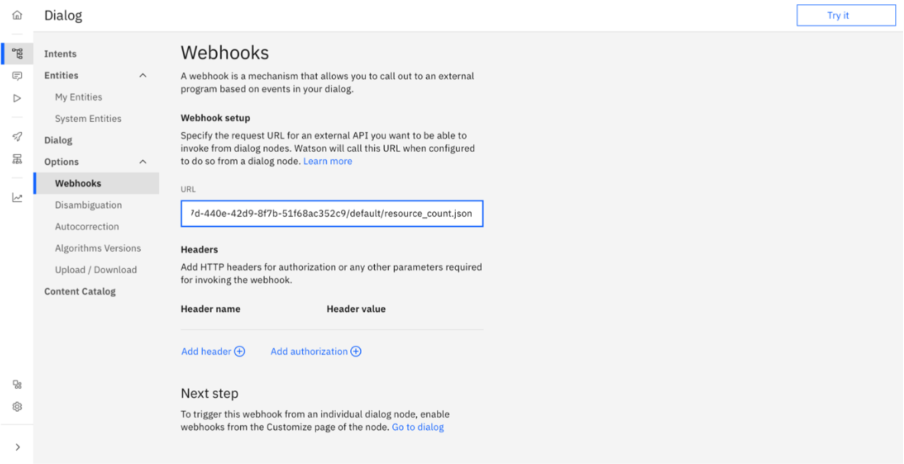
Webhooks are used to bridge two applications/programs/functions which gets triggered by a specific event and transfer data on a near real time basis.

The webhook mechanism allow program to call an external program. When used in a dialog skill, a webhook gets triggered at the moment Watson Assistant processes a node for which webhook is enabled. The webhook collects the specified data or the data that the program collects from the user during the conversation and stores in context variables. It sends the data as part of a HTTP POST request to the URL you specify as part of your webhook definition. The URL that receives the webhook is the listener. It performs a predefined action on the information passed and returns a desired response.

Let’s see how to enable it.

Step 1 — Open Watson Assistant > Dialogs > Options > Webhooks:

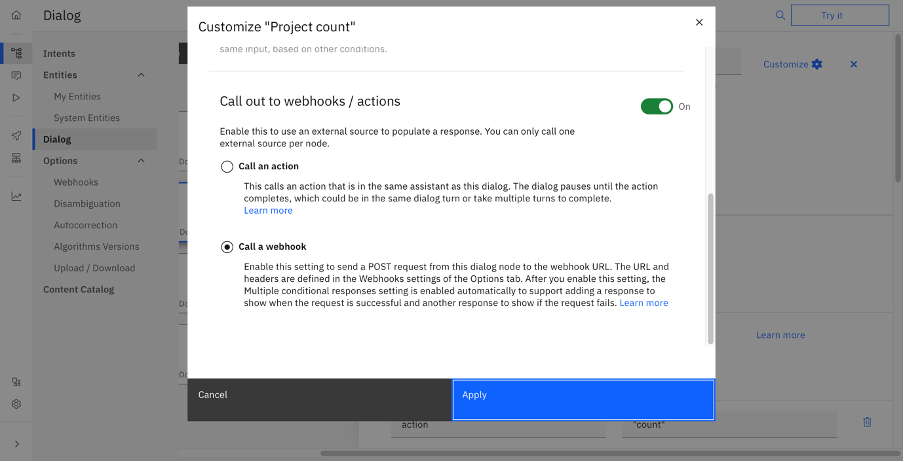
You will land on the window which looks like the one below. Now, we need to copy the endpoint of a web action we created above and paste it in the URL field along with the “.json” suffix as shown in the below screenshot:



Step 2 — Build Dialog:

Now, let’s create the conversational part. Navigate yourself to the Dialog section and click on Create Dialog. You will see how a basic dialog tree builds up. Create a new dialog node by clicking on Add Node.

Go to the dialog node you just created and click on *Customize* on the top right corner with gear icon. Now enable *Call out to webhooks / actions* and select the *Call a webhook* as shown below:



Step 3 — Define Responses:

Select the intent you want to use and provide the key, value pair for a webhook call:

Graphical user interface, text, application, email

Description automatically generated

After that name the returning variable.

Values from the repones can be extracted using the following code:

<? $webhook\_result\_1.extract.data.count ?>

Graphical user interface, text, application, email

Description automatically generated

Step 4 — Check Result:

You can easily try out your assistant by clicking on “Try it” on the right side. Pose different questions and notice the result! Have a look at the structure $webhook\_result\_1 by clicking on “Manage Context” in the Try Out Pane.

Graphical user interface, application

Description automatically generated

Now you have successfully integrated third party App/API with your Assistant, Cheers!!

See the use case in action:

Asset:<https://techzone.ibm.com/collection/watson-assistant-for-backend-data-access>

Demo:<https://htmlpreview.github.io/?https://github.com/jaypandyaibm/WAWFuction.github.io/blob/main/AcmeCorp.html>

Now Your turn