# IBM Training

## Student Exercises

## Lab-3: Create a COVID-19 Chabot

## Hands-On Lab

# Table of Contents

# Introduction

This lab will build a chatbot to respond to questions about COVID-19. Watson Assistant and Watson Discovery services from IBM will be used to build the chatbot.

# Objectives

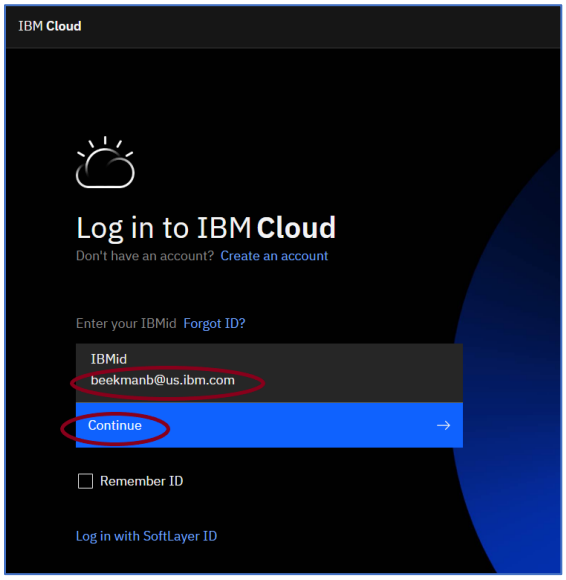
The goal of this lab is to familiarize the user with the Watson Assistant service. Watson Assistant is IBM's AI offering that lets you build, train, and deploy conversation interactions into any application, device, or channel. Watson Assistant can be deployed on any cloud or on-premises environment.

After completing this lab, you will be familiar with these features of Watson Assistant and IBM Cloud.

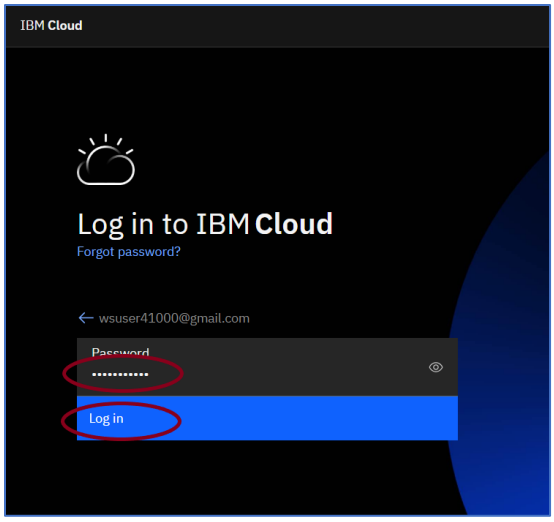
1. Provision an instance of Watson Assistant
2. Add a dialog skill to your Watson Assistant instance
3. Connect your Watson Assistant with Watson Discovery
4. Create Cloud Functions
5. Integrate data sources via a Watson Assistant webhook

# Exercise 1: Create a Watson Assistant Instance

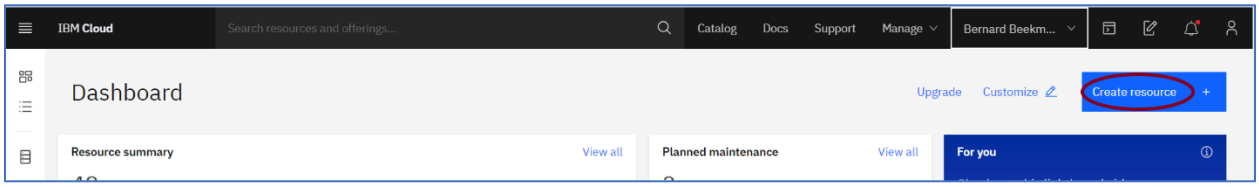
1. Log into your IBM Cloud account by typing in the url **cloud.ibm.com** in your Firefox or Chrome browser.
2. Enter your **IBMid** and click **Continue.**



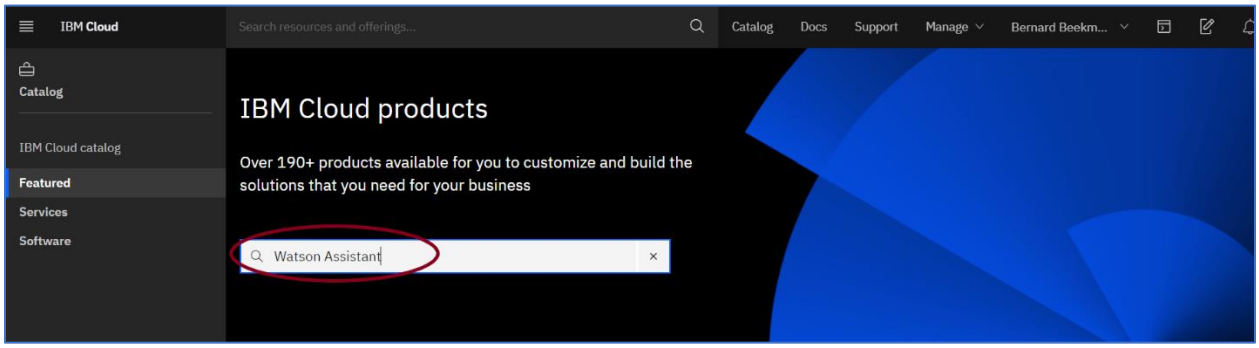

3. Enter your **Password** and click **Log in**.




4. Click **Create Resource.**

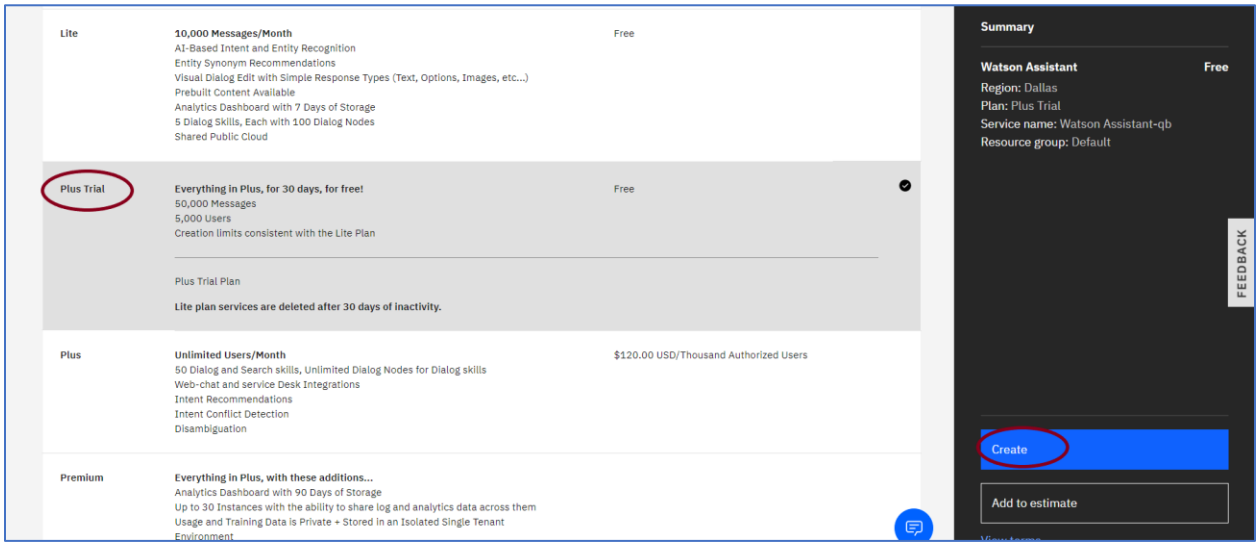



5. Enter **Watson Assistant** and click the <Enter> key.

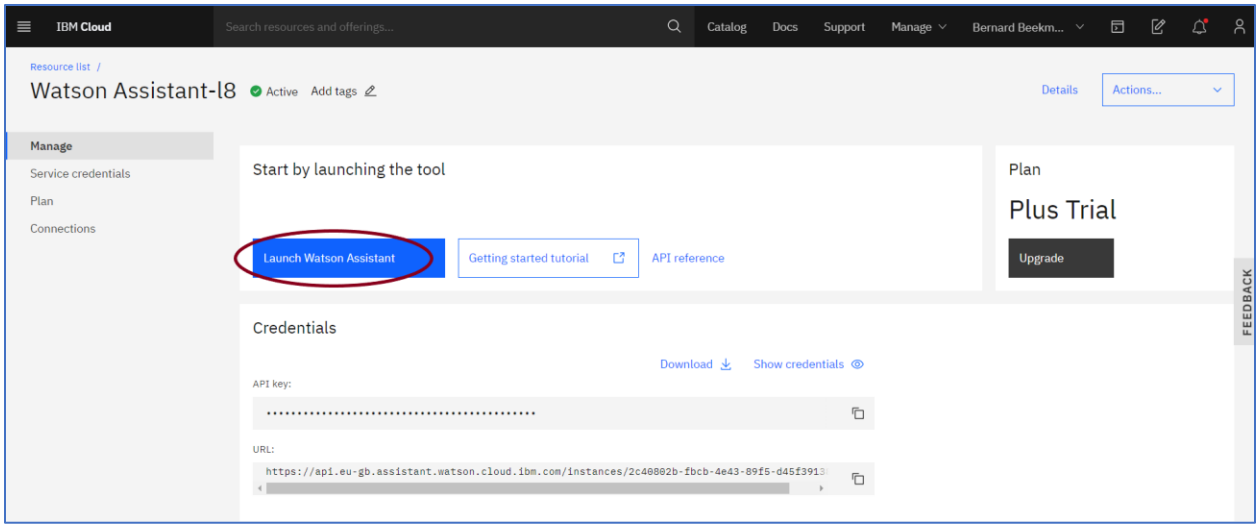



6. Click on **Watson Assistant**.

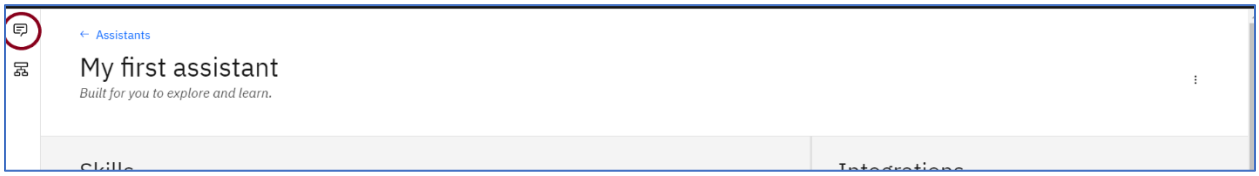

7. Click on the **Plus Trial** and click **Create.**



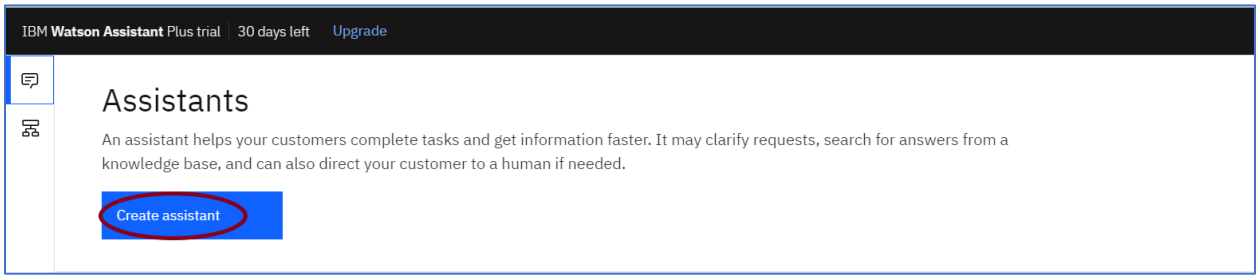

8. Click on **Launch Watson Assistant**.



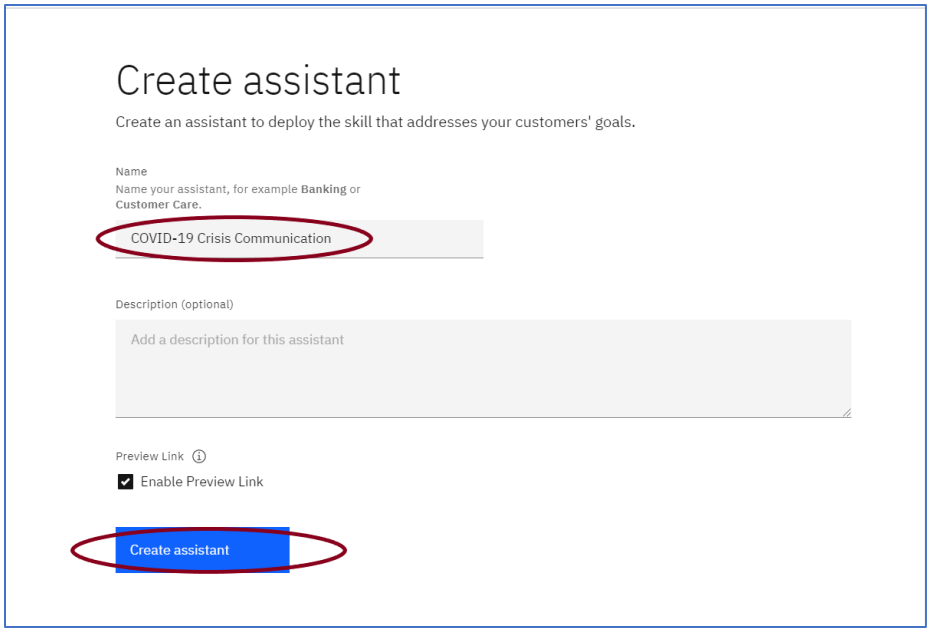

9. The My first assistant is created automatically. Click on the assistant icon  to create a second assistant.




10. Click on **Create assistant**.




11. Enter **COVID-19 Crisis Communication** for the **Name** and click **Create assistant**.

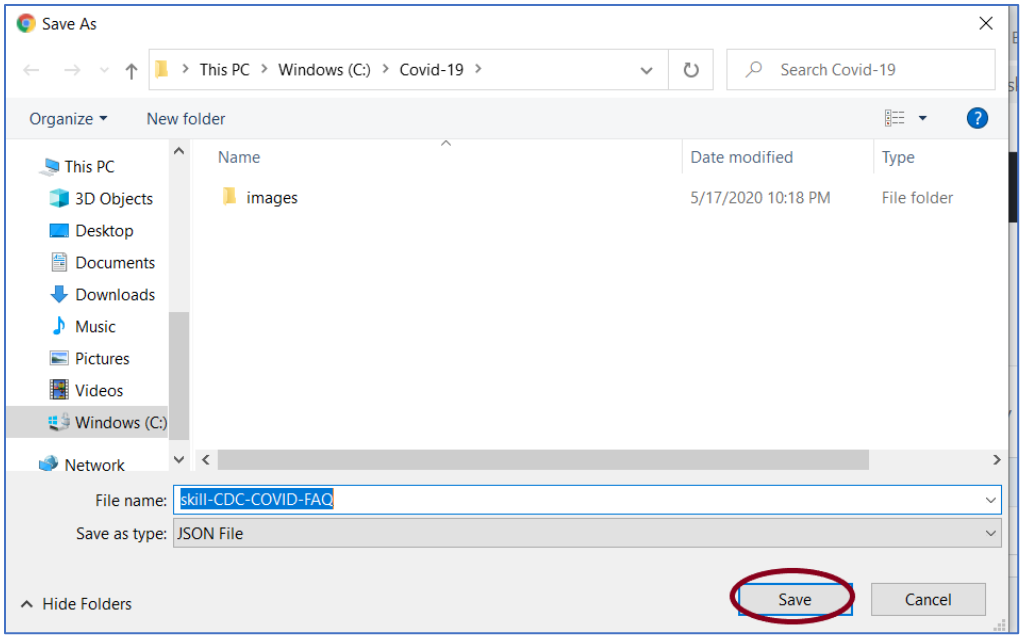



## Exercise 2: Download the COVID-19 FAQ file.

In this exercise, you will download a COVID-19 FAQ file. This file will contain the configuration for Watson Assistant to be able to respond to COVID-19 inquiries. The file consists of pre-defined intents, entities, and dialogs.

1. Click here to download the FAQ file.
2. Right-click on **Raw,** then click on **Save link as …**


3. Navigate to the location that you want to save the file then click **Save**.




Intents, Entities, and Dialogs are discussed in more detail below to explain the content of the JSON file and how intents, entities, and dialogs are used to by Watson Assistant to implement a chatbot. The lab steps continue with Exercise 3: Create a Watson Assistant Skill.

## Intents

***Intents*** are purposes or goals that are expressed in a customer's input, such as answering a question or processing a bill payment. By recognizing the intent expressed in a customer's input, the Watson Assistant service can choose the correct dialog flow for responding to it.

In planning the intents for your application, consider what your customers might want to do, and what you want your application to be able to handle on their behalf. For example, you might want your application to help your customers make a purchase. If so, you can add a `#buy_something` intent. (The `#` that is added as a prefix to the intent name helps to clearly identify it as an intent.)

After you decide which business requests that you want your application to handle for your customers, you must teach Watson about them. For each business goal (such as `#buy_something`), you must provide at least 5 examples of utterances that your customers typically use to indicate their goal. For example, `I want to make a purchase.`

Ideally, find real-world user utterance examples that you can extract from existing business processes. The user examples should be tailored to your specific business. For example, if you are an insurance company, a user example might look more like this, `I want to buy a new XYZ insurance plan.`

The examples that you provide are used by your assistant to build a machine learning model that can recognize the same and similar types of utterances and map them to the appropriate intent.

## Entities

***Entities*** represent information in the user input that is relevant to the user's purpose.

If intents represent verbs (the action a user wants to do), entities represent nouns (the object of, or the context for, that action). For example, when the *intent* is to get a weather forecast, the relevant location and date *entities* are required before the application can return an accurate forecast.

Recognizing entities in the user's input helps you to craft more useful, targeted responses. For example, you might have a `#buy_something` intent. When a user makes a request that triggers the `#buy_something` intent, the assistant's response should reflect an understanding of what the *something* is that the customer wants to buy. You can add a `@product` entity, and then use it to extract information from the user input about the product that the customer is interested in. (The `@` prepended to the entity name helps to clearly identify it as an entity.)

Your assistant detects entities in the user input by using one of the following evaluation methods:

### Dictionary-based method

Your assistant looks for terms in the user input that match the values, synonyms, or patterns you define for the entity.

- **Synonym entity**: You define a category of terms as an entity (`color`), and then one or more values in that category (`blue`). For each value you specify a bunch of synonyms (`aqua`, `navy`). You can also pick synonyms to add from recommendations made to you by Watson.

  At run time, your assistant recognizes terms in the user input that exactly match the values or synonyms that you defined for the entity as mentions of that entity.

- **Pattern entity**: You define a category of terms as an entity (`contact_info`), and then one or more values in that category (`email`). For each value, you specify a regular expression that defines the textual pattern of mentions of that value type. For an `email` entity value, you might want to specify a regular expression that defines a `text@text.com` pattern.

  At run time, your assistant looks for patterns matching your regular expression in the user input, and identifies any matches as mentions of that entity.

- **System entity**: Synonym entities that are prebuilt for you by IBM. They cover commonly used categories, such as numbers, dates, and times. You simply enable a system entity to start using it.

### Annotation-based method

When you define an annotation-based entity, which is also referred to as a contextual entity, a model is trained on both the *annotated term* and the *context* in which the term is used in the sentence you annotate. This new contextual entity model enables your assistant to calculate a confidence score that identifies how likely a word or phrase is to be an instance of an entity, based on how it is used in the user input.

- **Contextual entity**: First, you define a category of terms as an entity (`product`). Next, you go to the *Intents* page and mine your existing intent user examples to find any mentions of the entity, and label them as such. For example, you might go to the `#buy_something` intent, and find a user example that says, `I want to buy a Coach bag`. You can label `Coach bag` as a mention of the `@product` entity.

  For training purposes, the term you annotated, `Coach bag`, is added as a value of the `@product` entity.

  At run time, your assistant evaluates terms based on the context in which they are used in the sentence only. If the structure of a user request that mentions the term matches the structure of an intent user example in which a mention is labeled, then your assistant interprets the term to be a mention of that entity type. For example, the user input might include the utterance, `I want to buy a Gucci bag`. Due to the similarity of the structure of this sentence to the user example that you annotated (`I want to buy a Coach bag`), your assistant recognizes `Gucci bag` as a `@product` entity mention.

  When a contextual entity model is used for an entity, your assistant does *not* look for exact text or pattern matches for the entity in the user input but focuses instead on the context of the sentence in which the entity is mentioned.

  If you choose to define entity values by using annotations, add at least 10 annotations per entity to give the contextual entity model enough data to be reliable.

## Dialogs

The **dialog** uses the intents that are identified in the user's input, plus context from the application, to interact with the user and ultimately provide a useful response.

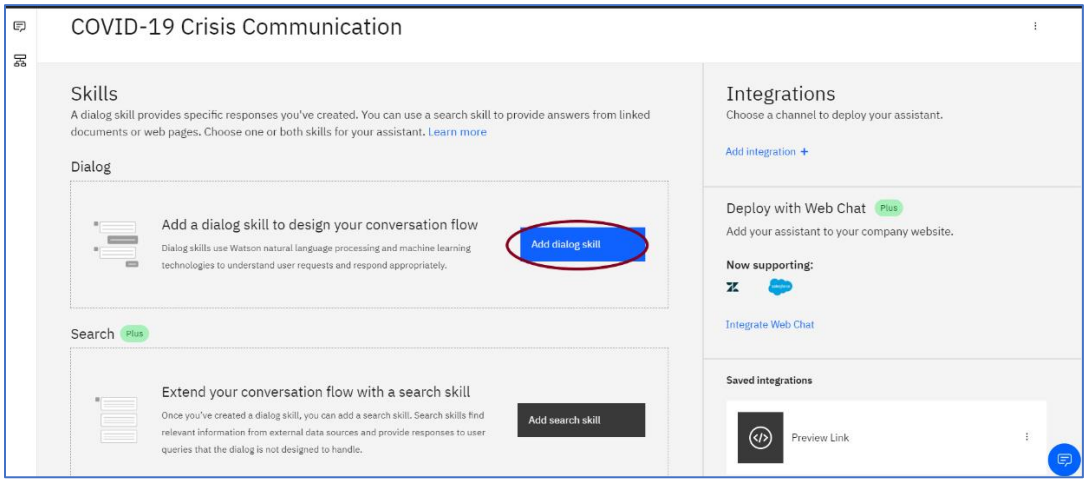
The dialog matches intents (what users say) to responses (what the bot says back). The response might be the answer to a question such as `Where can I get some gas?` or the execution of a command, such as turning on the radio. The intent and entity might be enough information to identify the correct response, or the dialog might ask the user for more input that is needed to respond correctly. For example, if a user asks, `Where can I get some food?` you might want to clarify whether they want a restaurant or a grocery store, to dine in or take out, and so on. You can ask for more details in a text response and create one or more child nodes to process the new input.

# Exercise 3: Create a Watson Assistant Skill

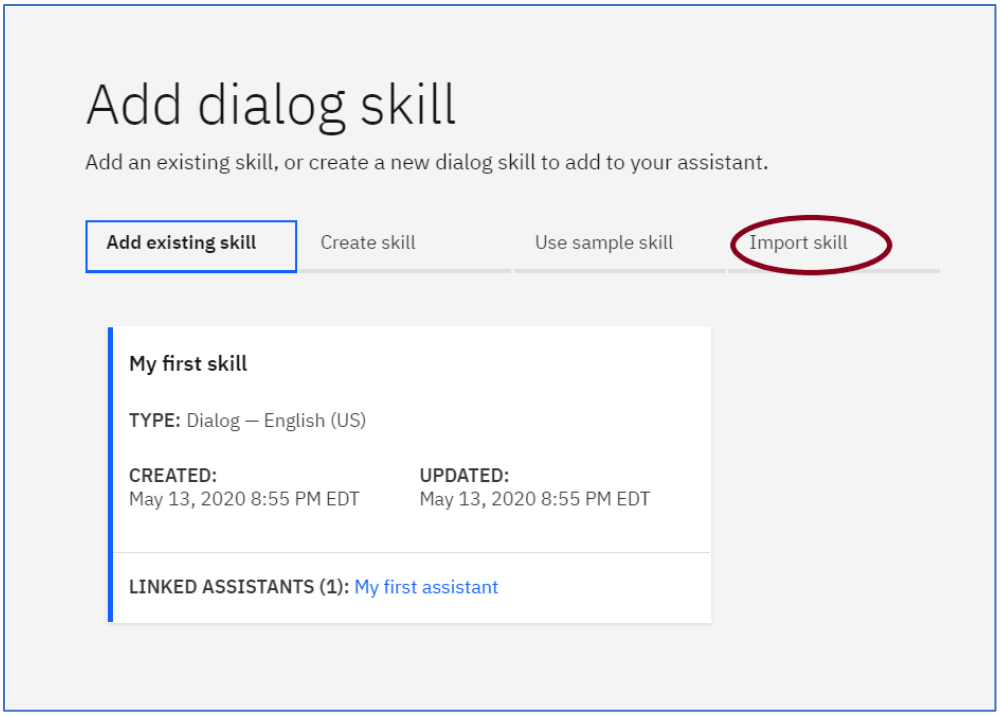
Watson assistant receives user input and routes it to the appropriate skill. Two types of skills can be created in the assistant.

- The **dialog skill** interprets the user input and directs the flow of the conversation. The dialog gathers any information it needs to respond or perform a transaction on the user's behalf. This skill understands typical questions or requests from users and answers or fulfills them by following a dialog that is scripted by you.
- The **search skill** routes complex customer queries to Watson Discovery. Watson Discovery treats the user input as a search query. It finds information relevant to the query from the configured data sources, extracts the passage, and returns it so the assistant can share the information with the user as its response.

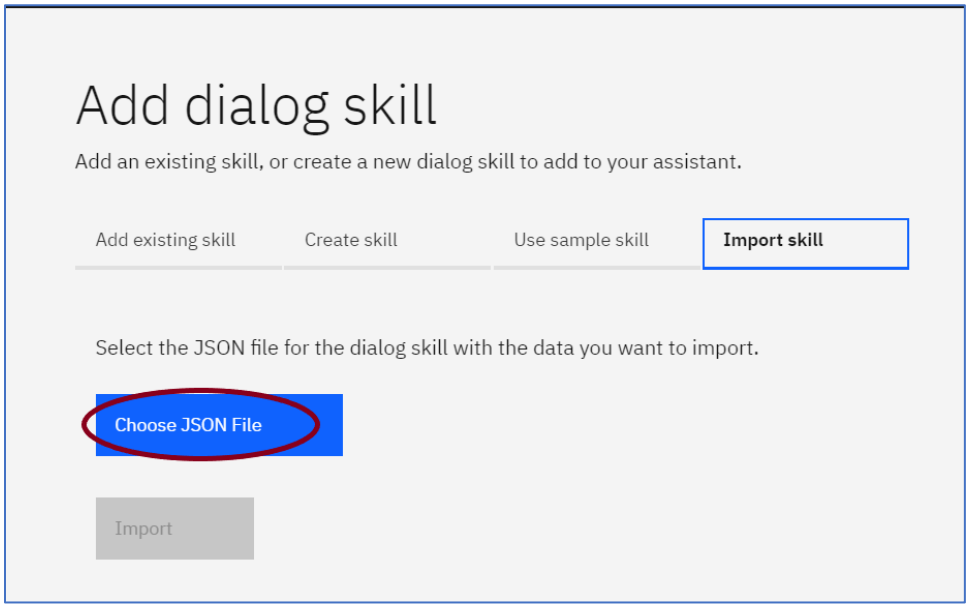
1. Click **Add dialog skill**



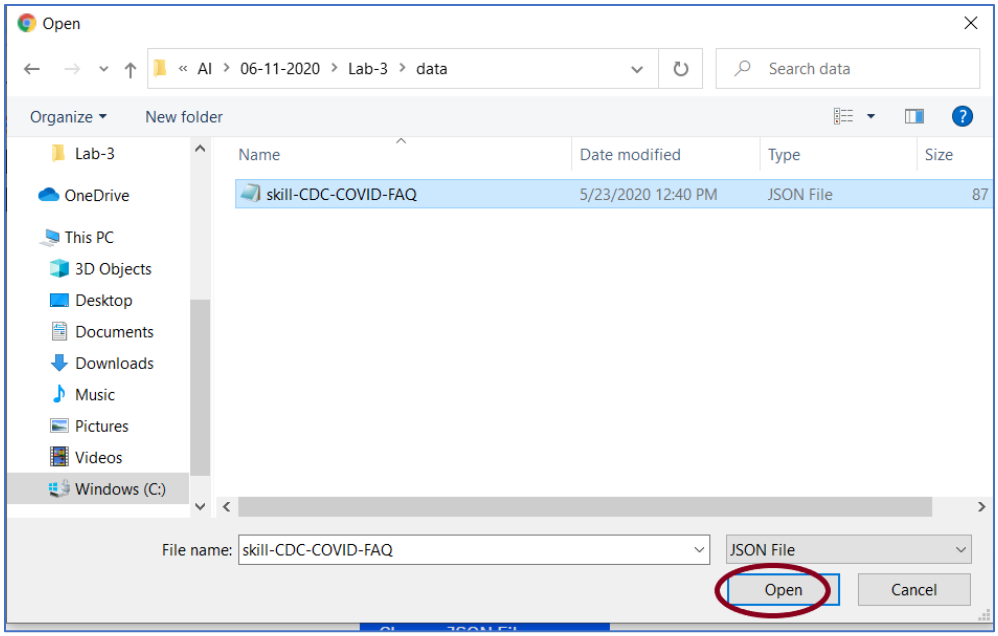

2. Click **Import skill**.



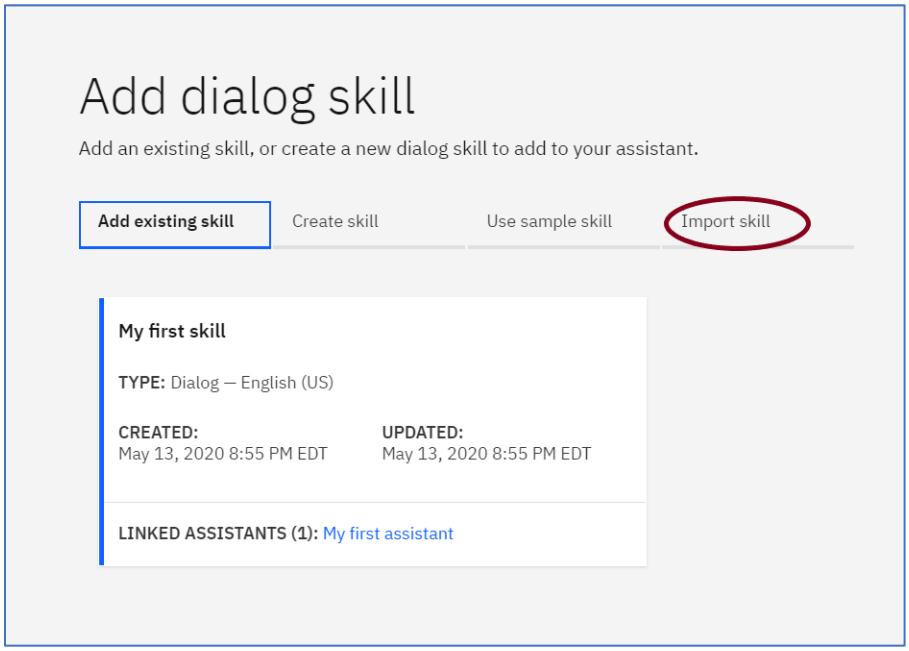

3. Click on **Choose JSON File**.



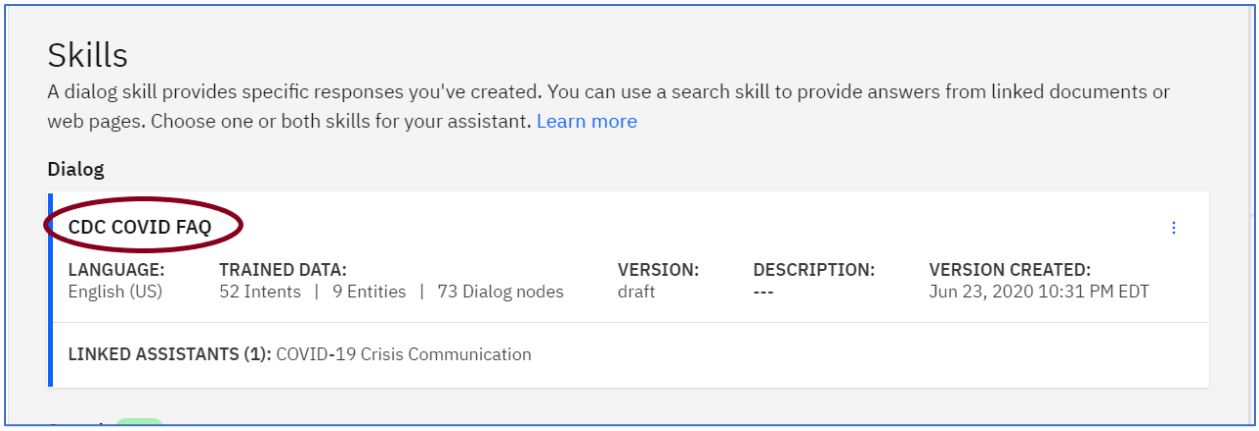

4. Navigate to the directory where you stored the JSON file, click on the file, and click



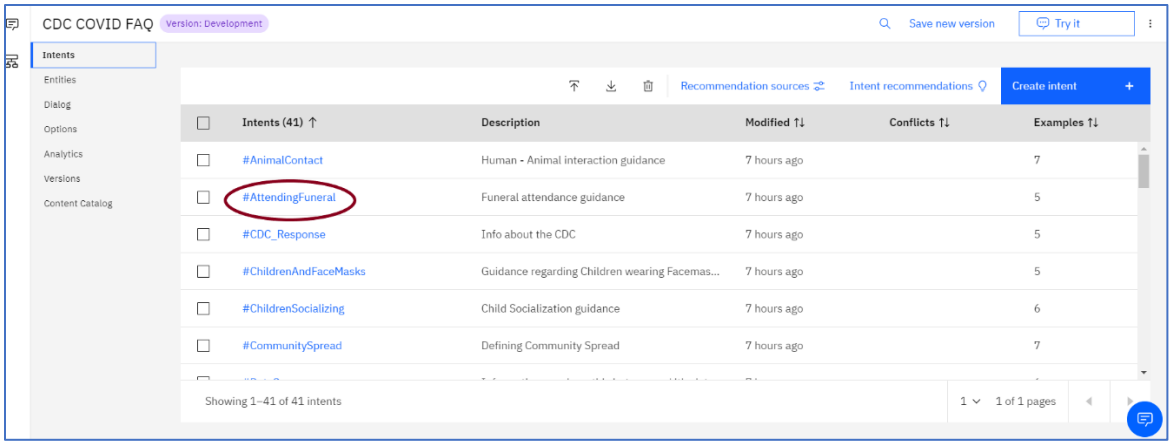

5. Click **Import**.



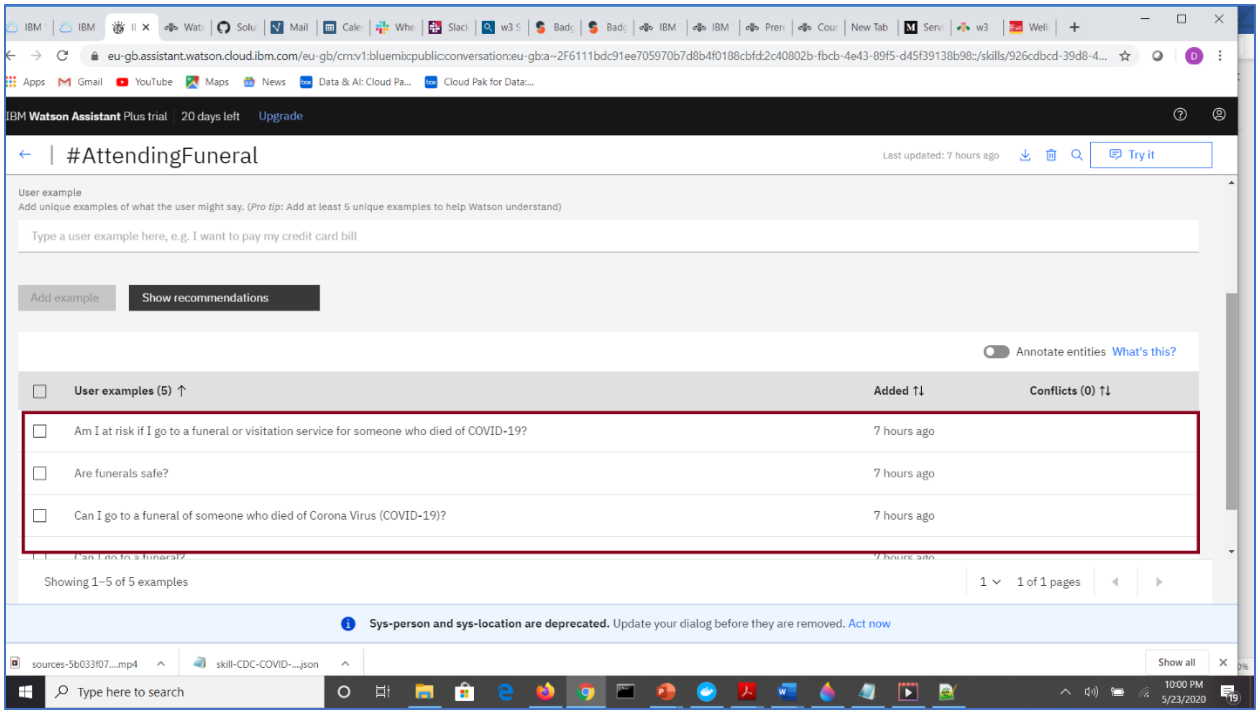

6. The file was successfully imported, and the CDC COVID FAQ skill was added. You can see there are 52 **Intents**, 9 **Entities**, and 73 **Dialog nodes**. Click on the **CDC COVID FAQ** Skill.




7. You can browse through the Intents. Click on one of them and view the examples provided to train the intent. #AttendingFuneral was selected in the figure below.

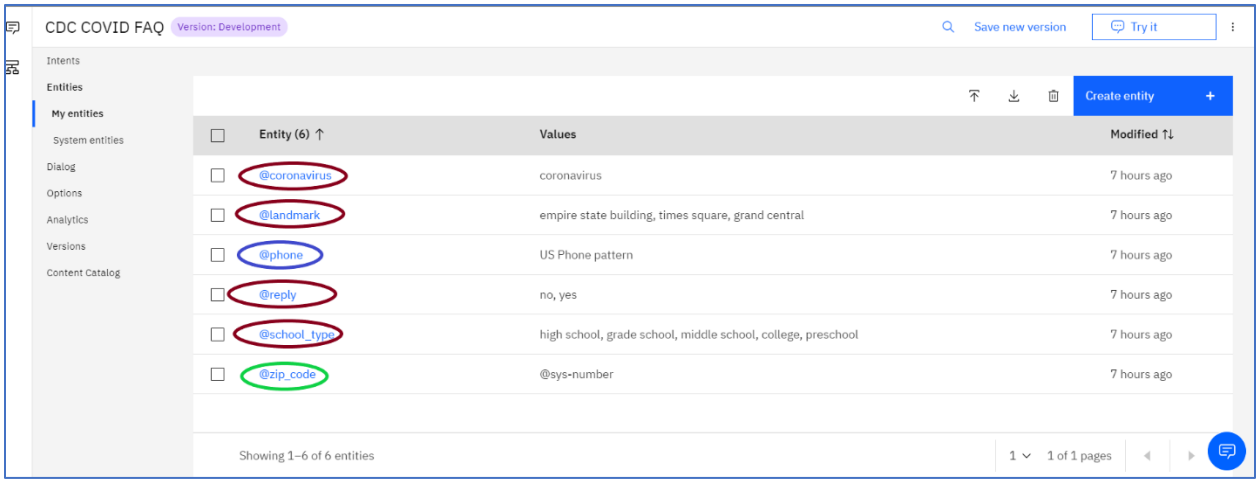



8. Scroll down to view the examples.

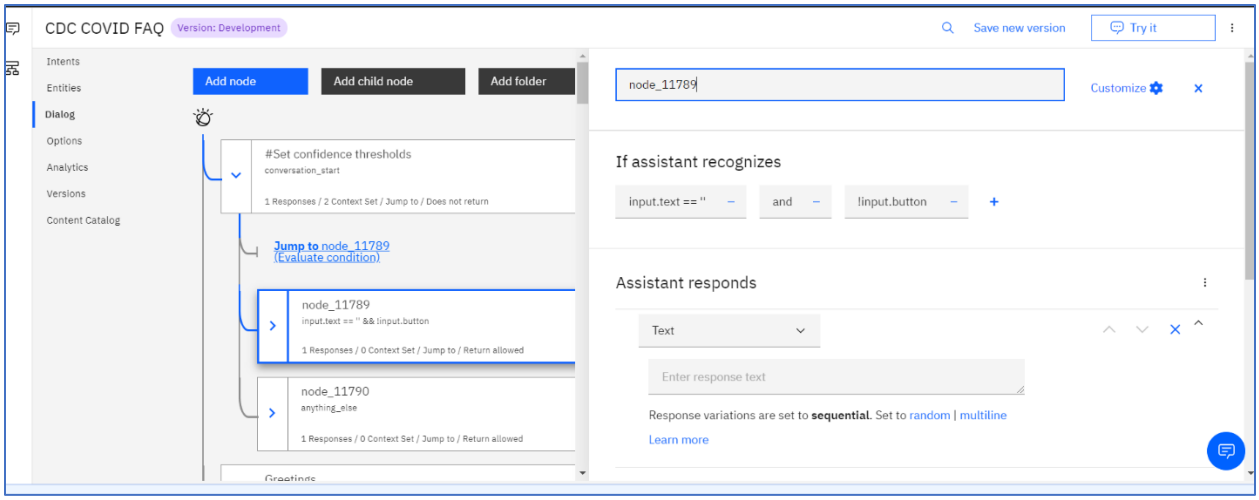



9. Click on the back arrow icon ← to return to the **Intents** page.


10. Click on **Entities.** Six (6) entities are shown. Note, there are also 3 system entities that are enabled in the **System entities** tab. That adds up to the 9 total entities documented above. The entities all use the Dictionary-based evaluation method. Four entities (shown in Maroon) use dictionaries, 1 entity (shown in Blue) uses a regular expression, and 1 entity (shown in Green) uses a system entity.




11. Click on the Dialog option. Browse through the dialog nodes to get a feel for the navigation logic.

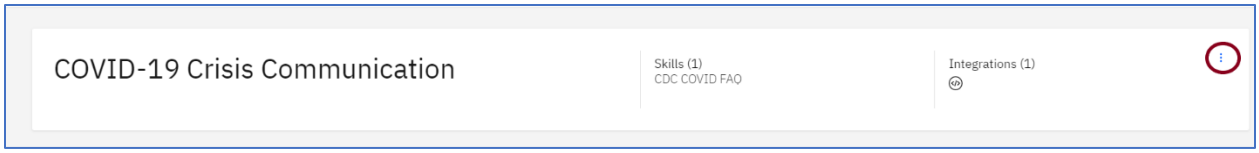



12. Click on the Assistant icon  to return to the All Assistants page.

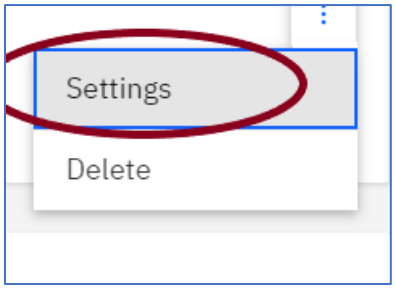

# Exercise 4: Get Assistant and Skill Identifier and API keys.

You will need identifiers and API keys to integrate the chatbox with other services.

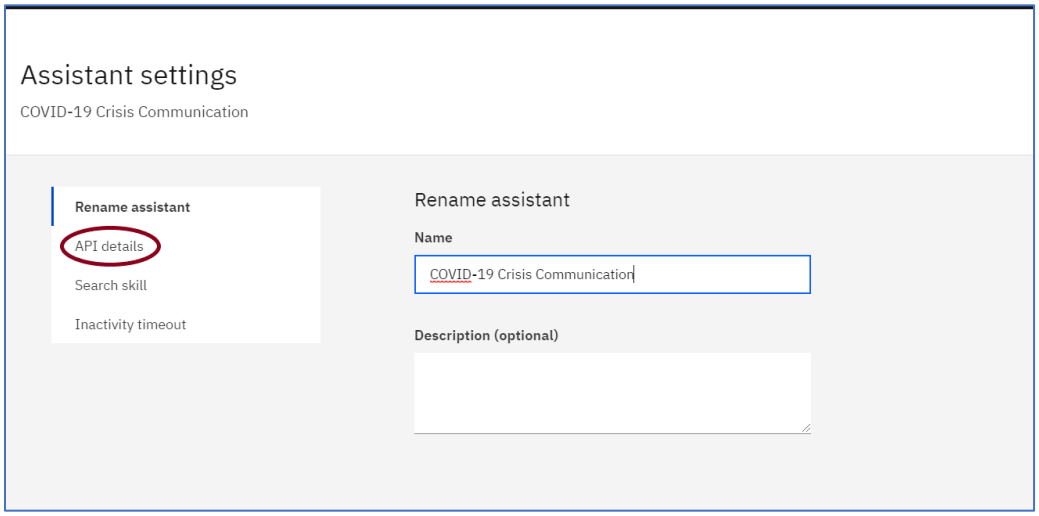
1. Click on the vertical ellipse  at the right side of the COVID-19 Crisis Communication Assistant.



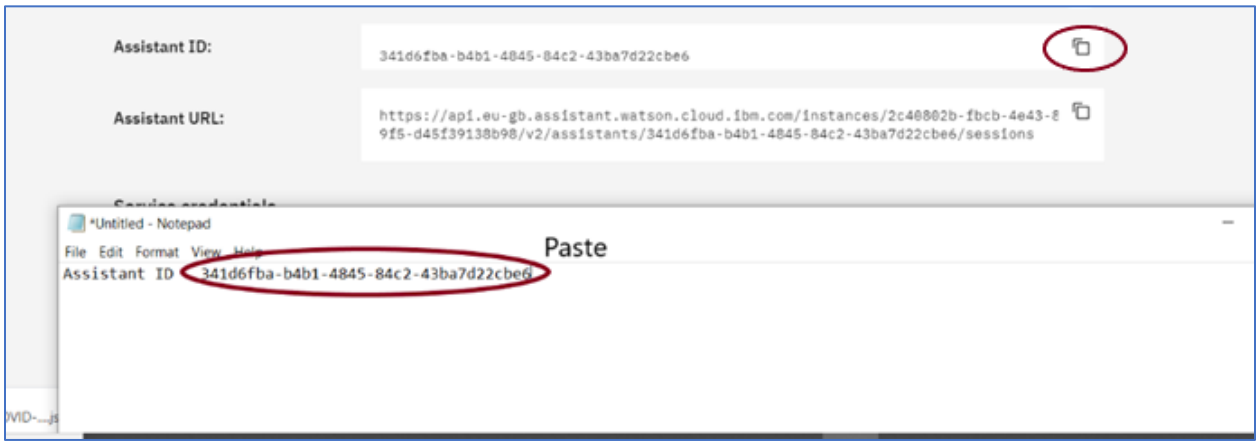
2. Click **Settings**.



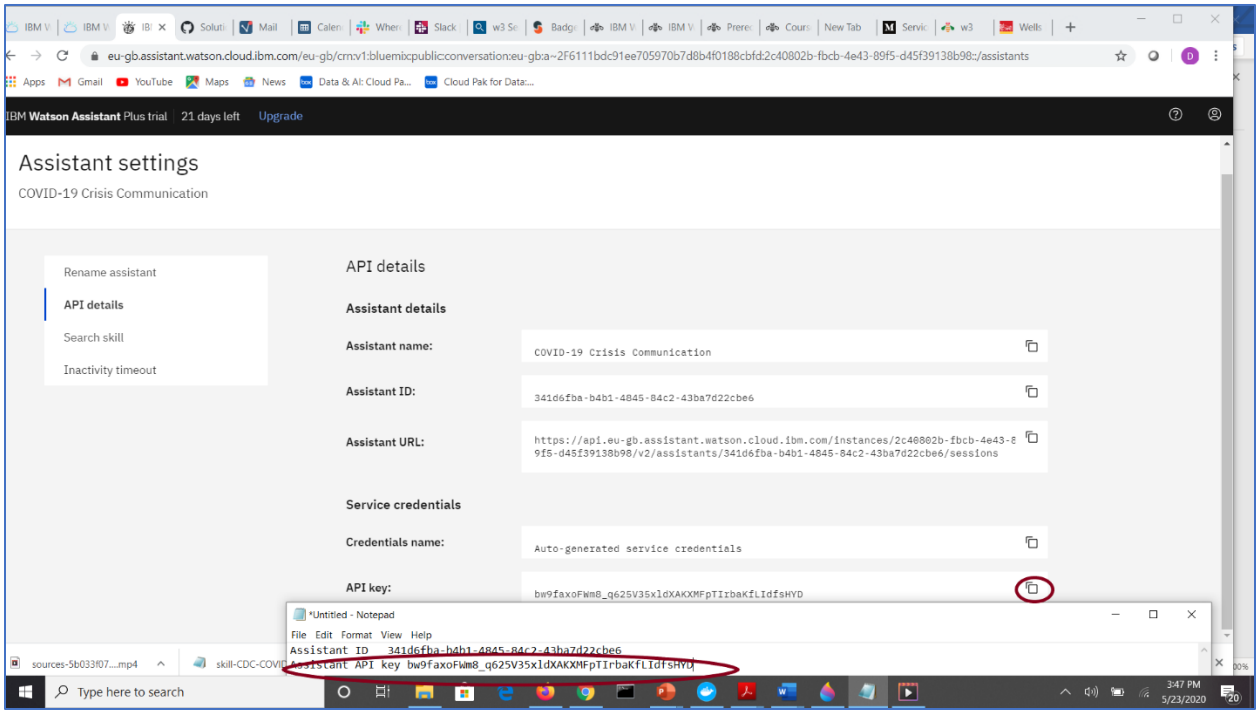

3. Click on **API details**.




4. Open Notepad or another editor that you generally use. Type in Assistant Id: and then copy and paste the **Assistant ID** in.

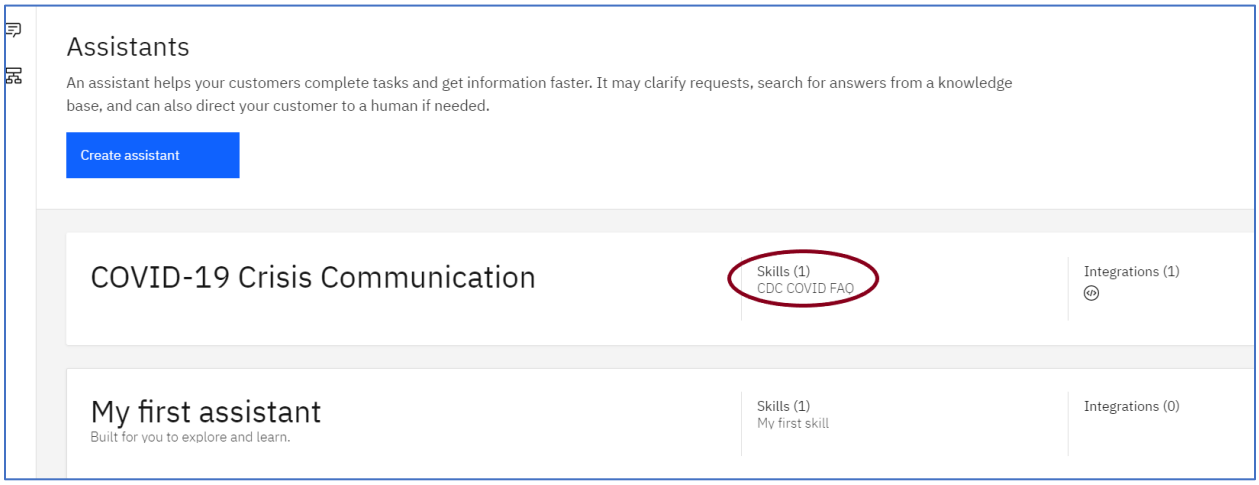



5. Type in Assistant API key, and then cut and paste the API key.

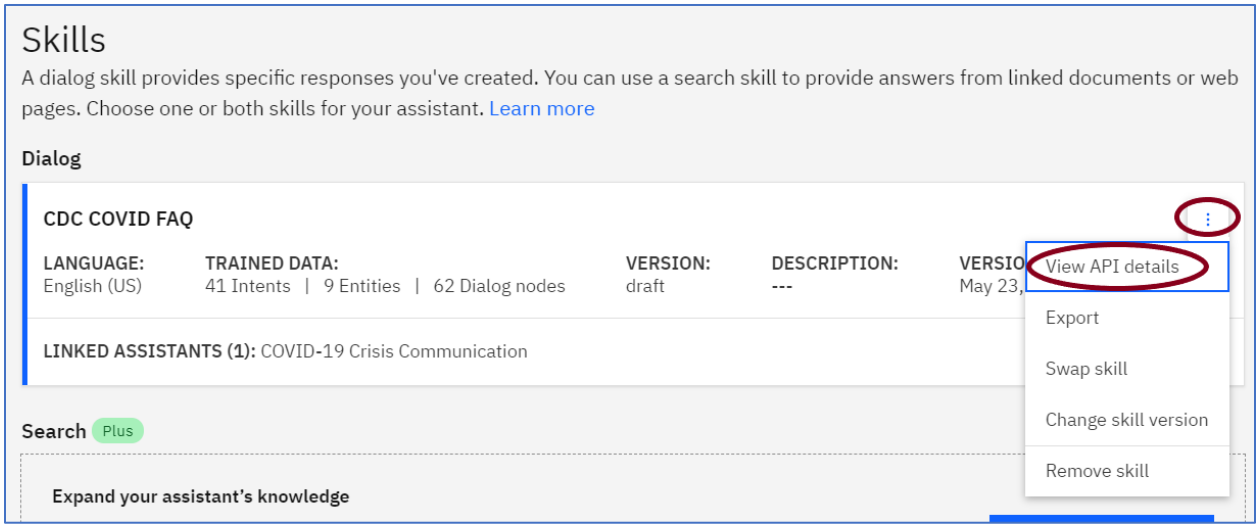



6. Close the Assistant settings page by click on the “x”.

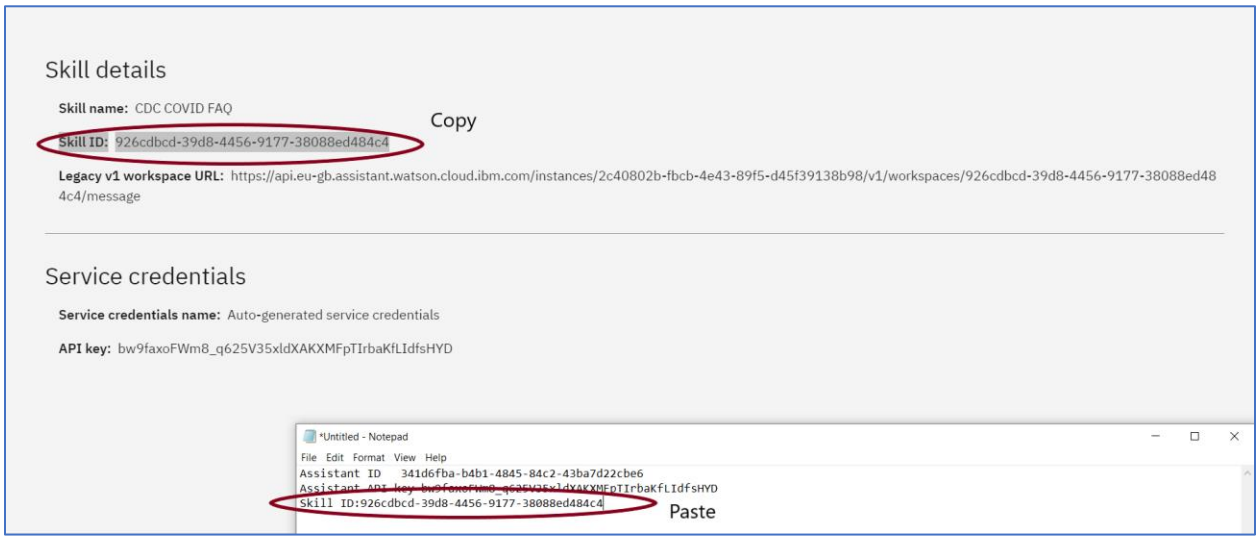

7. Click on the **CDC_COVID_FAQ Skills** link.




8. Click on the vertical ellipse ⋮ and then click on **View API details**.

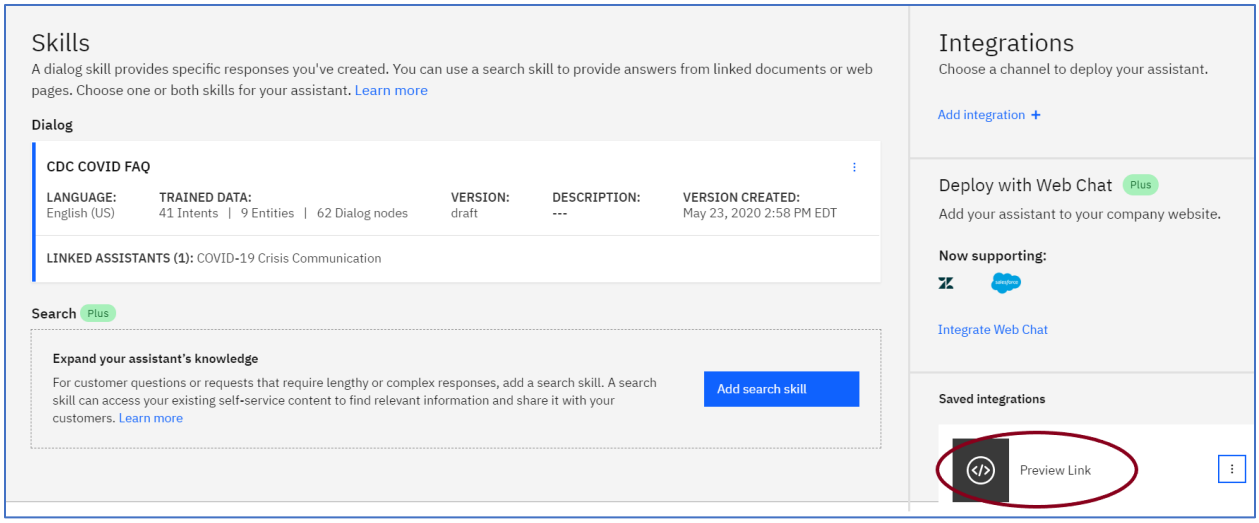



9. Copy and Paste the **Skill ID** to Notepad.

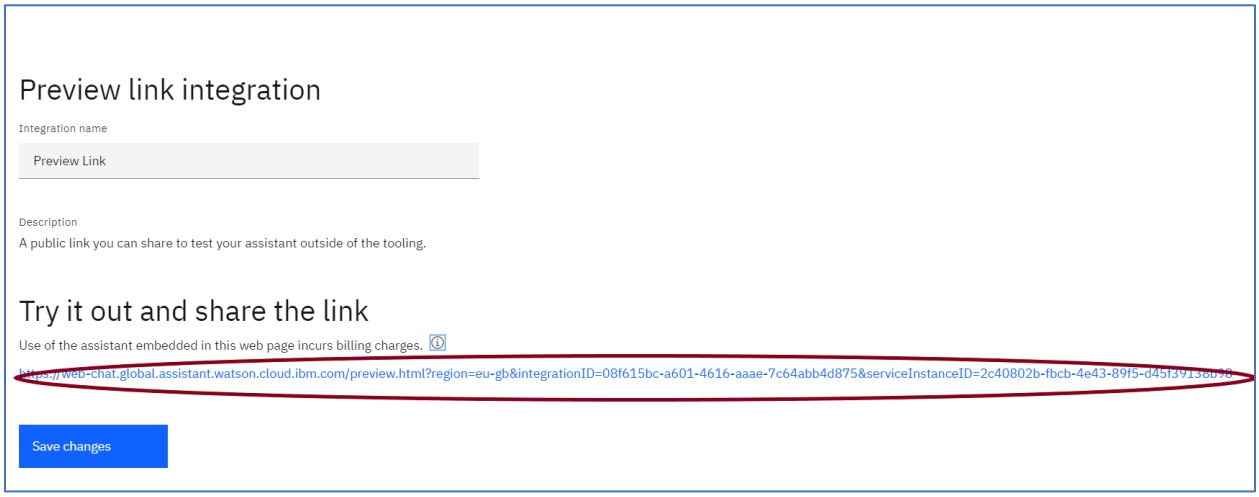



10. Close the **Skill details** page.

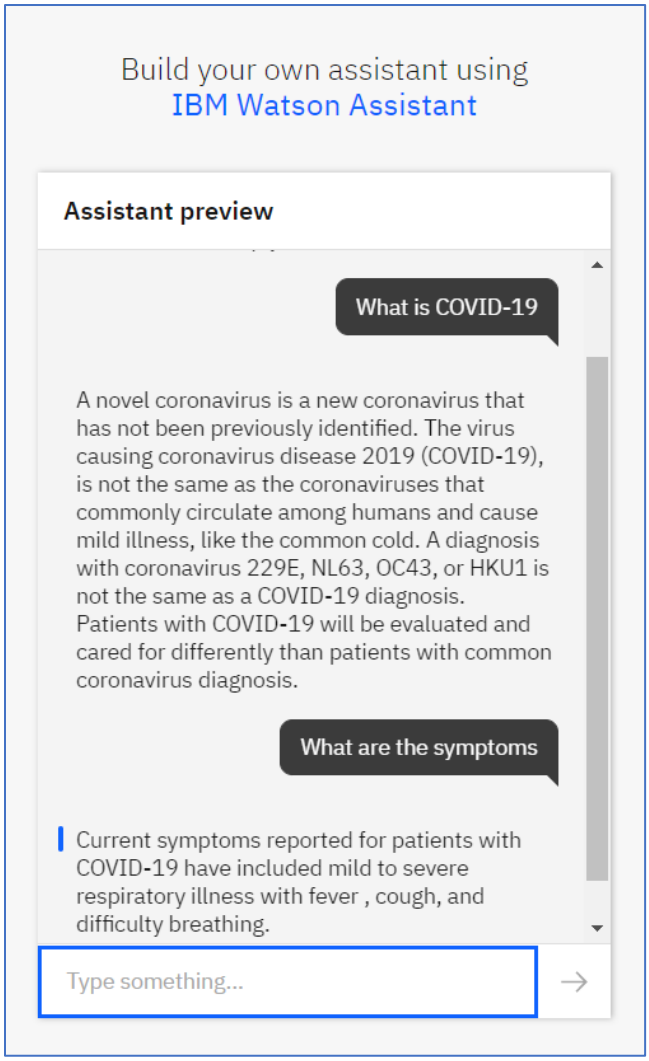

11. Click on **Preview** link.



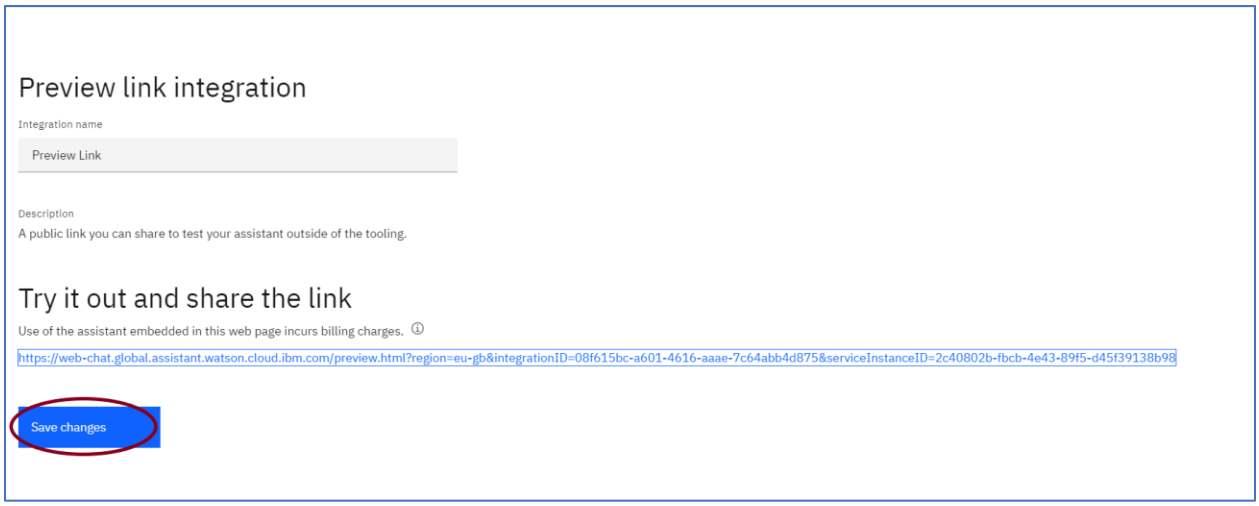

12. Click link to try out the chatbot.




13. Type in “What is COVID-19” and then click on the right arrow icon. Type in “What are the symptoms and then click on the right arrow icon. You should see the responses shown in the figure below.




14. Close the browser tab corresponding to the Assistant preview.
15. Click **Save changes**.



# Integrating Data Sources

The data that is currently driving the chatbot is static data. You can connect your Watson Assistant chatbot to data sources to query dynamic data. The following section shows you how to do this by adding webhooks to Watson Assistant.

Our crisis communication chatbot uses two different data sources:

- Watson Discovery
- COVID-19 API

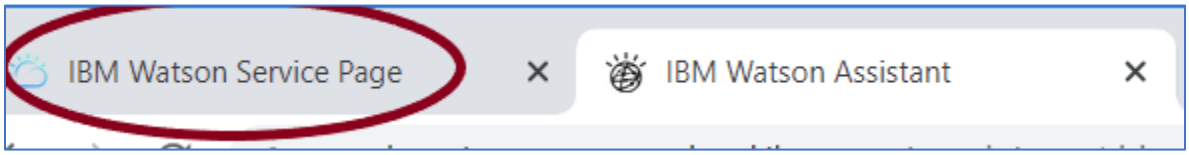
## Defining webhooks

First, let's look at what webhooks do. A *webhook* is a mechanism that allows you to call out to an external program based on something happening in your program. When used in a dialog skill, a webhook is triggered when the assistant processes a node that has a webhook enabled. The webhook collects data that you specify or that you collect from the user during the conversation and save in context variables.

The chatbot sends the data as part of a HTTP POST request to the URL that you specify as part of your webhook definition. The URL that receives the webhook is the listener. It performs a predefined action using the information that you pass to it as specified in the webhook definition and can optionally return a response.

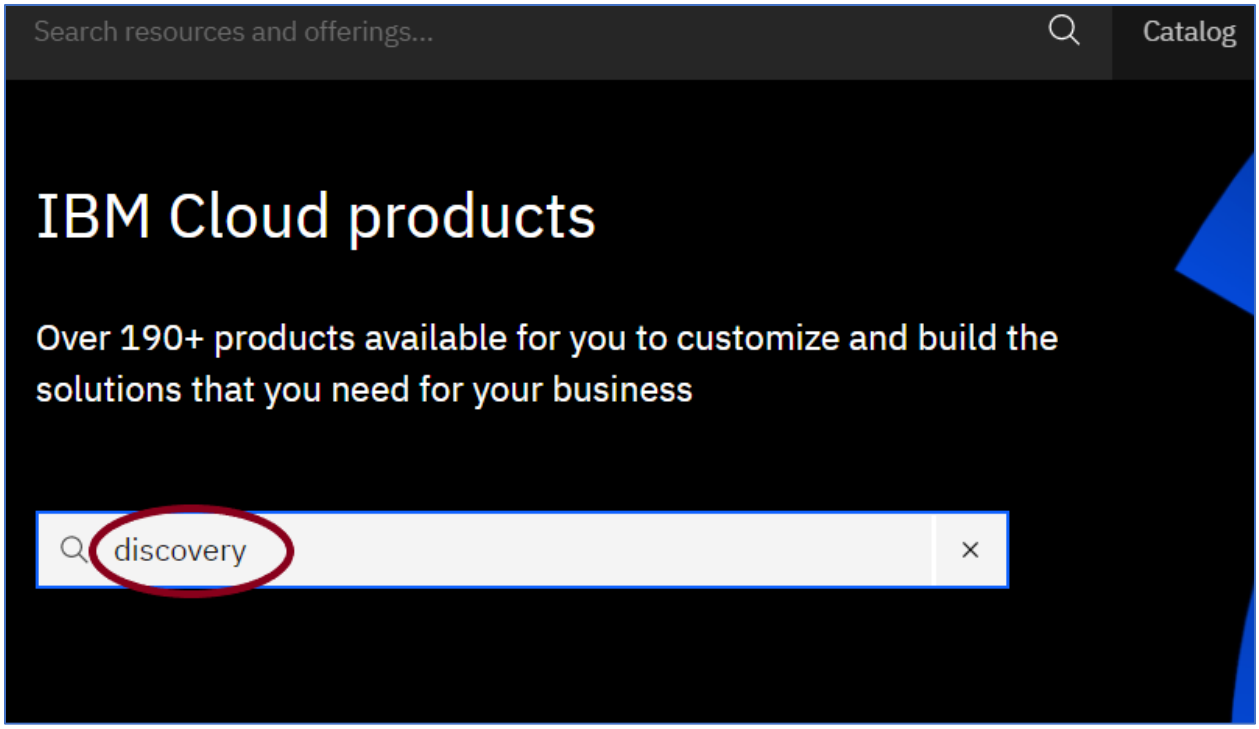
# Exercise 5: Configuring Connection to Watson Discovery

1. This lab is designed to be able to standalone. So, the assumption here is that a Discovery service needs to be created. If one already exists, please skip to step 7. Otherwise, click on the Watson Service browser tab.

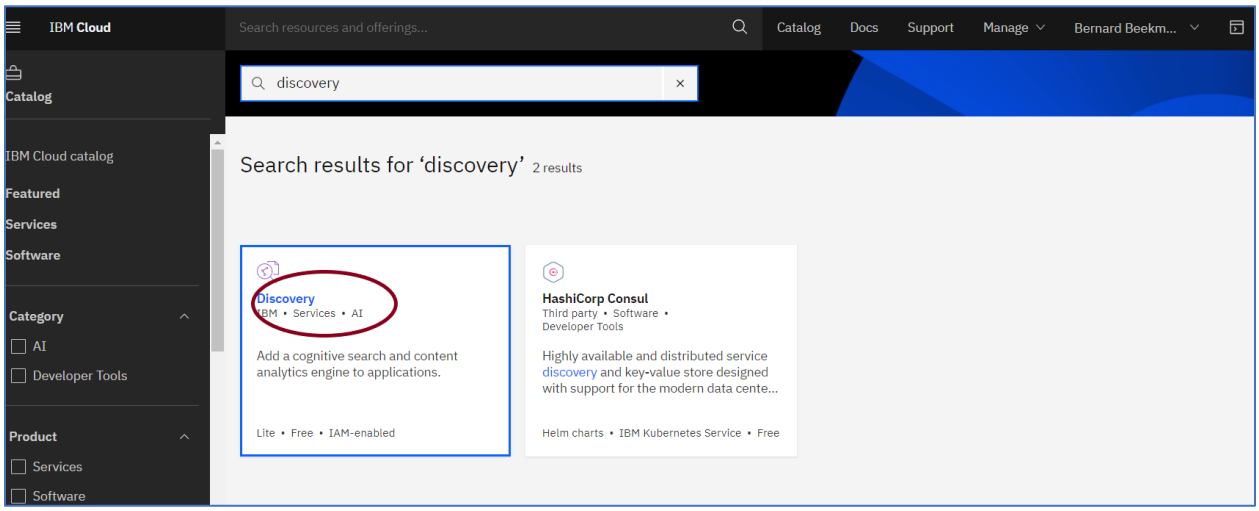



2. Click on **Catalog**.

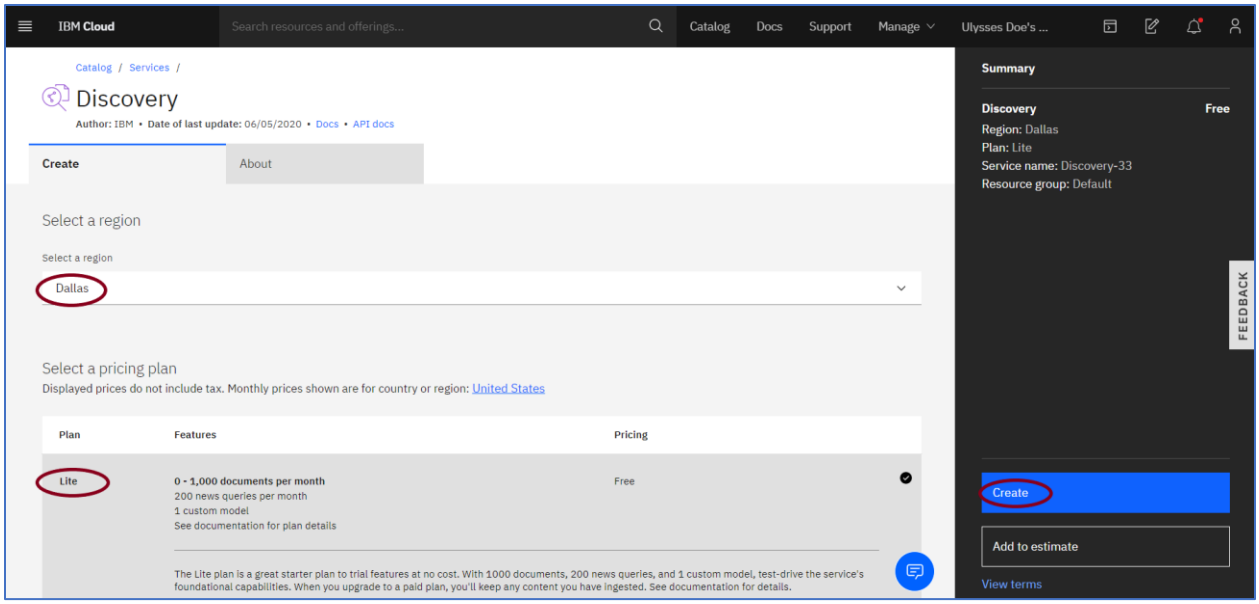

3. Enter discovery in the search box and click the <Enter> key.




4. Click **Discovery**.



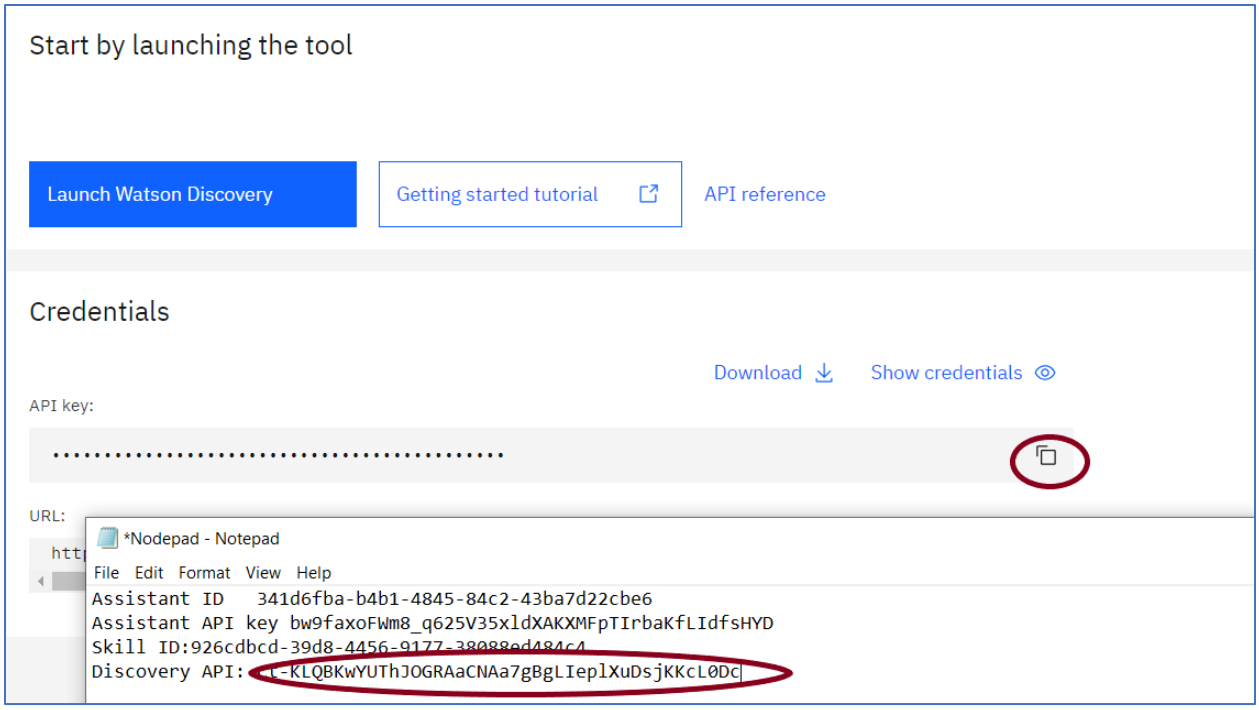

5. Click **Create** to create the Discovery instance. **MAKE SURE TO CHANGE THE DISCOVERY INSTANCE TO BE THE SAME REGION AS THE WATSON ASSISTANT INSTANCE.**



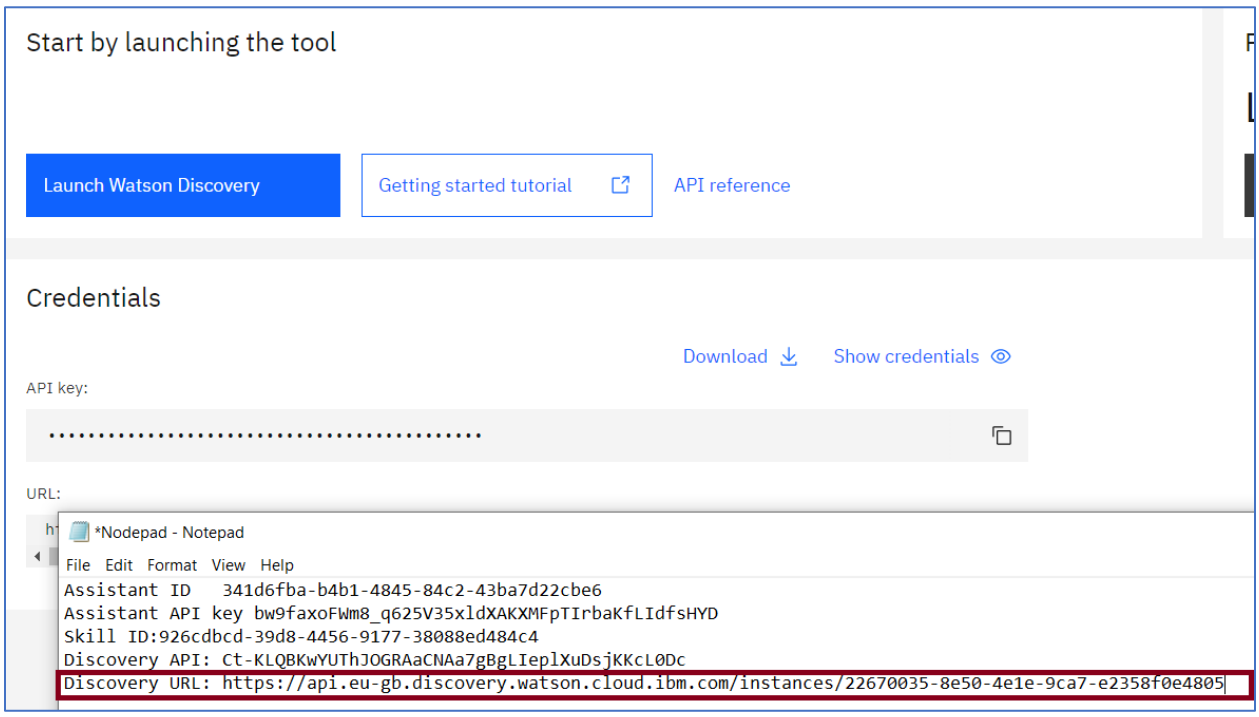

6. The service may take some time to create. The **Resource** list page is displayed with the Discovery service showing a status of Provision in progress.


7. When the service becomes Active, click on the Discovery service.

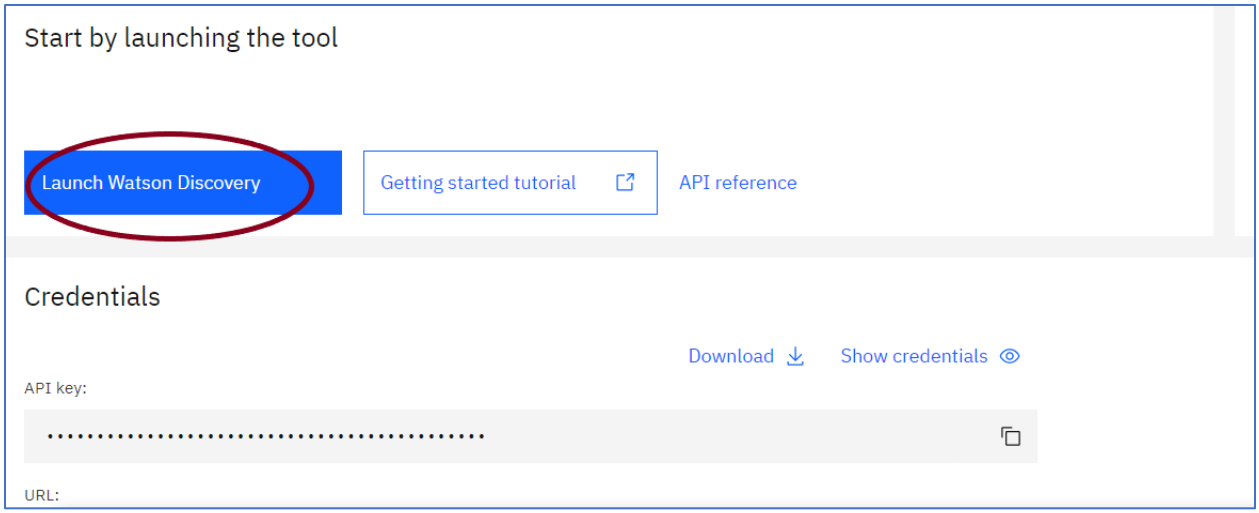

8. Copy and Paste the API key into the Notepad file. Label it Discovery API.



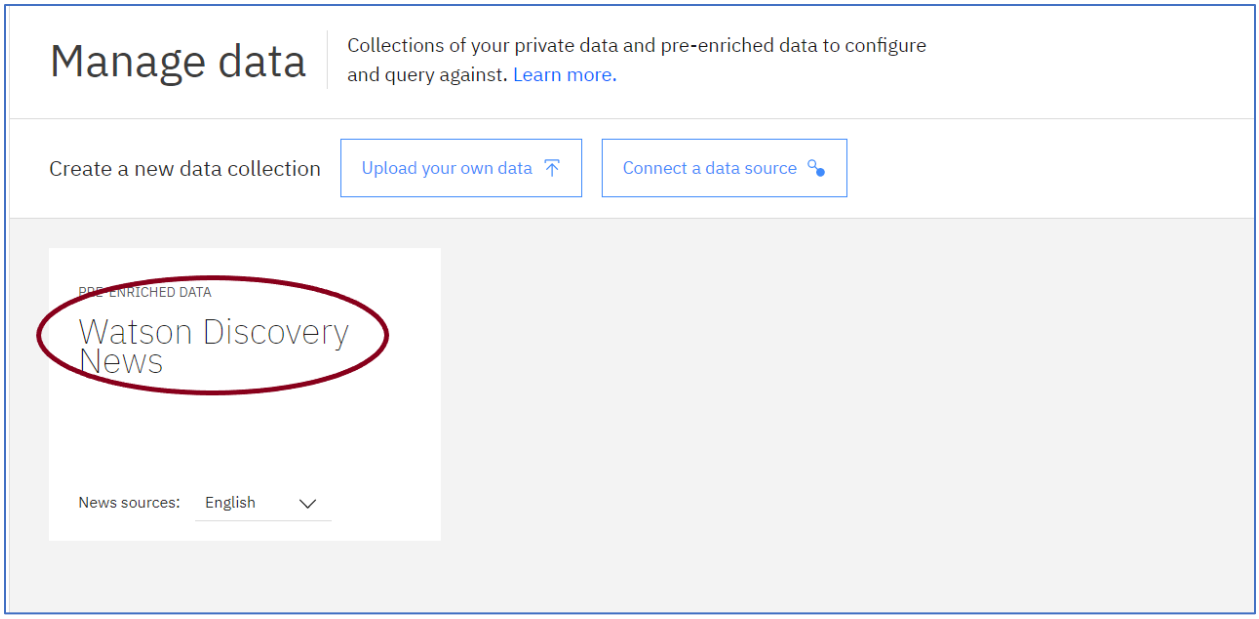

9. Copy and Paste the **URL** into Notepad file. Label it Discovery URL.



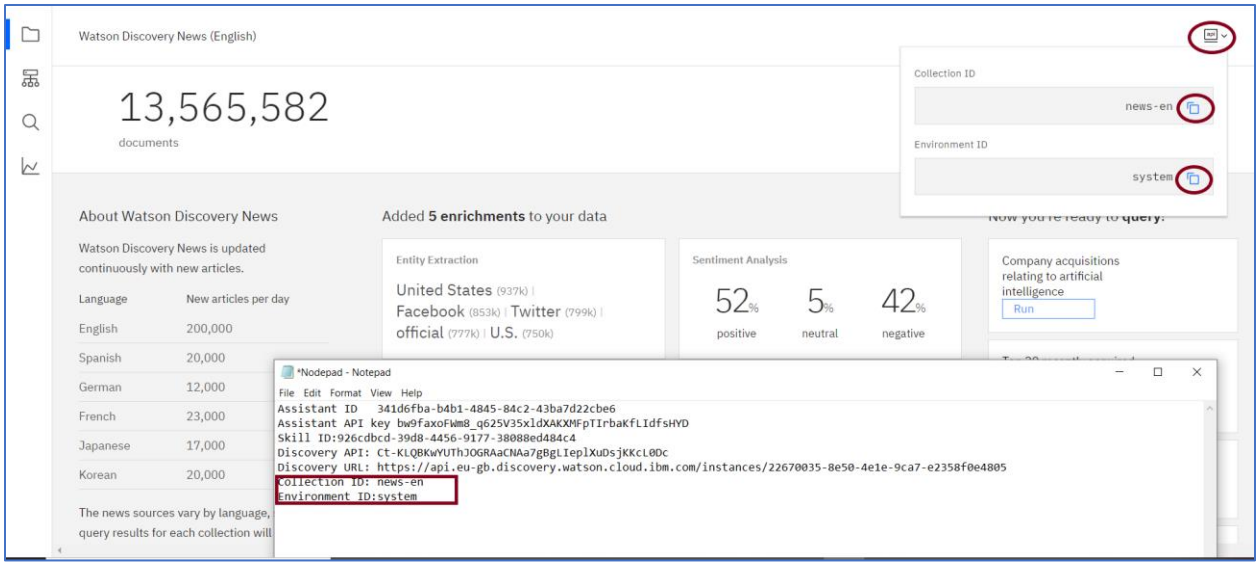

10. Click **Launch Watson Discovery**.




11. Click **Watson Discovery News** to open the Watson Discovery NEWS service, which is a prepopulated discovery dataset updated and maintained by the Watson Discovery team.



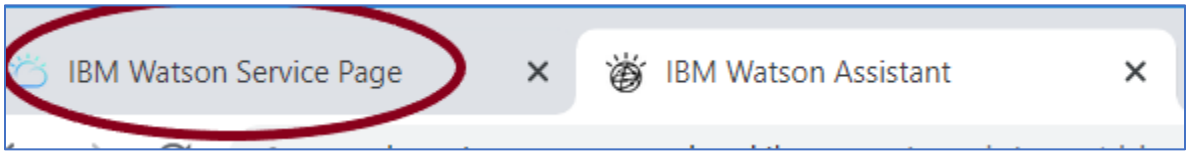

12. From the top right corner, click on the API icon  . Copy and Paste the Collection ID and the Environment ID into the Notepad file.




## Exercise 6: Create Cloud Functions

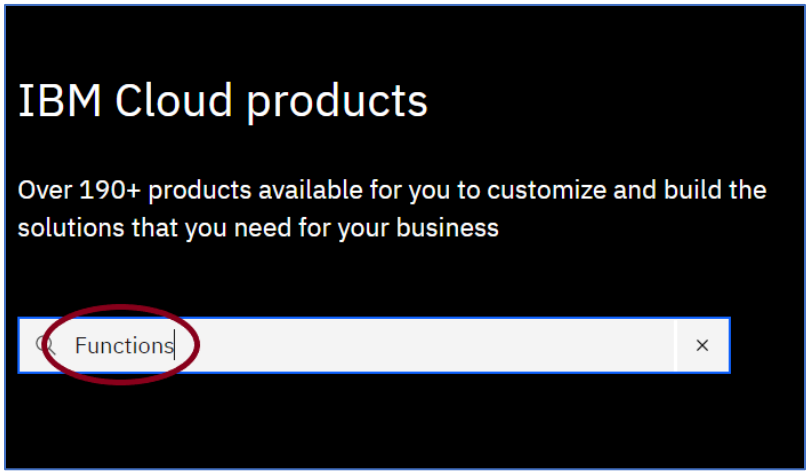
IBM Cloud Functions are a Functions-as-a-Service (FaaS) platform based on Apache OpenWhisk. You can run your application code without servers, scale it automatically, and pay nothing when not in use.

1. Click on the **Watson Service Page** browser tab.

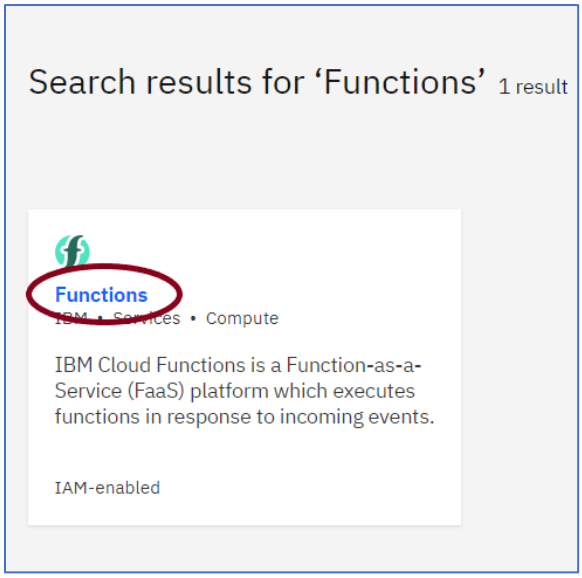



2. Click on **Catalog**.

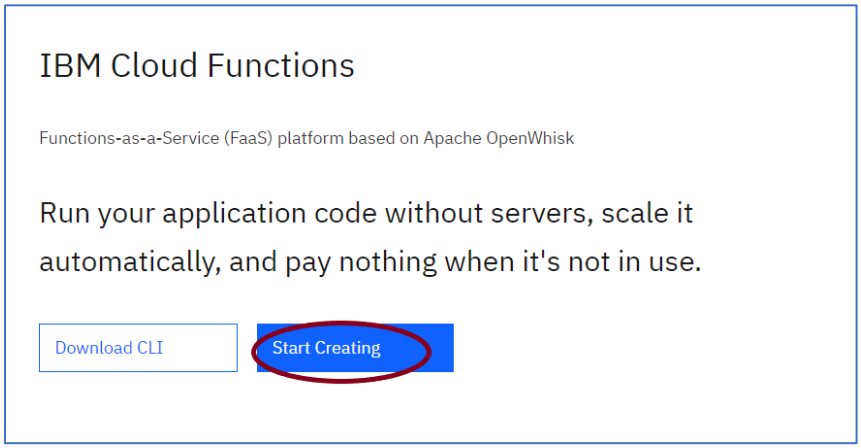

3. Type in **Functions** in the **Search** text box and press the <Enter> key.



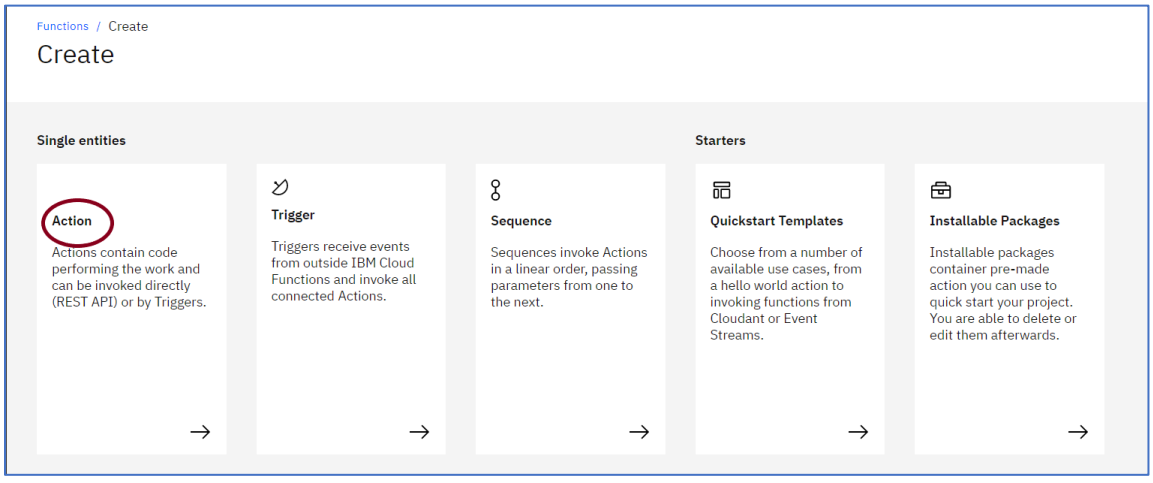
4. Click on the **Functions** tile.



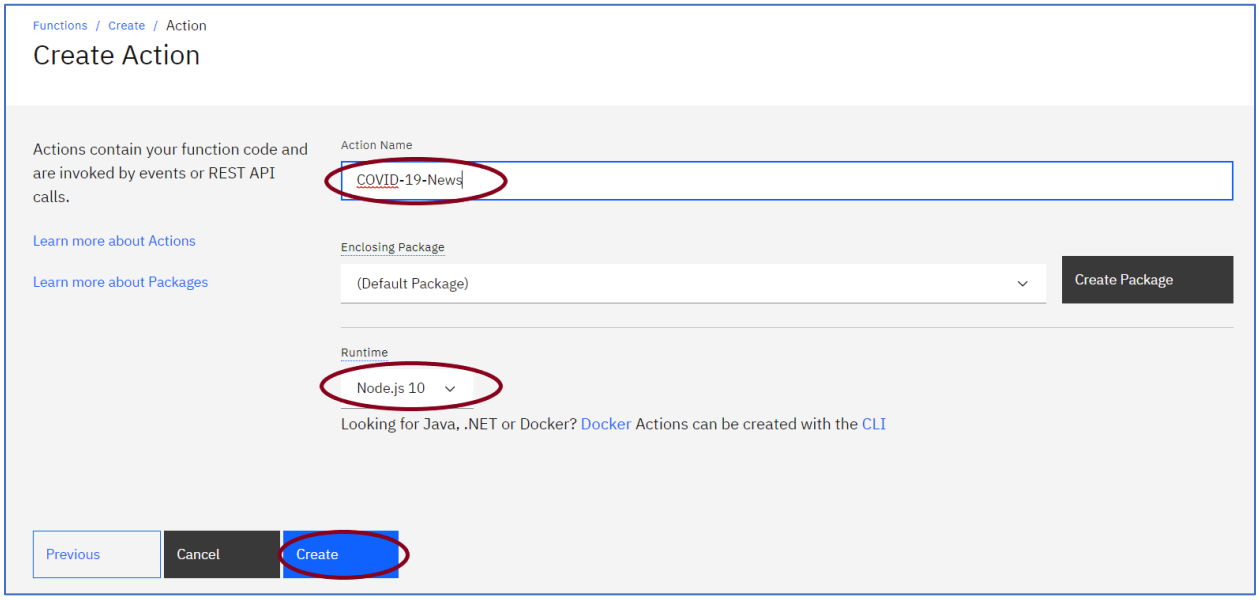

5. Click **Start Creating**.



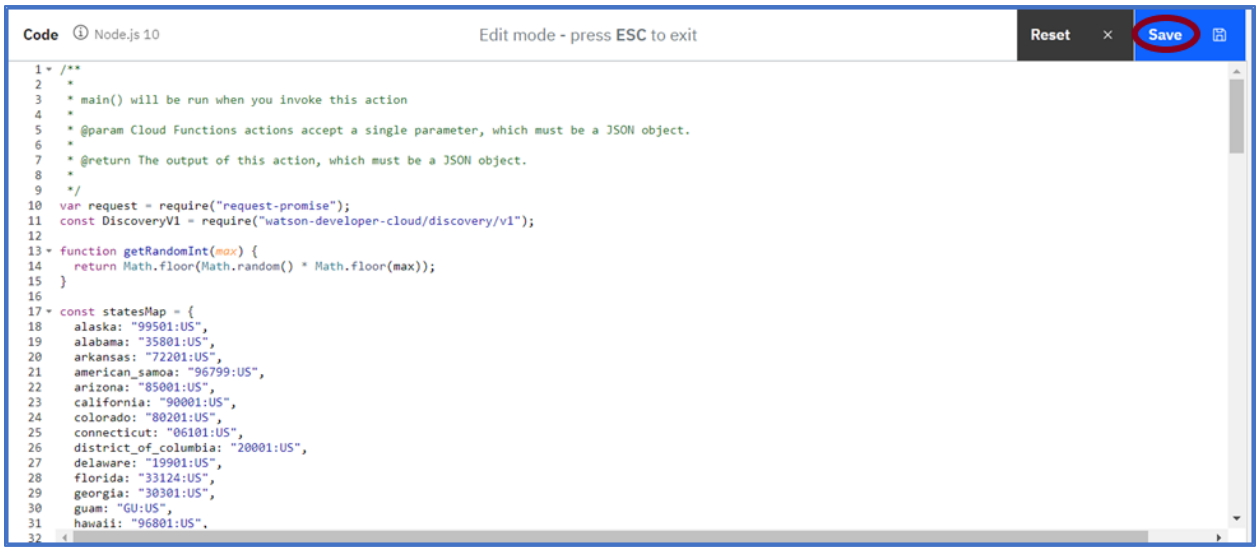

6. Click **Action**.



7. Enter **COVID-19-News** for the **Action Name**, select **Node.js 10** for the **Runtime**, and click **Create**.




8. Copy and paste the code from action/covid-webhook.js

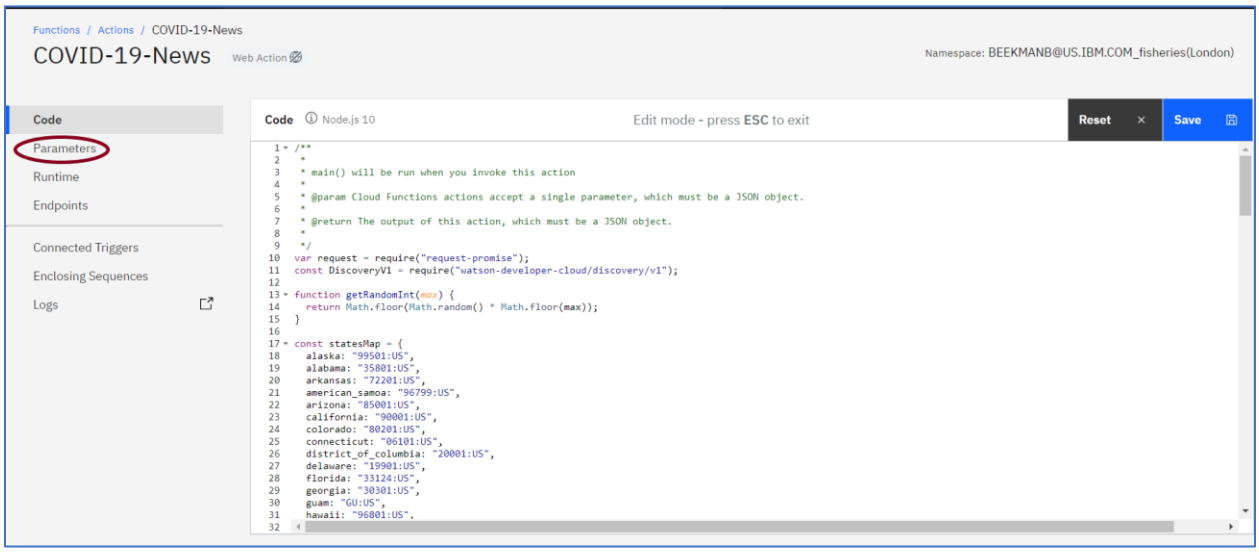



9. Our code has two main parts. We decide whether to call the COVID-19 API or Watson Discovery based on a parameter sent on the function call. If a query param of `type=api` is set, you call the COVID-19 api on the summary endpoint.

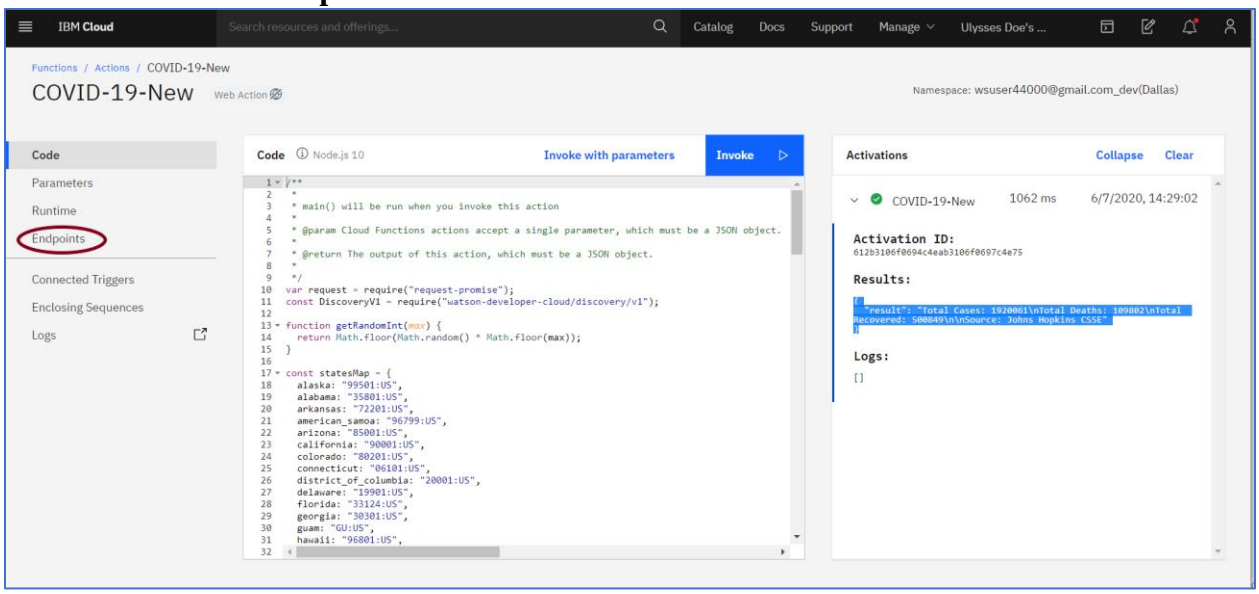
   For example, the response for `type=api` and `location=United States of America` is shown below.


```
{
 "result": "Total Cases: 1920061\nTotal Deaths: 109802\nTotal Recovered: 500849\n\nSource: Johns Hopkins CSSE"
}
```

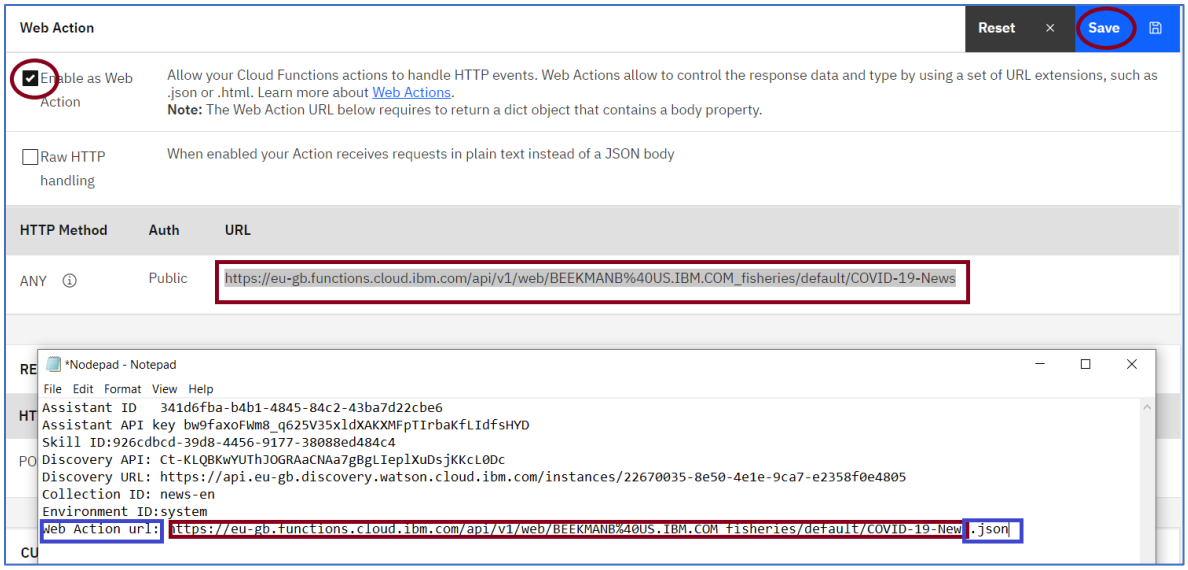
10. You could define static parameters as part of the Cloud Function by clicking on **Parameters**, but we will pass the parameters from the Watson Assistant.




11. Click on the **Endpoints** tab on the left.




12. Click on **Enable as a Web Action**. Copy and paste the http url to the Notepad file. Label it as Web Action url and add .json to the end of the url, as shown in blue below. Click on **Save**.



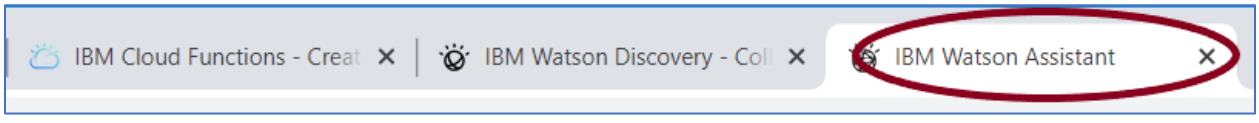

# Exercise 7: Integrate data sources via a Watson Assistant webhook.

A webhook is a mechanism that allows you to call out to an external program based on something happening in your Watson Assistant dialog skill. A webhook is triggered when the assistant processes a node that has a webhook enabled. The webhook collects data that you specify or that you collect from the user during the conversation and save in context variables. It sends the data as part of a HTTP POST request to the URL that you specify as part of your webhook definition. The URL that receives the webhook is the listener. It performs a predefined action using the information that you pass to it as specified in the webhook definition and can optionally return a response.

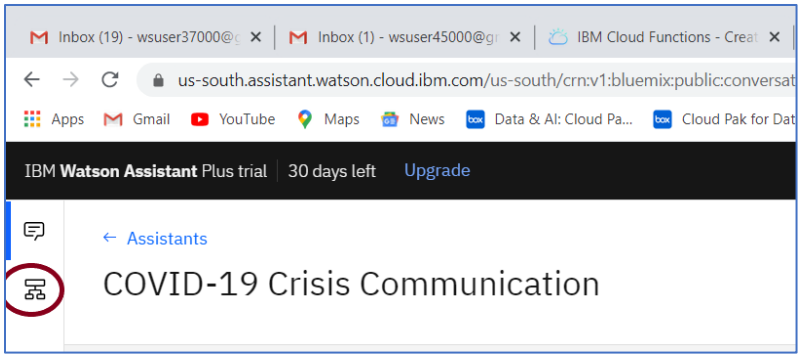
You can use a webhook to do the following types of things:

- Validate information that you collected from the user.
- Interact with an external web service to get information. For example, you might check on the expected arrival time for a flight from an air traffic service or get a forecast from a weather service.
- Send requests to an external application, such as a restaurant reservation site, to complete a simple transaction on the user's behalf.
- Trigger a SMS notification.
- Trigger a IBM Cloud™ Functions web action.

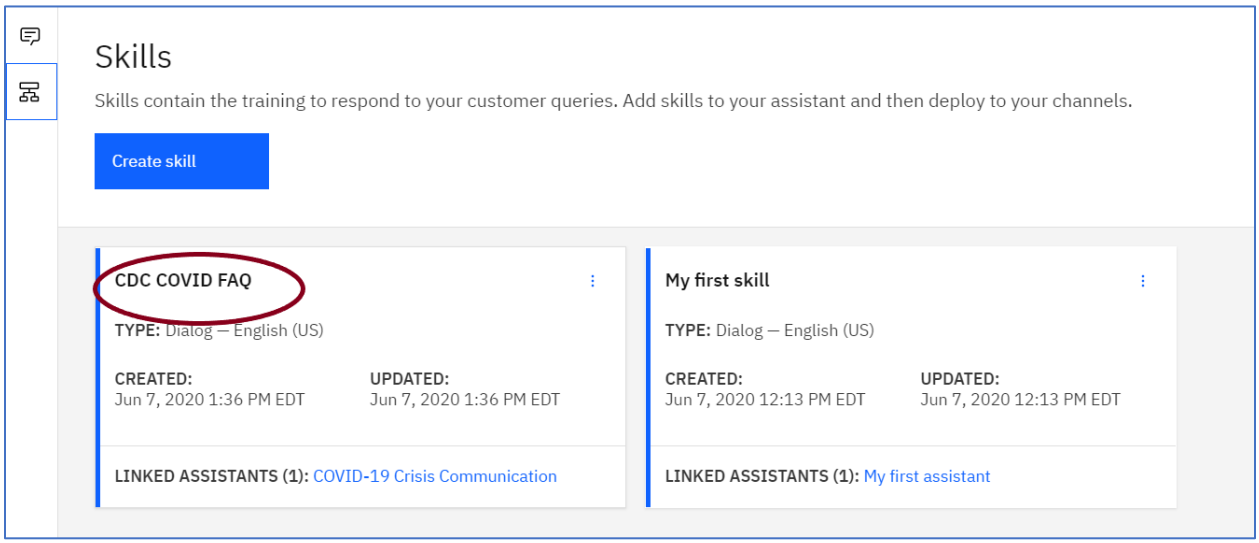
1. Click on **Watson Assistant**.



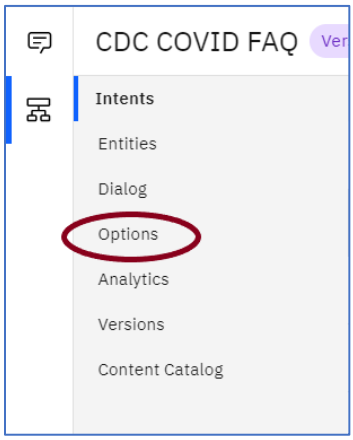

2. Click on the Skills icon 



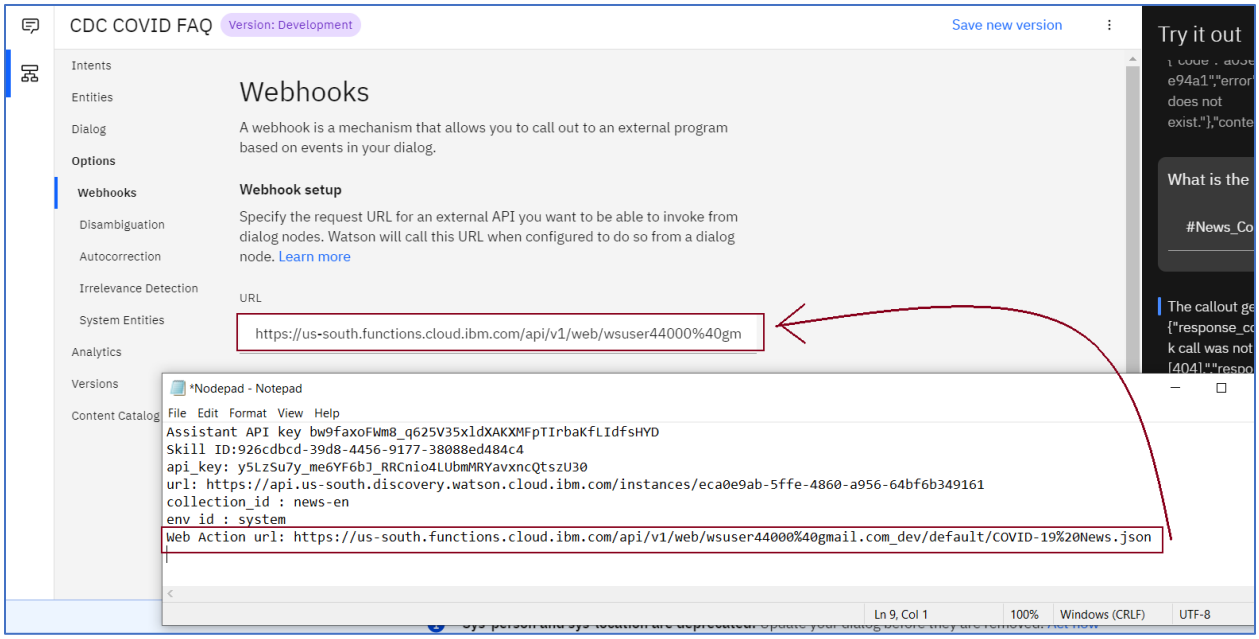

3. Click on the **CDC COVID FAQ** skill.



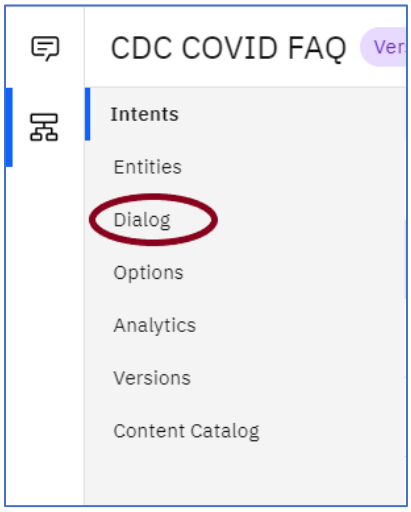

4. Click on **Options**.



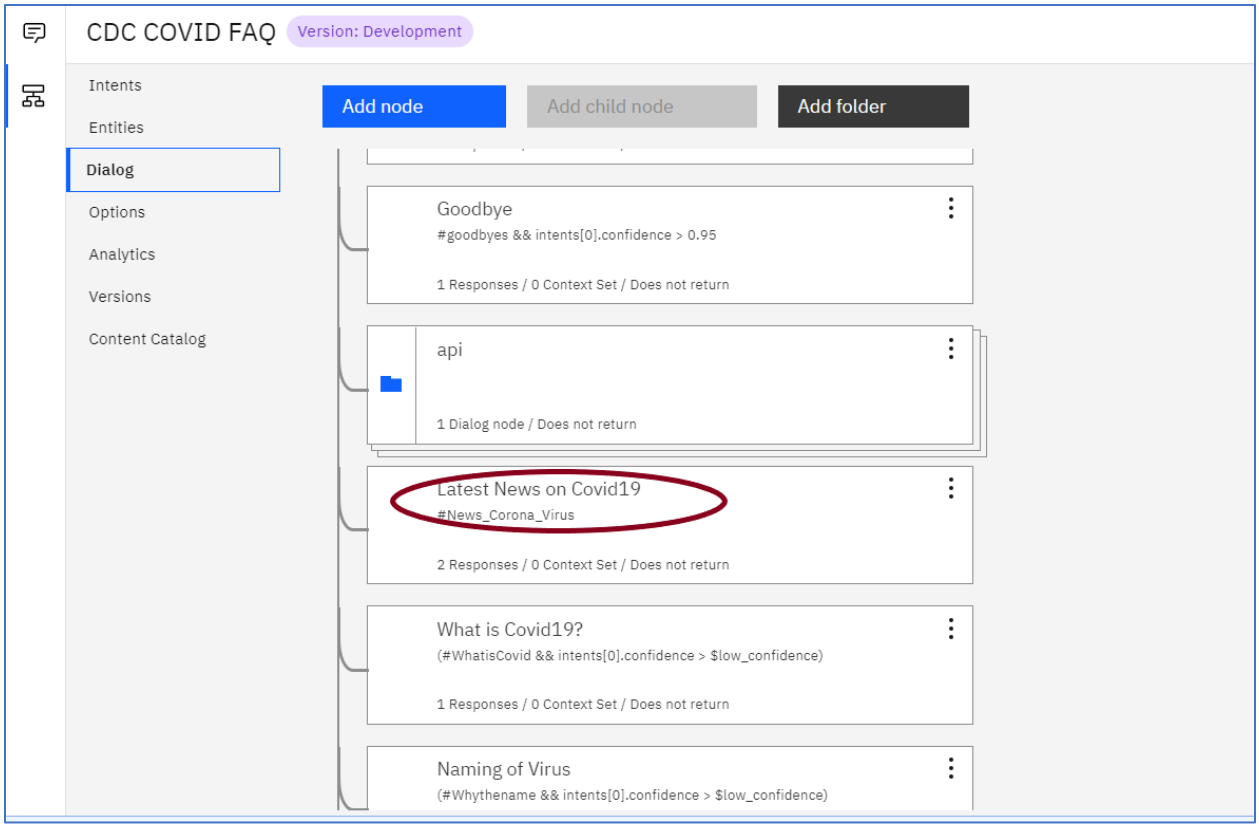

5. Cut and paste the Web Action url from the Notepad file into the **URL** field.



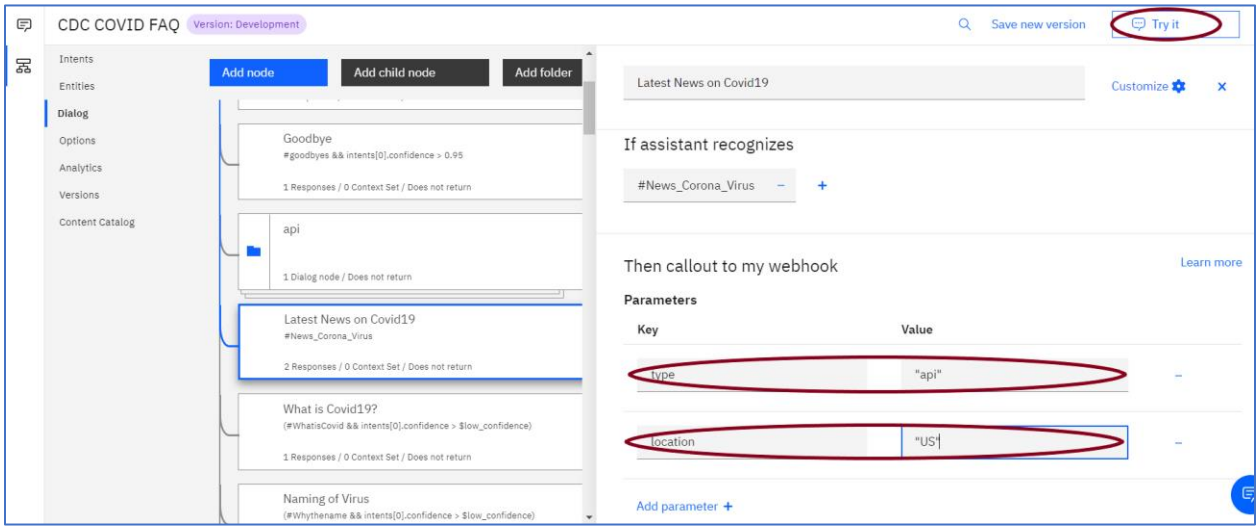

6. Click on **Dialog**.



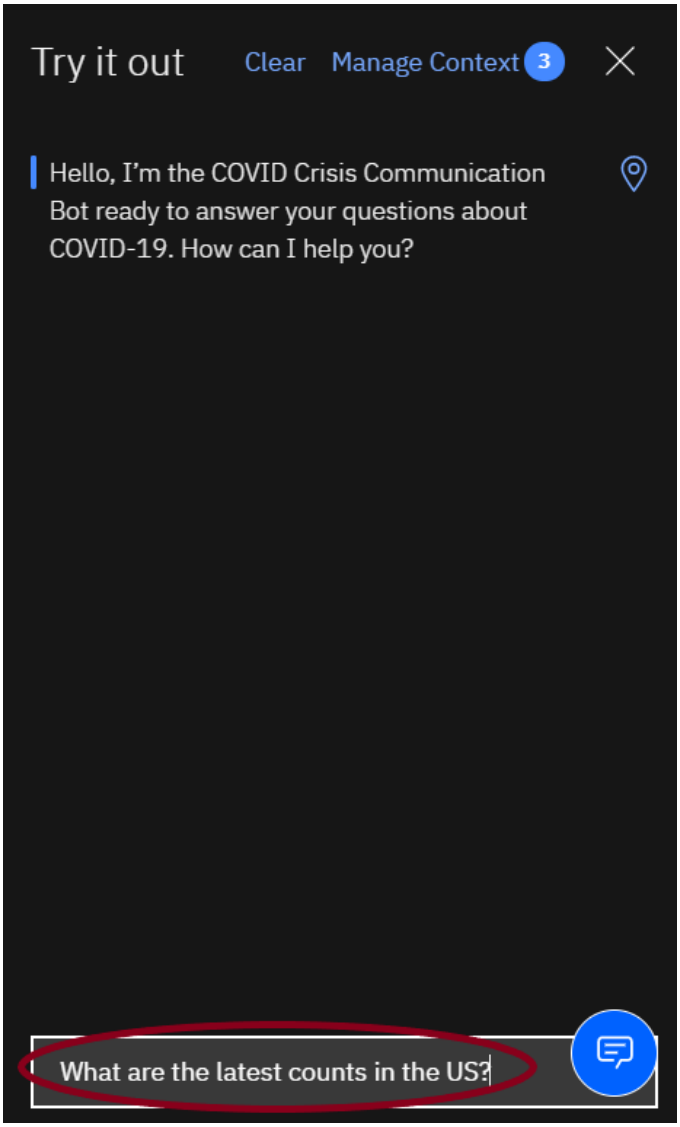

7. Click on **Latest News on Covid19**



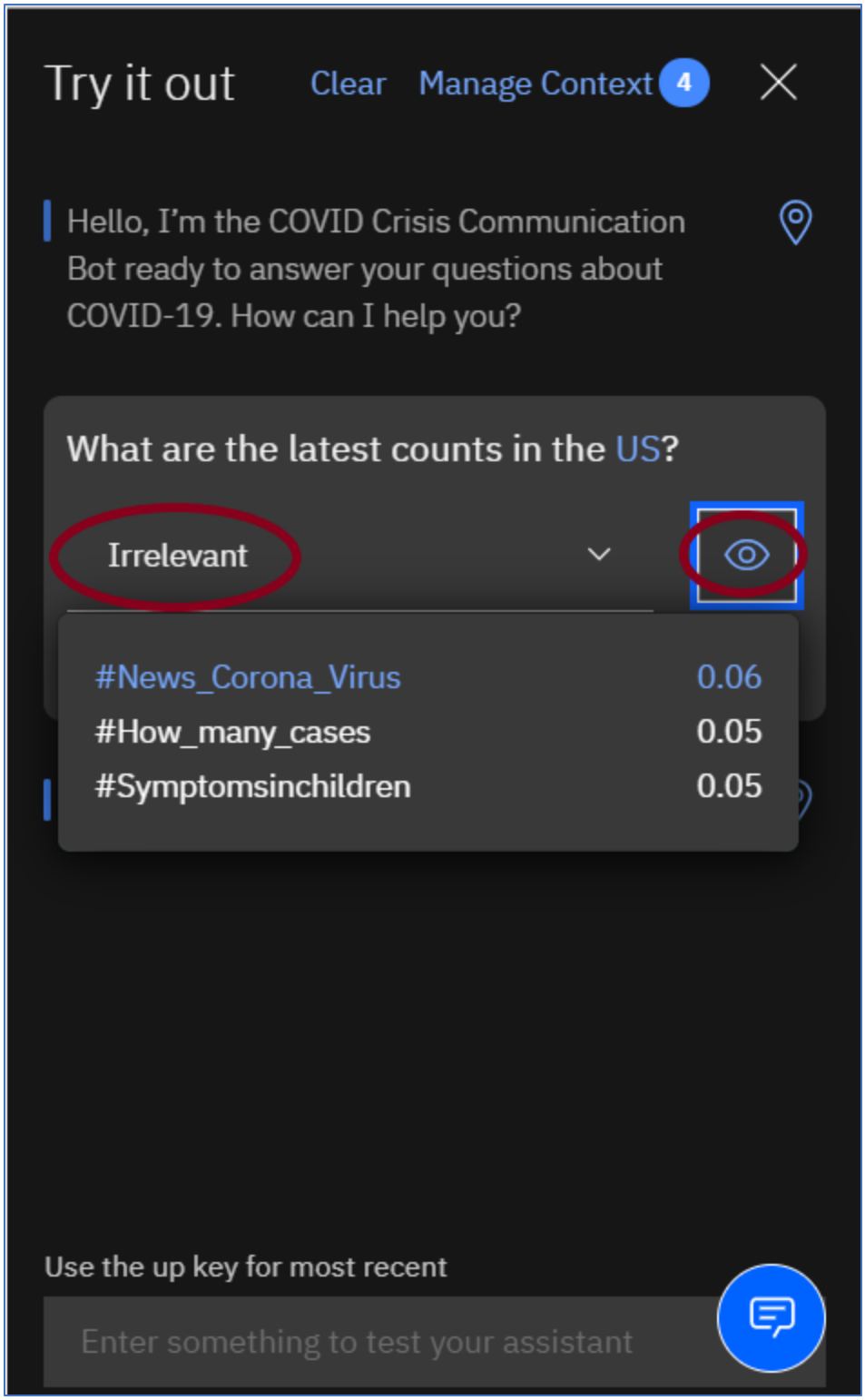

8. Add parameters **type** with value **“api”**, and **location** with value **“US”**, then click **Try It**.



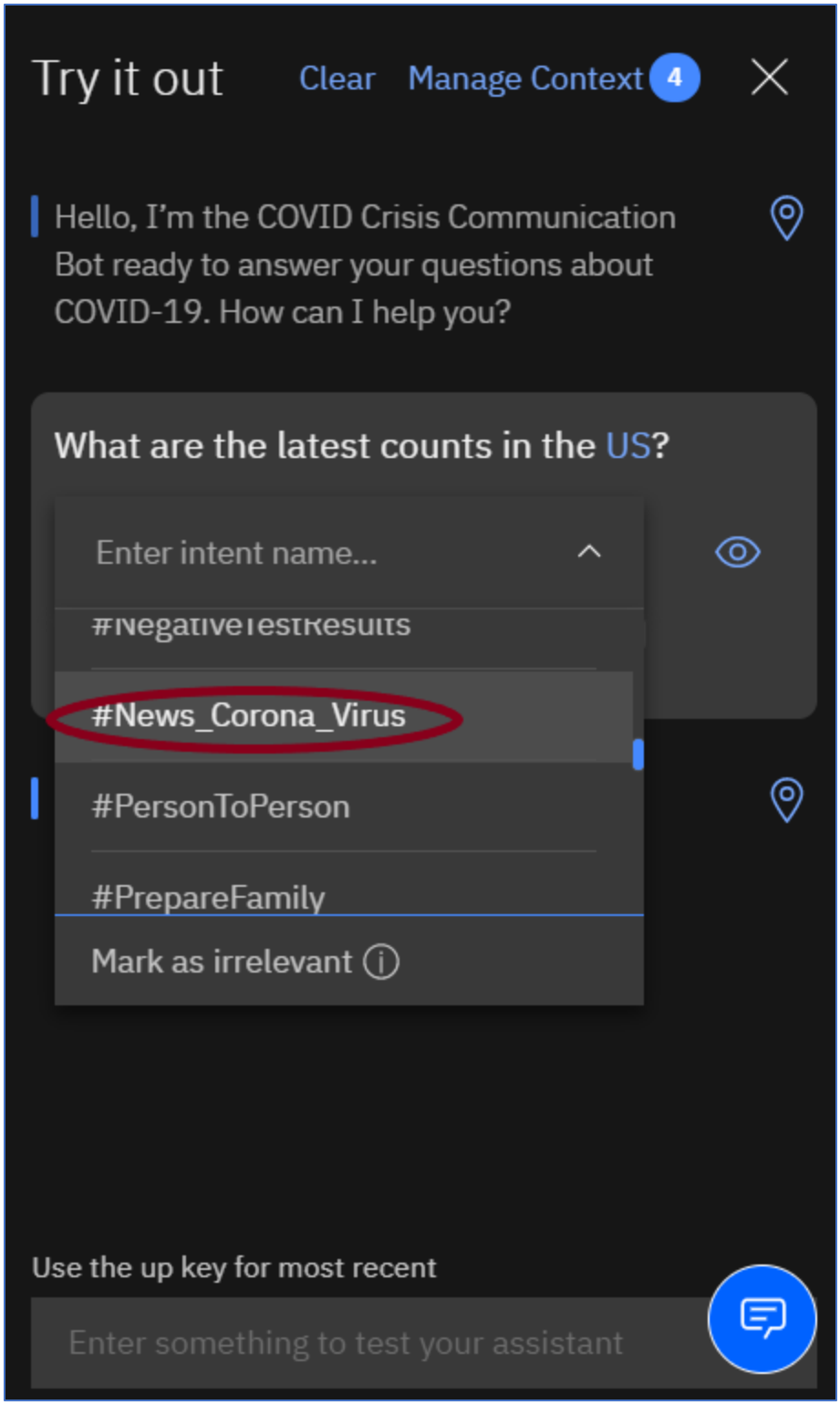

9. Enter “What are the latest counts in the US” and then press <Enter>



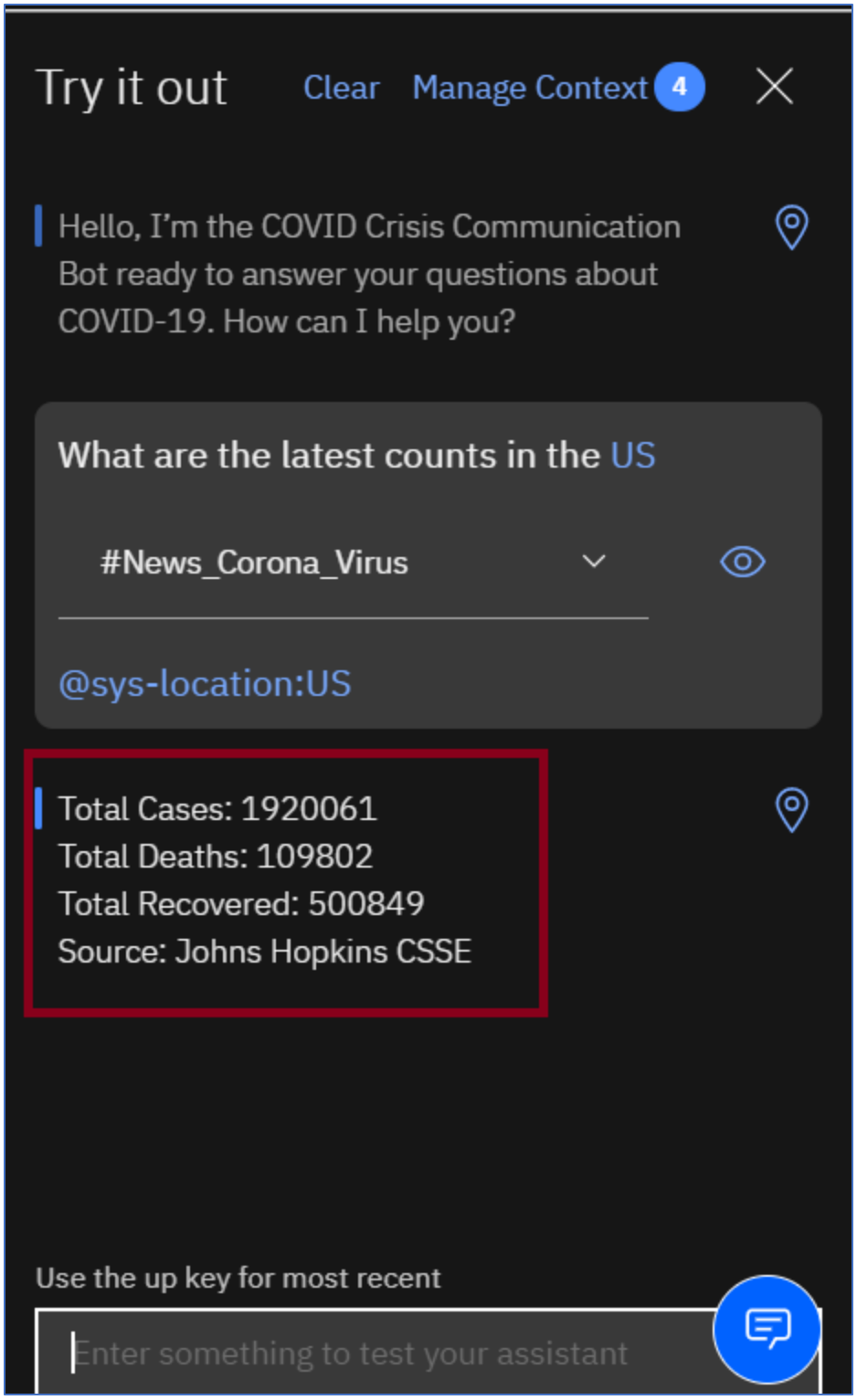

10. The system responds with Irrelevant, meaning that it couldn't find an Intent. By hovering over the eye icon, you can see that #News_Corona_Virus intent had the highest confidence, but was below the cutoff threshold for relevance.



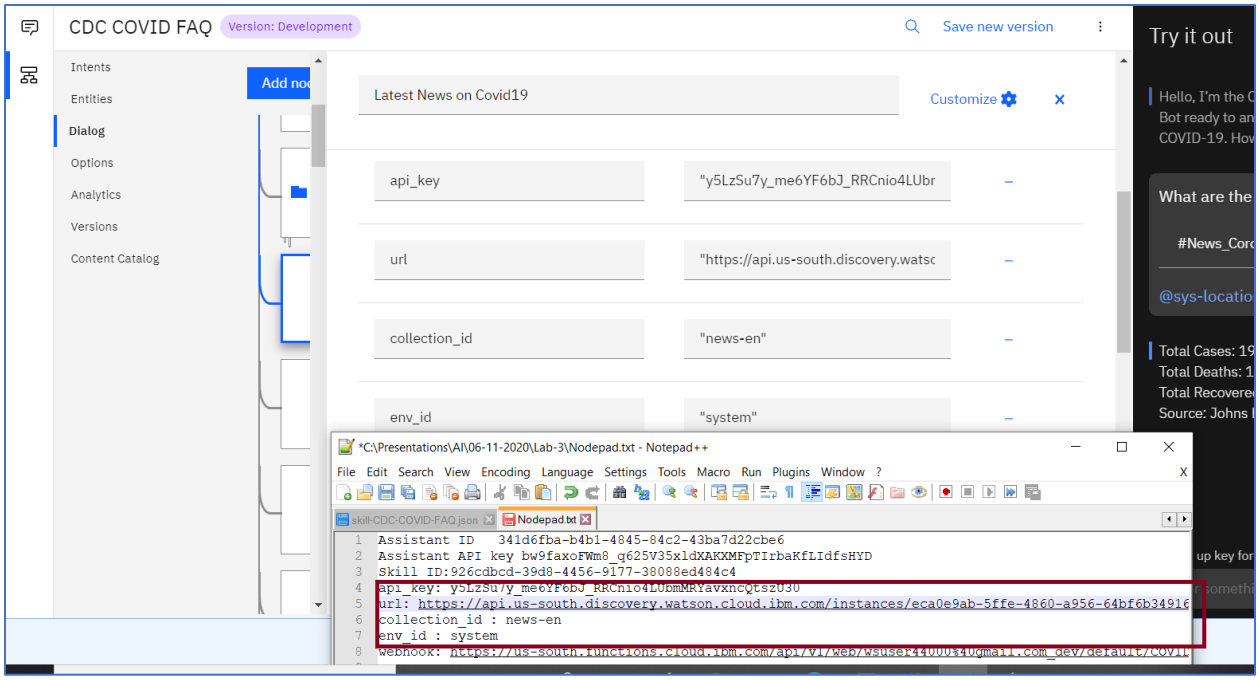

11. Click the down arrow icon next to Irrelevant, and select the #New_Corona_Virus intent to train the assistant.




12. Wait until **Watson is Training** message disappears. Retype in the question, "What are the latest counts in the US" and press the <Enter> key. The system responds with the latest counts.

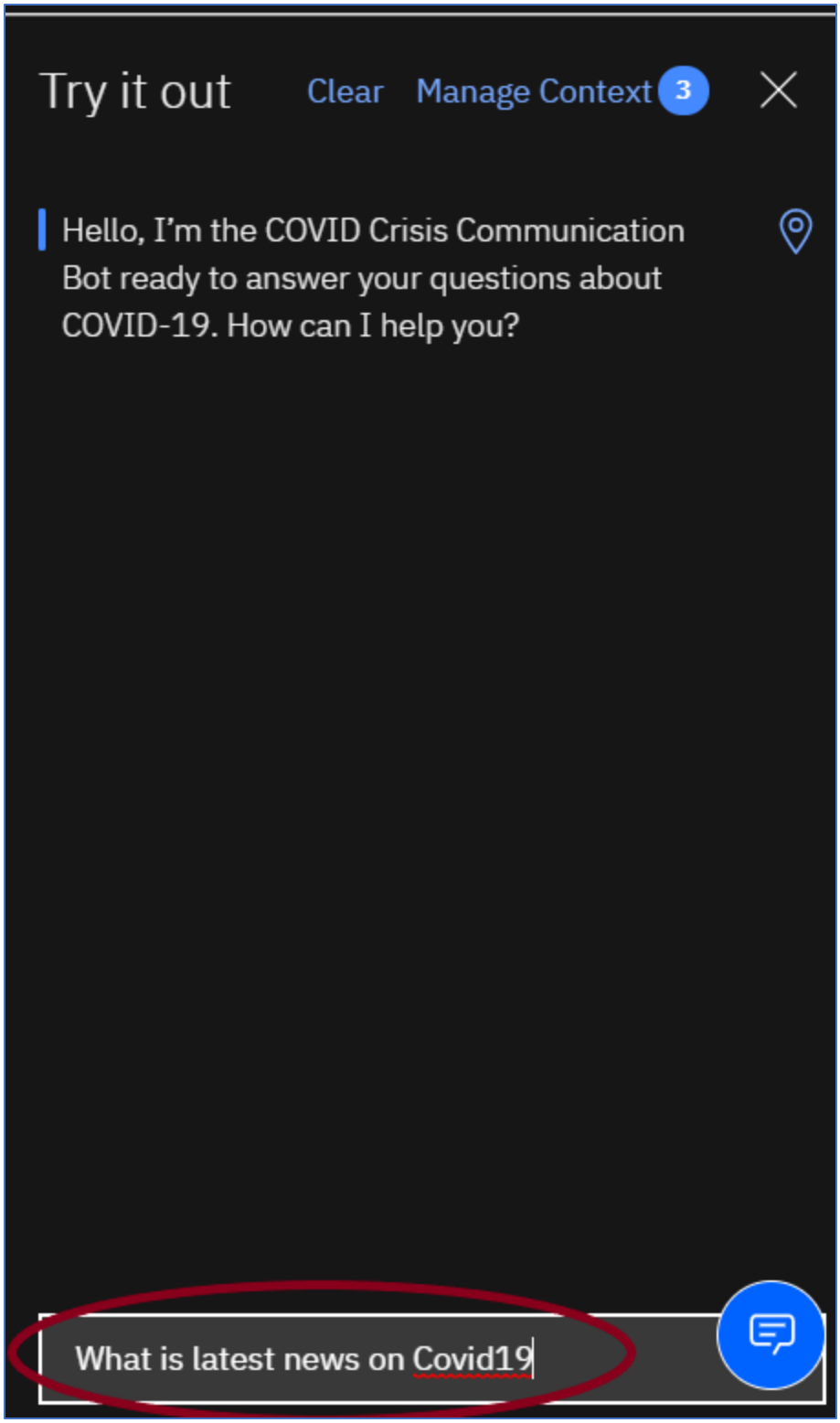



13. Add the Watson Discovery Parameters by copying and pasting from the Notepad file. Add api_key, url, collection_id, and env_id.

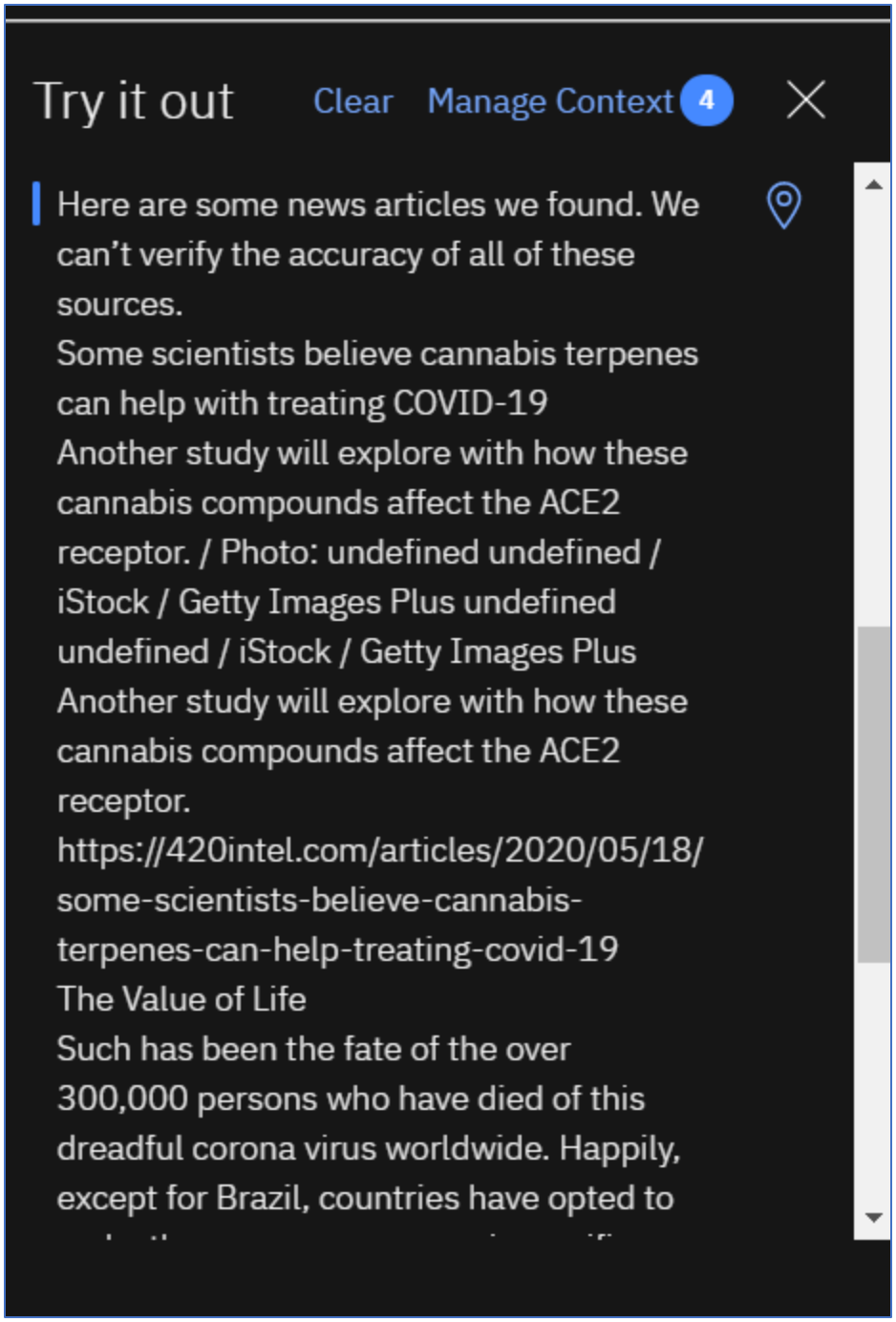



14. Change the value for type to be “napi”.


15. Type “what is the latest news on Covid19” and press the <Enter> key (you don’t need to type the quotes).




16. The system responds with a count of 3 articles from the Watson News collection.




This exercise showed how to integrate Watson Assistant with two data sources. For the purposes of this lab, we showed the connectivity within a single intent. In practice, we would have two intents, one that would get the latest data counts, and the second would collect the latest news.

## You have completed the Lab!