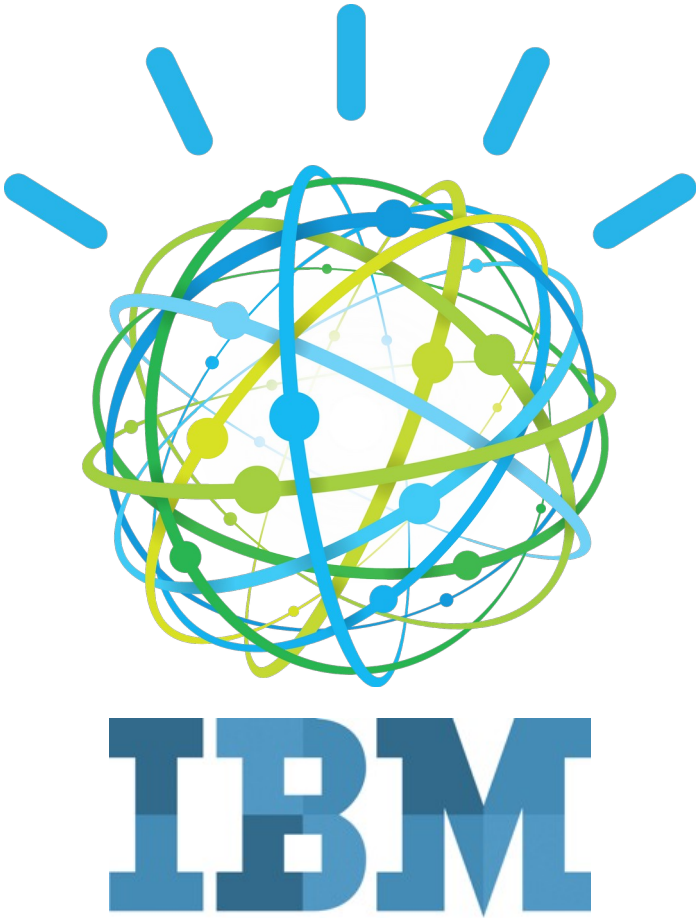
IBM Watson Discovery

Watson Assistant Integration with Webhook (Cloud Function/Node.js)

Cognitive Solutions Application Development

Updated: Apr 27, 2020

Klaus-Peter Schlotter
kps@de.ibm.com



Version 5

Watson Library V5

Table of Contents

Overview

The IBM Watson Developer Cloud (WDC) offers a variety of services for developing cognitive applications. Each Watson service provides a Representational State Transfer (REST) Application Programming Interface (API) for interacting with the service. Some services, such as the *Speech to Text* service, provide additional interfaces.

The Watson Discovery service adds a cognitive search and content analytics engine to applications to identify patterns, trends and actionable insights that drive better decision-making. Securely unify structured and unstructured data with pre-enriched content, and use a simplified query language to eliminate the need for manual filtering of results.

The app created in this lab can run locally or in the IBM Cloud.

- In a first step, a Watson Discovery service instance is created on the IBM Cloud.
- Next, you will adjust the IBM Cloud demo to use the German document collections
- Then the chatbot from lab 3 will be enhanced with Discovery capabilities.

Objectives

- Learn how to use the Watson Discovery service on IBM Cloud
- Learn how to integrate with Watson Assistant using the Search Skill feature.

Prerequisites

Before you start the exercises in this guide, you will need to complete the following prerequisite tasks (see the setup guide).

- Create a IBM Cloud account

Section 1: Create a Discovery service instance

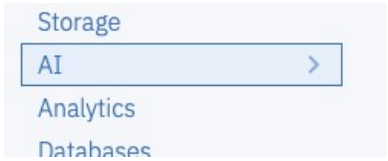
Step 1 In a web browser, navigate to the following URL

https://cloud.ibm.com

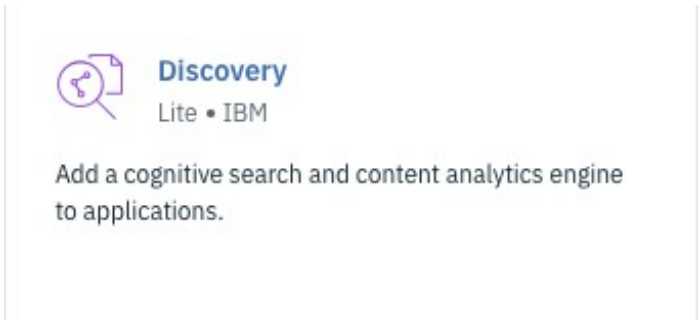
Step 2 Log in with your IBM Cloud credentials. This should be your IBMid.

Step 3 You should start on your dashboard which shows a list of your applications and services. Scroll down to the All Services section and click **Create Resource**.

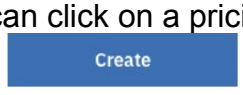
Step 4 On the left, under Services, **click** on *AI* to filter the list and only show cognitive services.



Step 5 **Click** on the Watson Discovery service.



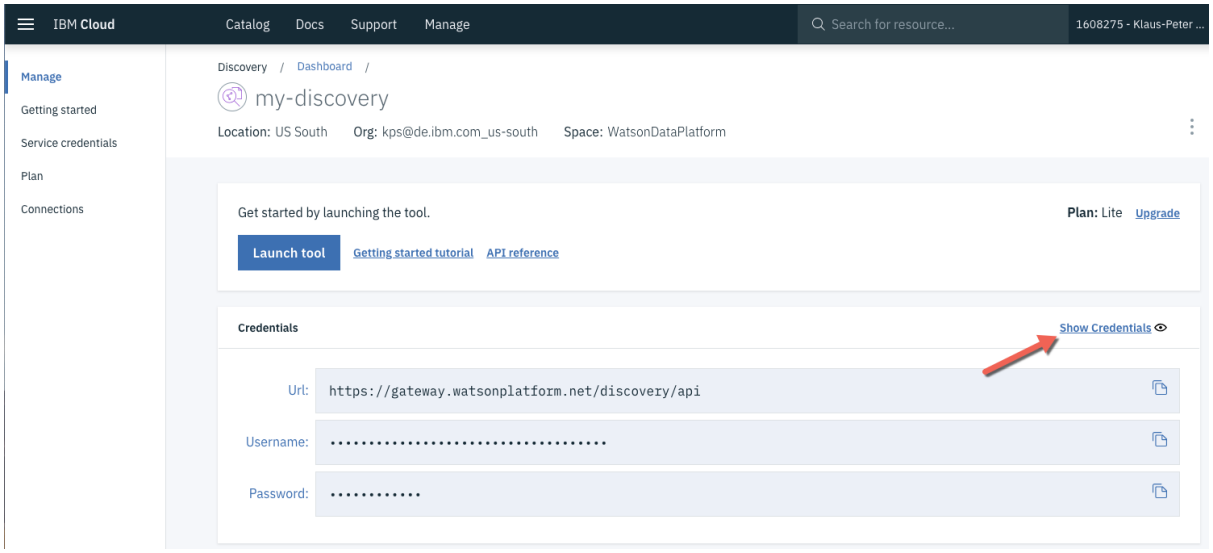
Step 6 Review the details for this service. At the top, there will be a description of the service. At the bottom, you can review the pricing plans. The Lite plan for this service provides no cost monthly allowances for workspaces, intents, and API calls. Enjoy your demo!

Step 7 At the top, you can enter information for your new service. In the middle, you can click on a pricing plan to select it. Fill out the fields as follows, then **click** 

at the bottom.

Field	Value
Service name	my-discovery
Selected Plan	Lite

Step 8 IBM Cloud has created a new service instance.



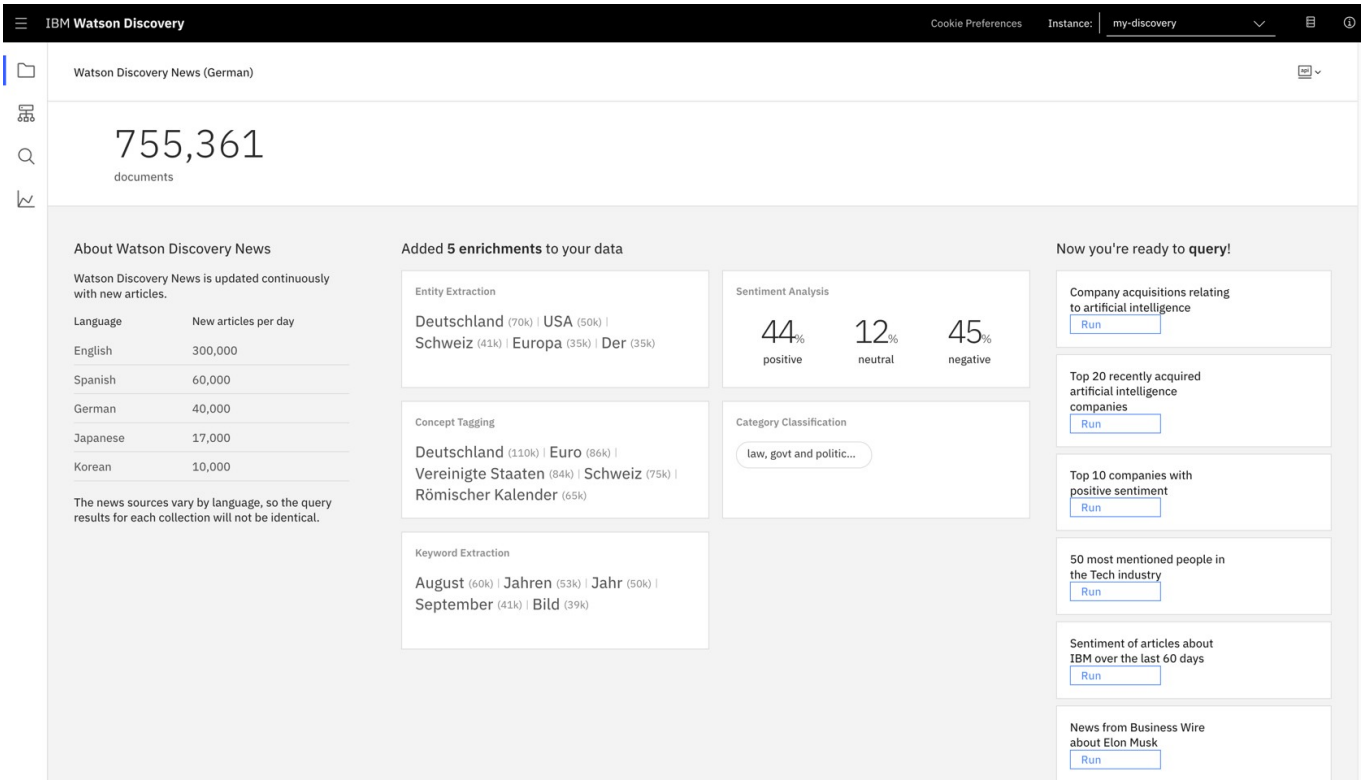
Step 9 In the *Credentials* section **click** Show Credentials . You should see the user-name and password for your service. Later in this exercise, you will enter these values into a JSON configuration file for your Node.js application. Feel free to copy them to your clipboard, to a text file, or just return to this section of the IBM Cloud web interface when the credentials are needed.

Step 10 **Click** the Launch tool button to open the Discovery service configuration tool.

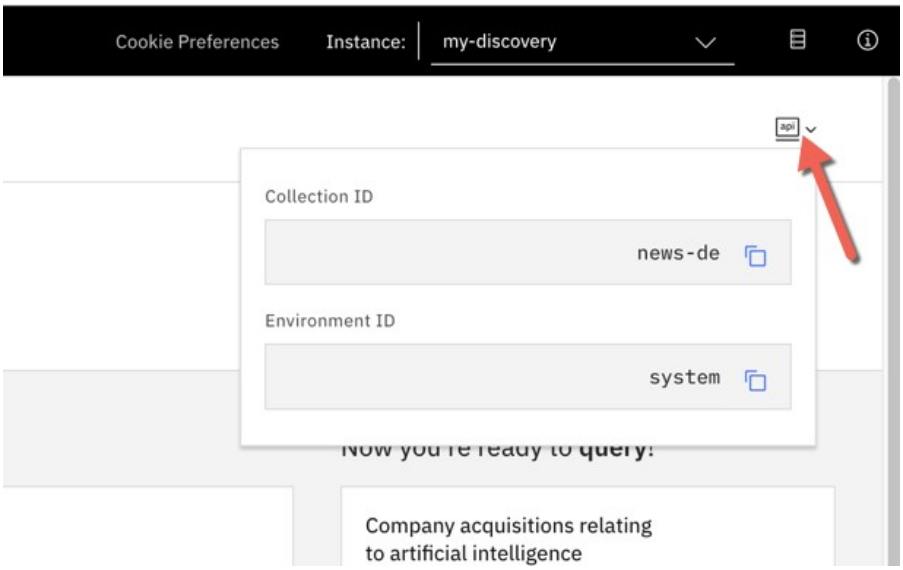
Step 11 **Select** your language and **click** on the *Watson Discovery News* collection, which is available in every service instance, to open it.

This collection is provided by IBM and gets updated daily, in German with approximately 40000 news articles per day containing a 60 day period of news (~ 3 Million entries).

On the overview page of the document collection you can see infos such as *Document count*, *Collection Info*, and *Insights from enriched data*. This enriched data is added by Watson when the documents get ingested.

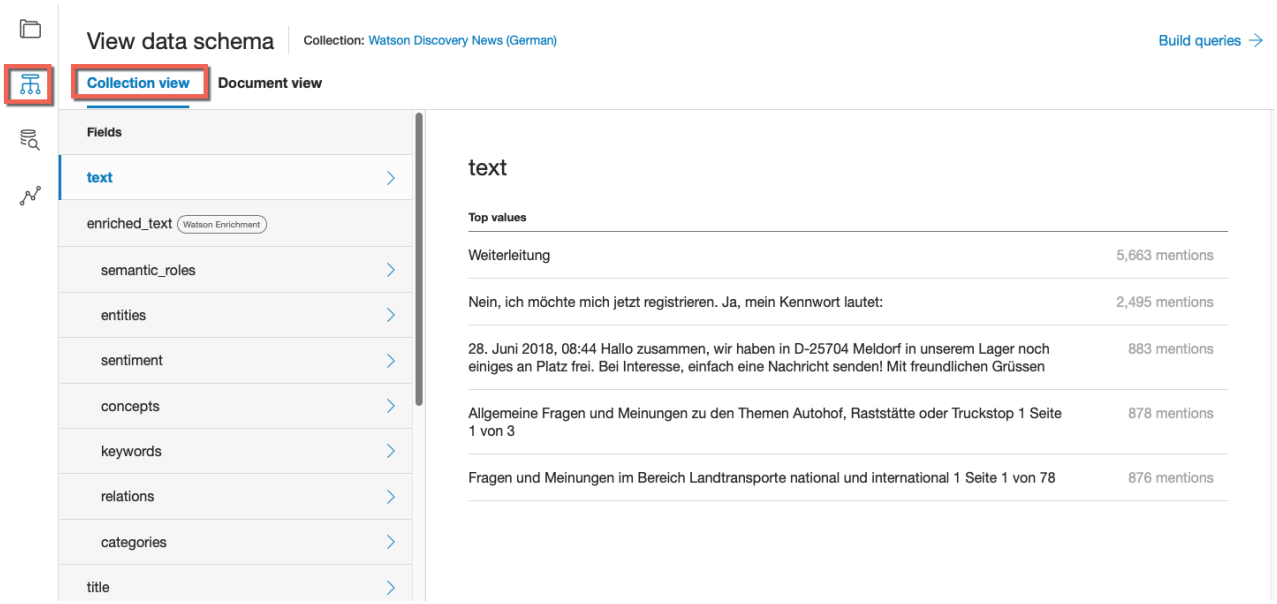


The *Use this collection in API* information is needed to access this information from your application, *Collection Id* and *Environment Id*.

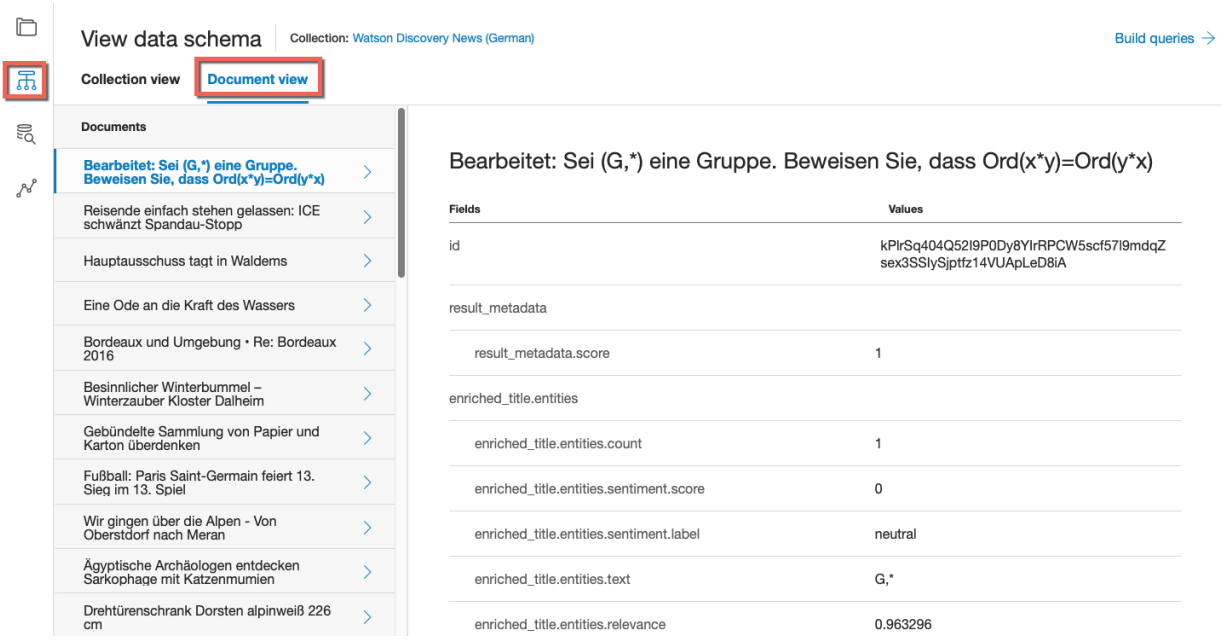


Step 12 On the *View data schema* page you can look at your collection in a

a) *Collections View* which shows you a summary of the Watson Enrichments on the collection scope

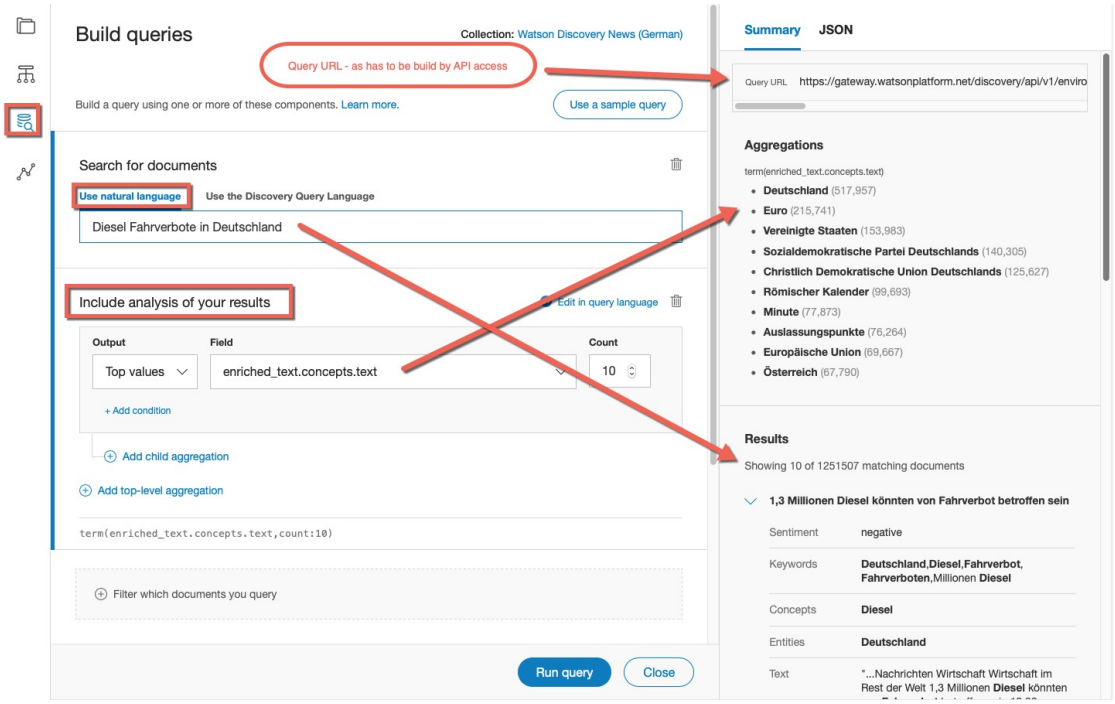


b) *Document View* page which shows you Watson Enrichments on a document level.

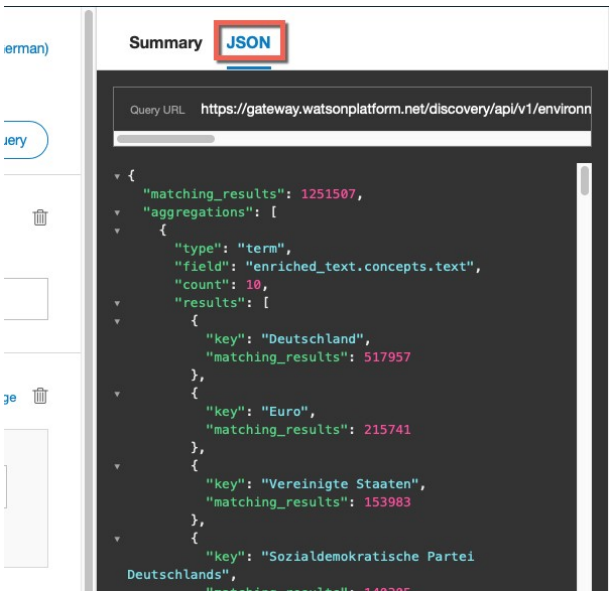


See the documentation of the service for more information.

Step 13 On the *Build queries* page you can search your collection f.e. *Use natural language* and include an analysis of the Watson Enrichment.



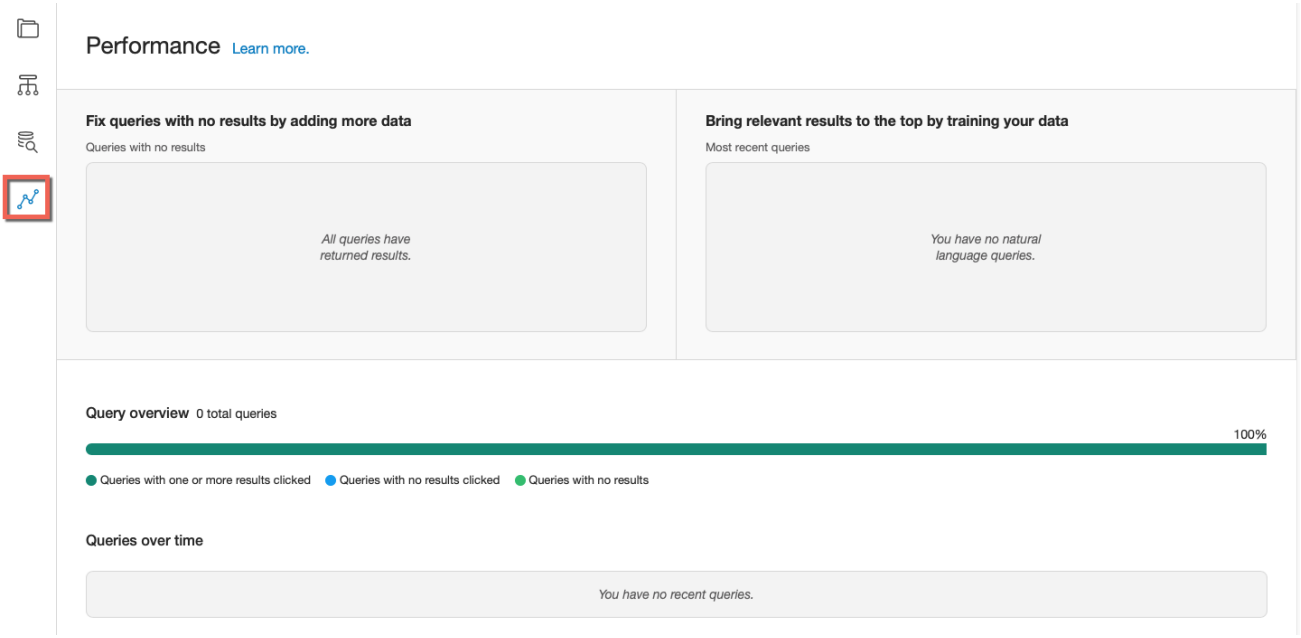
Step 14 Click the JSON tab at the top right then you see the result that you would get when you call this from the API.



Here is a nice video of the tutorial listed at the end of this document on how to work with the Discovery enriched-data to build powerful queries using this IBM News collection.
https://www.youtube.com/watch?v=N-HaIpPGde0&feature=youtu.be

More information can be found in the documentation.

Step 15 On the *Performance* page you can see how your collection performs over time. Right now, there is no data available.



Section 2 Integrate Discovery into a Chatbot (Webhook)

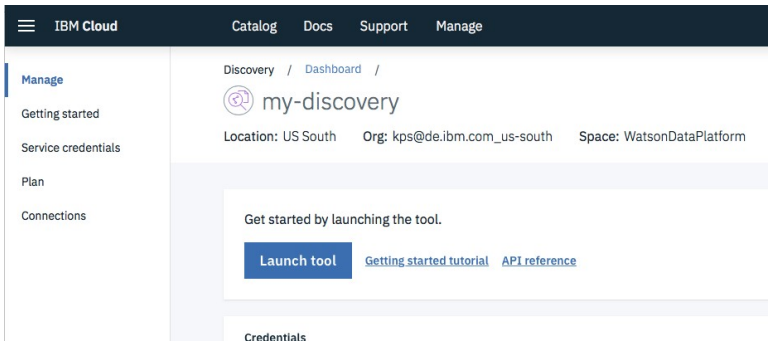
In this section we will integrate the Discovery Service into the Chatbot created in Lab 3a.

Step 16 As a starting point for this section you should follow Lab 3a Section 1. This section will list shortcuts on how to get started quickly such as importing the Conversation workspace or cloning the app source code.

Create a Watson Discovery Document Collection

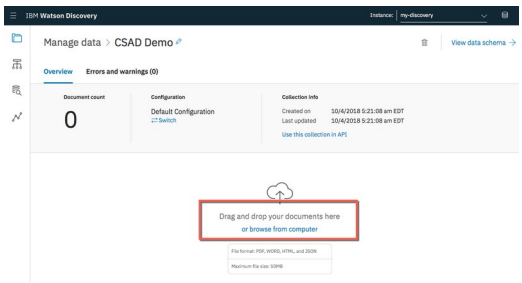
Step 17 Download the zip file manualdocs.zip from github (**click** the Download button) and extract it into a folder. It's a web manual for a Ford car.

Step 18 In your cloud console dashboard **click** on the "my-discovery" service created above and **click** the Launch tool button.

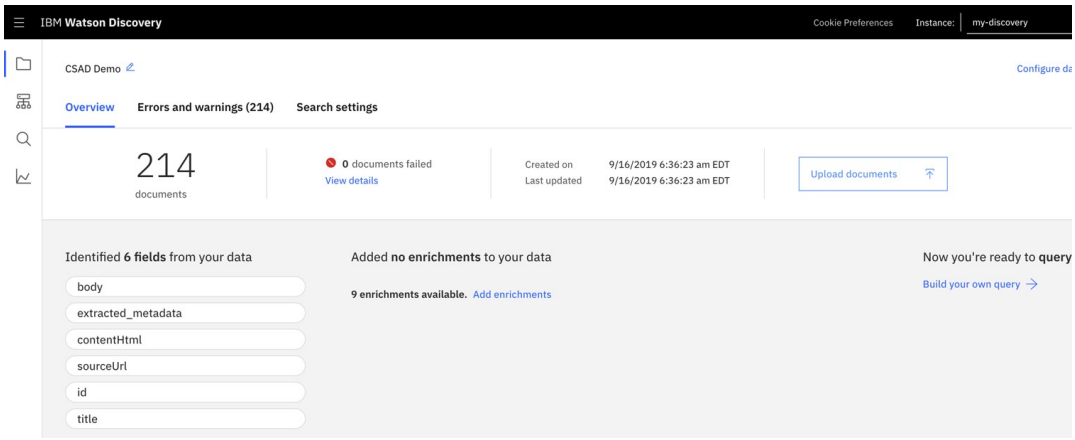


Step 19 On the *Manage data* page **click** Upload your own data .
As collection Name **enter** *CSAD Demo* and click Create.

Step 20 The new collection opens and you can either **browse** or **drag 'n' drop** all the documents extracted above (234 json files).

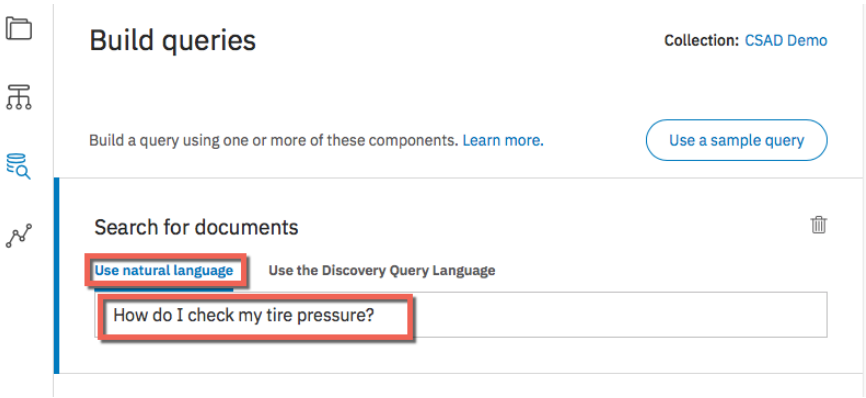


Step 21 The Discovery Service now indexes the documents. This may take a few minutes.



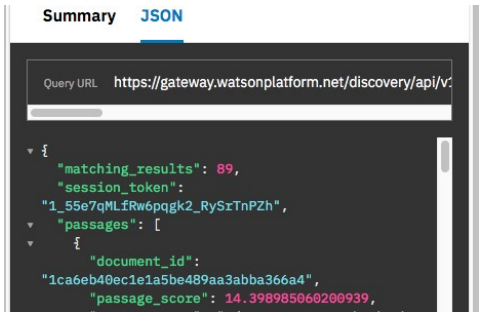
Step 22 On the left, click the button to open the *Build queries* page.

Step 23 **Enter** the following query: `How do I check my tire pressure?`



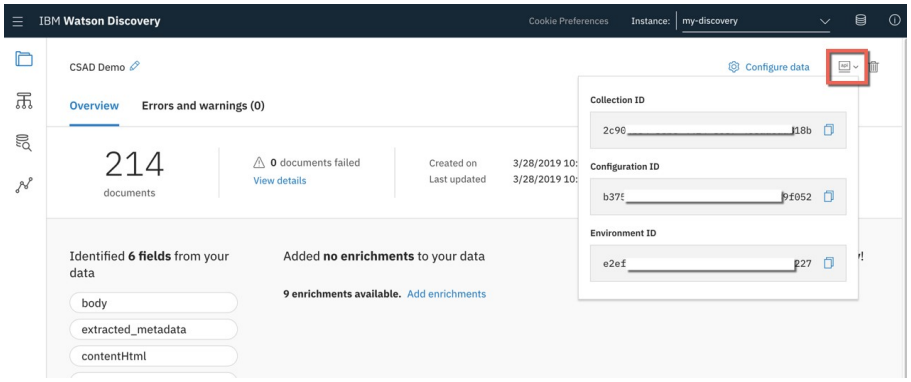
and press the Run query button.

On the left you can then see the results. Especially switch between the Summary and the JSON view of the results.



There is a lot more to explore here, but this is out of scope for this short tutorial. We just want to use the question submitted above later in our extended chatbot.

Step 24 On the top right of the *Overview* page **click**  to show the ids you have to copy for later use by the application in addition to the service credentials.



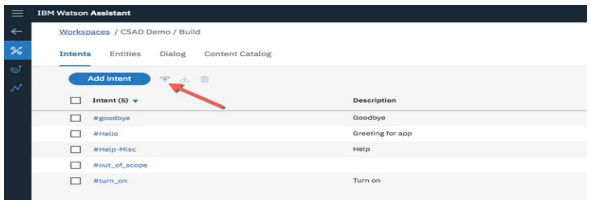
Update the Assistant Skill to integrate the Discovery Collection

Step 25 Download the out_of_scope_intents.csv from the Github repository to your project folder. Use the following in the terminal:

```
wget https://github.com/iic-dach/csadConversation/blob/master/resources/
Out_of_scope_intents.csv
```

Step 26 Go back to your Watson Assistant console that should still be open in the browser (Step 11)

Create a new intent *out_of_scope* and return to the list of intents. C**lick** the  button to import the user examples from the file downloaded above.



131 examples will be imported.

Step 27 On the *Dialog* tab **click** the menu ⋮ button on the *Turn on* node and **click** *Add node below*. **Name it** *Out of Scope*. In the field *If bot recognizes* **type** *#out_of_scope*.

> **Note:** Due to modularity we create the node in this location. Because this is also some kind of help question, it might be also appropriate, to insert this in the child tree of the *help* node.

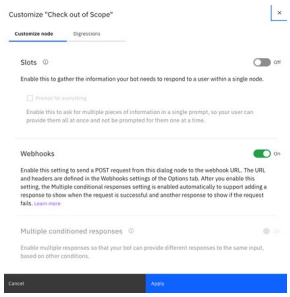
Step 28 Now **click** the menu ⋮ button on the *#out_of_scope* node. And **click** *Add child node*.

Step 29 In this child node, **name** it *Check out of Scope,* in **If bot recognizes**, just **type** *true*.

Step 30 Now **click** the menu ⋮ button on the *#out_of_scope* node. And **click** *Jump to (Recognizes Condition)*.

Step 31 **Click** the *Check out of Scope* node.

Step 32 **Click** Customize and **enable** *Webhooks*. Then click Apply



The webhook was already customized in Lab3a.

Step 33 Now you can add parameters for the webhook callout.

Add the following:

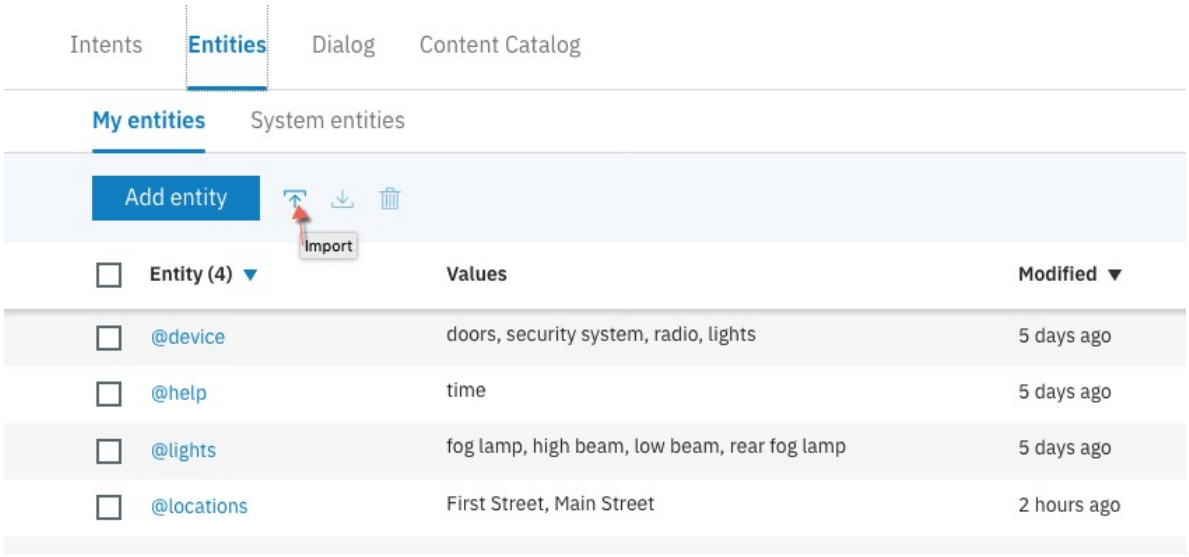
Key	Value
action	discovery
query	<? input.text ?>

Step 34 Add the following in Assistant responds. Complex responses can be easily built on the customize screen ⚙.

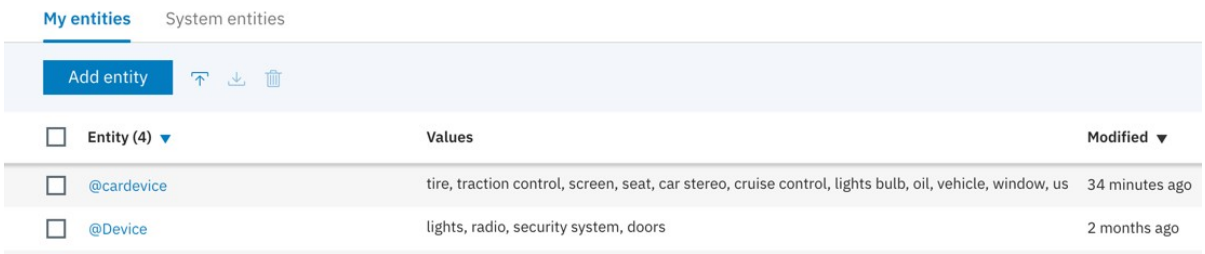
If assistant recognizes	Respond with
$webhook_result_2 && @cardevice	<? $webhook_result_2.message ?>
true	Sorry I haven't learned answers to questions like this.

Step 35 In your Skill's Entities add an Entity by importing the cardevice_entity.csv CSV file.

```
wget https://github.com/iic-dach/csadConversation/blob/master/resources/
cardevice_entity.csv
```



This will add an Entity @cardevice



Step 36 Now change the *Check out of Scope* node (Step 48) to the following

If bot recognizes	Respond with
@cardevice	That question is out of scope for this application, take a look at the Conversation Enhanced application to handle questions like these.
true	Sorry I haven't learned answers to questions like this.

"true" always executes when the above condition is not met.

Step 37 The backend is now adjusted for integrating the Discovery service into our chatbot application.

Add the Discovery Service call to the Cloud Function

In Lab 3a we created a Webhook/Cloud Function to provide the system time for the chatbot. To following code shows the Cloud Function with the added Watson Discovery Service integration. **Copy** and **Paste** from here.

Step 38 Update/replace the Cloud Function with the following code

```
const axios = require("axios");
const discoveryUrl =
"https://gateway.watsonplatform.net/discovery/api/v1/environments/{your environmen-
tId}/collectionsId/{your collection}/query?version=2019-04-30";
const auth = "Basic " + new Buffer.from("apikey" + ":" +
"{your_apikey").toString("base64");

async function main(params) {
  if (params.action === "gettime") {
    const deTime = new Date().toLocaleString("de-DE", {timeZone: "Europe/Berlin"});
    return {
      statusCode: 200,
      headers: {
        Content_Type: "application/json",
      },
      body: { message: deTime },
    };
  } else if (params.action === "discovery") {
    try {
      let response = await axios({
        method: "post",
        url: discoveryUrl,
        data: {
          natural_language_query: params.query,
          passages: true,
          "passages.count": 1,
        },
        headers: {
          Authorization: auth,
          "Content-Type": "application/json"
        },
      });
      return {
        statusCode: response.status,
        headers: {
          "Content-Type": "application/json",
          Accept: "application/json",
        },
        body: { message: response.data.passages[0].passage_text }
      };
    } catch (err) {
      return { body: {message: err.message }};
    }
  } else {
    return {
      statusCode: 404,
      headers: { "Content-Type": "application/json" },
      body: { messsage: `action ${params.action} not defined` }
    };
  }
}
```

Some constants where added to get the **axios** Promised base HTTP client, to define the **Wastson Discovery collection url** and the to create a **Basic Auth token** from username (the string “apikey”) and as password the services apikey.

Because getting the data from the Discovery Service is an asynchronous function, **function main(...)** has to be preceded with the **async** keyword. This corresponds with the **await** keyword in the axios command.

Step 39 Test your Assistant with **Try it out**

Step 40 Enter some statements into the Assistant input line:

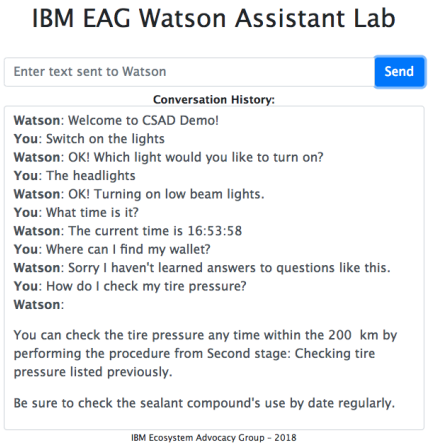
Switch on the lights

The headlights

What time is it?

Where can I find my wallet?

How do I check my tire pressure?



The question for the tire pressure should return the same as in **Step 23**, the first passage.

Section 4 Integrate Discovery into a Chatbot (Search Skill)

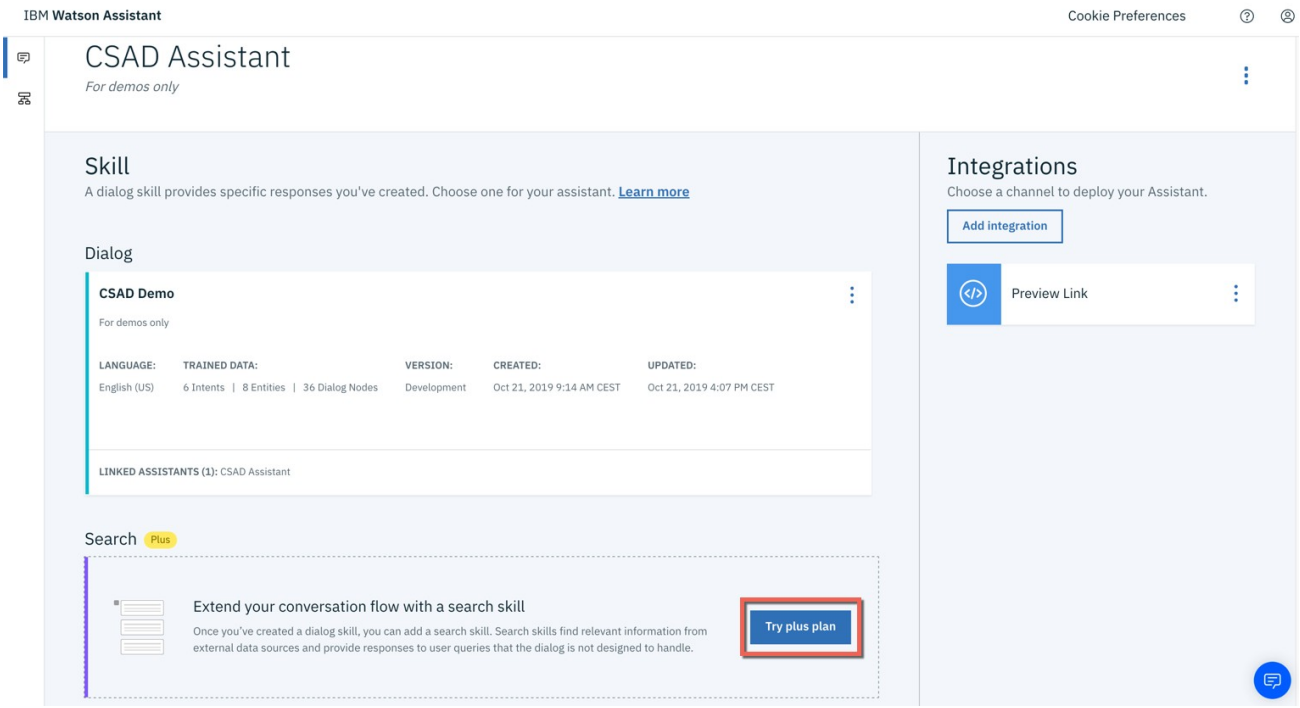
In the previous section we have seen in how to integrate the Watson Discovery service into a chatbot using the API.

In the Watson Assistant service there is now a *Search Skill* option available that allows to directly integrate a Discovery document collection into a chatbot.

To use this search skill we have to upgrade our Discovery service instance to the **Plus Trial** plan.

Create the Search Skill

Step 41 In the browser **open** your *Watson Assistant* console and **click** Try plus plan



Step 42 Then **click** Start free trial and **click** Agree to start the 30 day trial period.

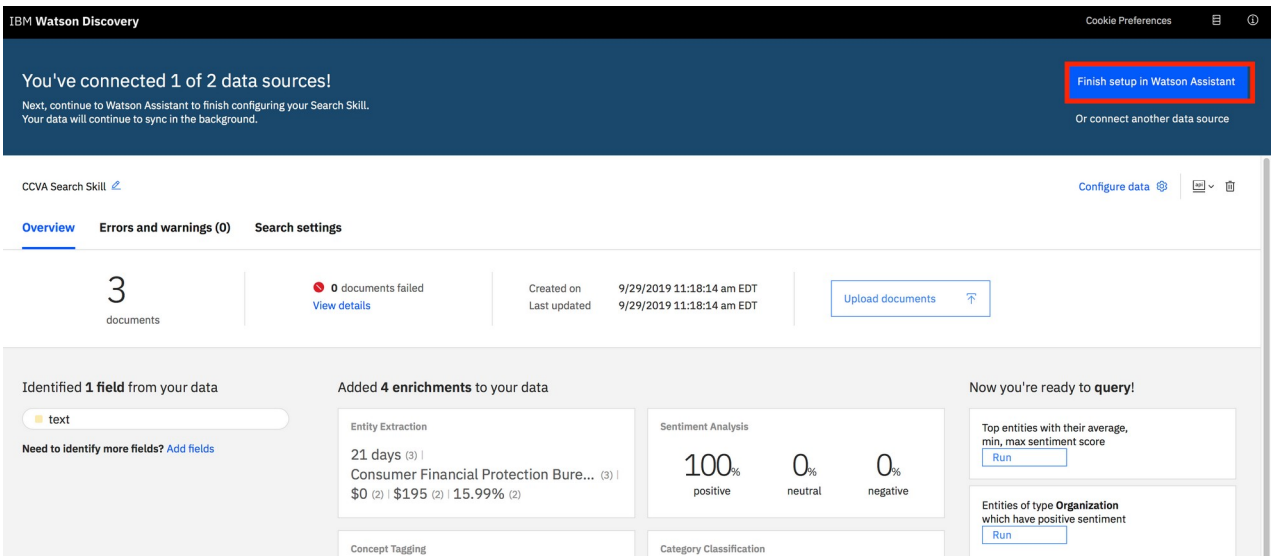
Step 43 After the upgrade **click** Add search skill.

Step 44 **Name it** f.e. `CSAD Search` and **click** *Continue*. Your previously created Discovery service will be displayed with the already defined collections.

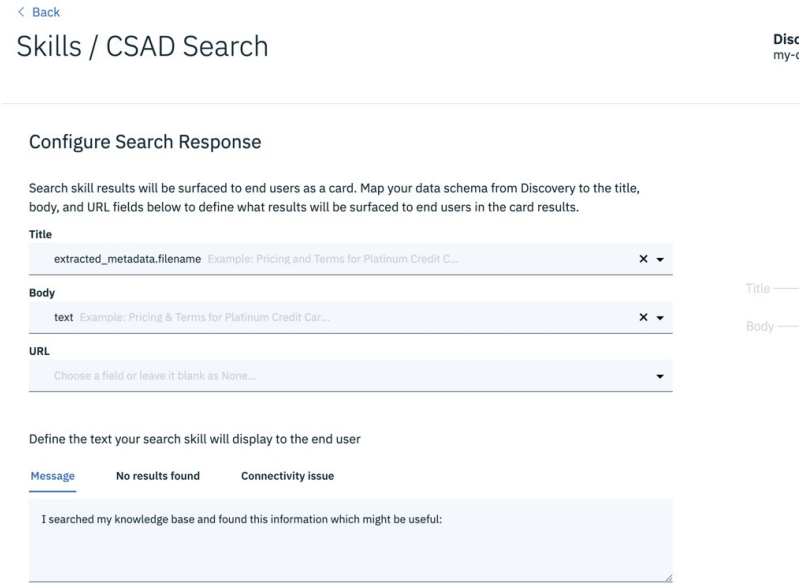
Step 45 **Click** Create new collection. Then **click** Let's get started. You cannot have more than two collection is your lite service.

Step 46 **Click** Upload documents. You can find these documents here or in the resources folder of the Github repository. **Name** your collection `CSAD Search`. Now upload the three PDF documents. **This may take a while!!**.

Step 47 Once the documents have been ingested, **click** *Finish setup in Watson Assistant*

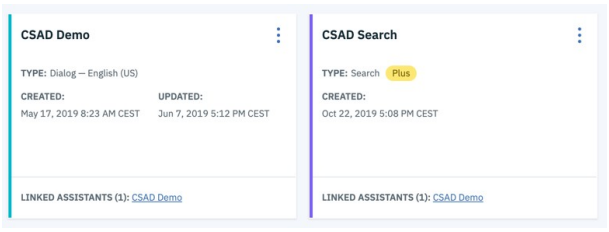


Step 48 Make sure the new collection is selected and **click** *Configure*.



For **Title** select the field: *extracted_metadata.filename*
For **Body**, select the filed: *text*.
For **url**, leave empty.

Then **click** *Create*.



Connect the Search Skill to a place in the dialog

Step 49 **Open** the *Intents* section of your Assistant skill

Step 50 **Click** Create intent enter the following values and **click** Create intent

Field	Value
Intent name	Credit_Card_Info
Description	Search Skill integration

Step 51 The user examples are phrases that will help Watson recognize the new intent. (Enter multiple examples by **pressing** "Enter" or by **clicking** the *Add example*). When finished, **click** ← at the top of the page.

Field	Value
User example	are there any international fees if i use my card in paris atm fees for my credit card fees for using my card abroad what are the apr fees for my silver credit card What are the fees for using an ATM in another country?

Step 52 **Open** the *Entities* section of you Assistant skill

Step 53 **Click** Create entity, **enter** `card_info` and **click** Create entity

Step 54 **Click** Add value. When finished, **click** ← at the top of the page.

Field	Value	Synonym
Entity name	card_info	
Value	APR	annual percentage rate
	cash advance	advanced cash, money advance
	international fees	International charges, foreign fees, foreign charges, fees while abroad, foreign exchange
	annual fee	yearly fee, annual membership, yearly membership, yearly cost, entry fee

Step 55 **Open** the *Dialog* section of your Assistant skill

Step 56 **Click** the menu ⋮ on the *Book a Table* node and then **click** *Add node below*, with the following values.

Field	Value
Name this node…	Information for a Card
If bot recognizes:	#Credit_Card_Info

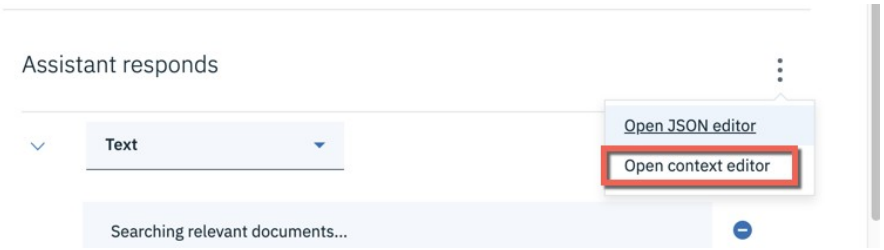
Step 57 **Click** Customize and **enable** *Multiple responses*. The click Apply

Step 58 Enter the following responses:

If Assistant recognizes	Respond with
@card_info:(annual fee)	Searching relevant documents…
@card_info:(international fees)	Searching relevant documents...
@card_info:(APR)	Searching relevant documents…
@card_info:(cash advance)	Searching relevant documents...
anything_else	Searching relevant documents...

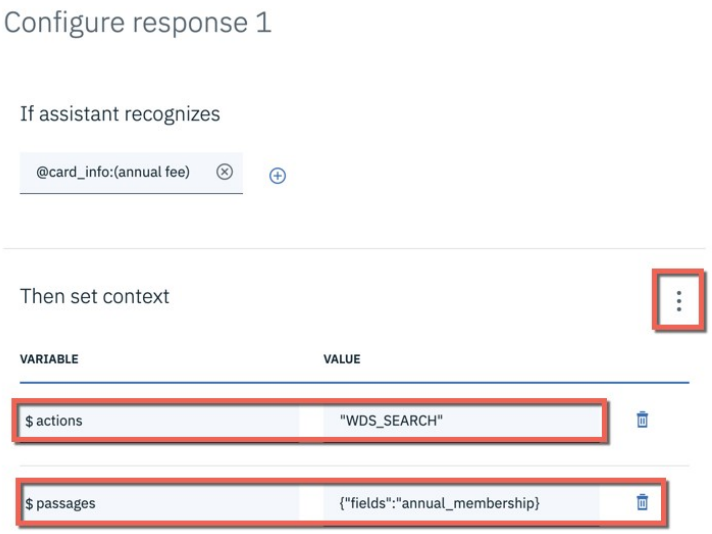
Step 59 Now customize the response for response 1, click ⚙

Step 60 Open the Context Editor

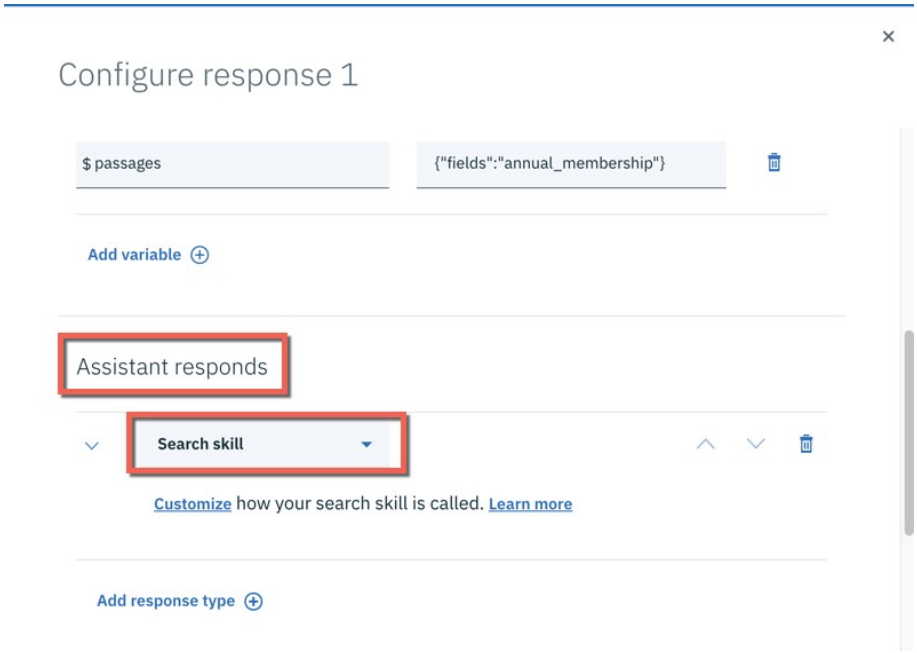


Step 61 Add two context variables

Variable	Value
actions	"WDS_SEARCH"
passages	{"fields":"annual_membership"}



Step 62 **Change** the *Assistant responds* option to *Search skill*. **Click** Save



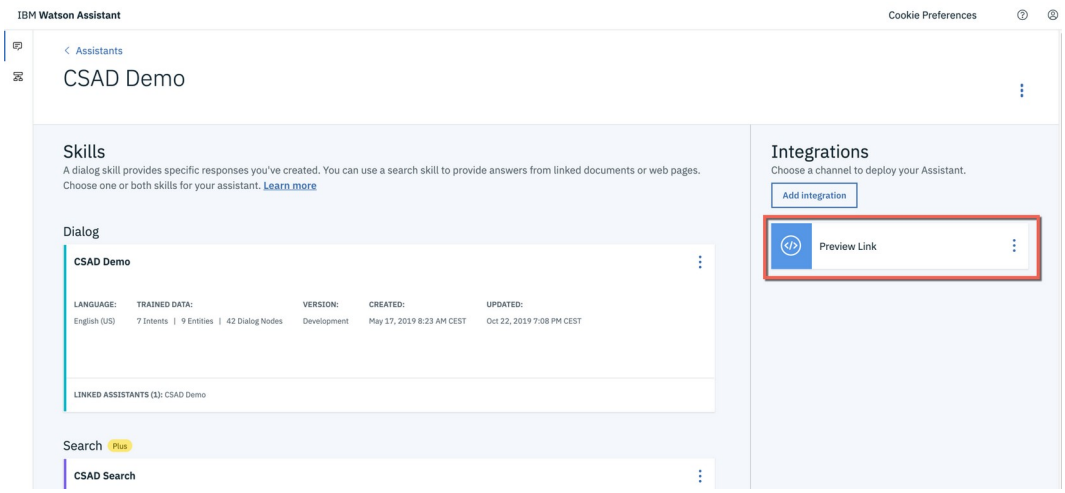
Step 63 Do the same for the second response, but for passages enter the following:

Variable	Value
actions	"WDS_SEARCH"
passages	{"fields":"international_fees"}

Step 64 Then **click** × to close the dialog.

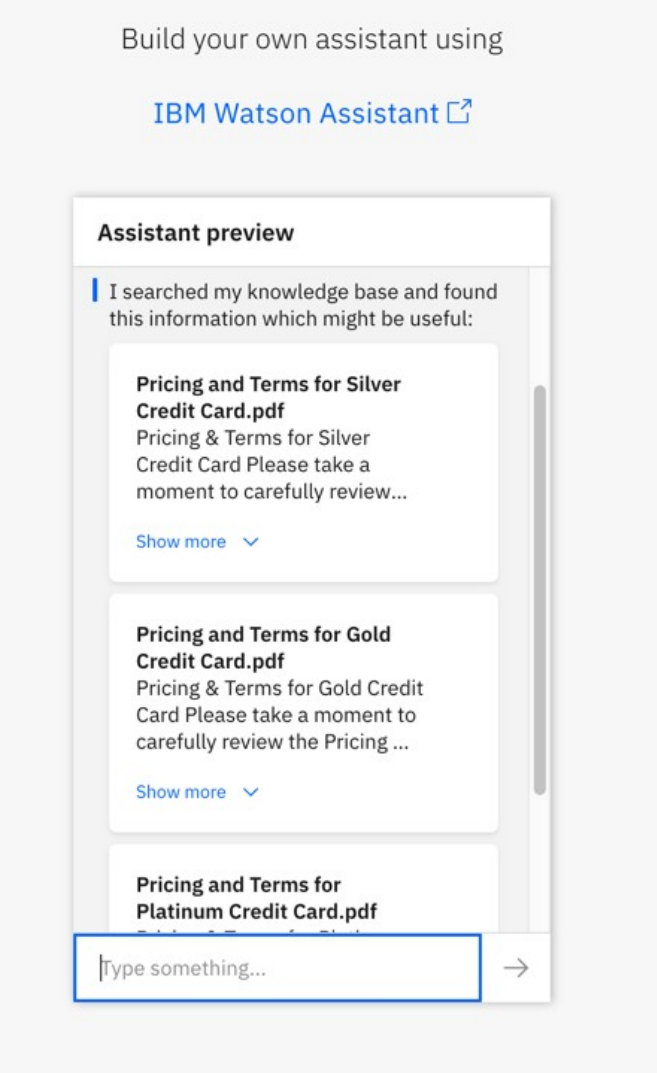
Step 65 All other responses just return "*Searching relevant documents...*"

Step 66 **Open** your Assistant and **click** the *Preview Link*. **Click** the *URL shown* to open the Assistant in the browser.



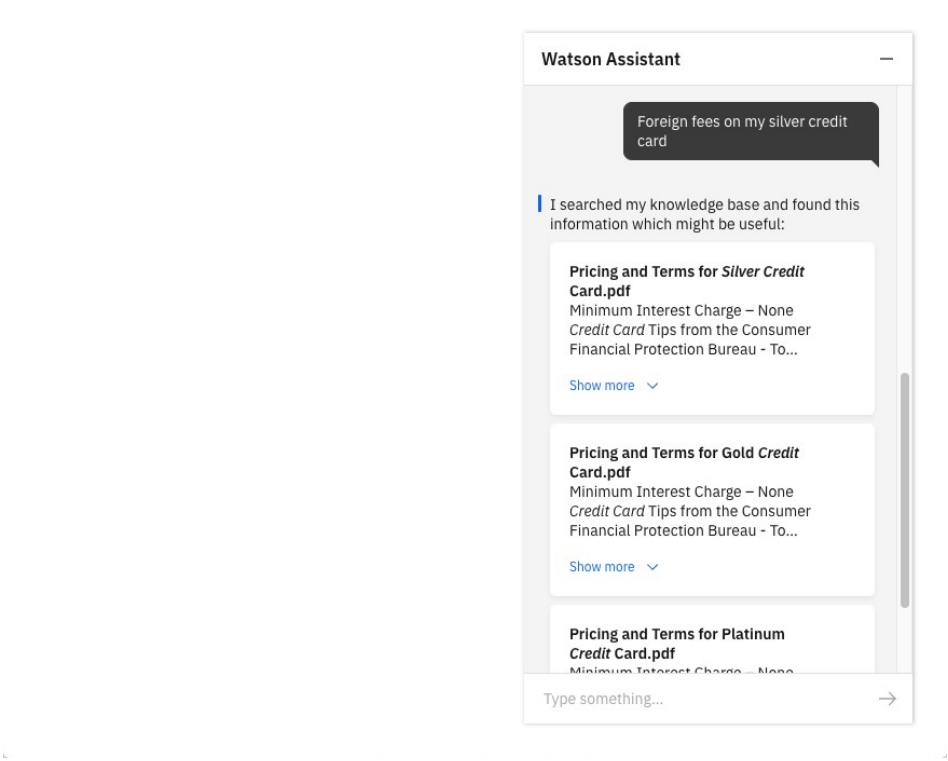
Step 67 Enter the following into your Assistant preview:

```
Foreign fees on my silver credit card
```



Step 68 In Lab 3a you have integrated the chat bot into your web site using the Webchat feature of Watson Assistant. The page should no look like the following when "`Foreign fees on my credit card`" is entered.

Home Page to integrate the Chatbot



You have successfully completed this lab.