

PROJECT REPORT

PROJECT TITLE: Intelligent Customer Help Desk With Smart Document Understanding

CATEGORY: Artificial Intelligence

MADE BY: Hemant Joshi

EMAIL ID: hemantjoshi344@gmail.com

Internship @smartinternz.com

Table Of Contents

- 1. Introduction**
- 2. Literature Survey**
- 3. Theoretical Analysis**
- 4. Experimental Investigation**
- 5. Flowcharts**
- 6. Results**
- 7. Application**
- 8. Conclusion**
- 9. Future Scope**
- 10. Advantages and Disadvantages**
- 11. References**

1.Introduction

1.1. Overview of the Project:

We will be able to write an application that leverages multiple Watson AI services like Discovery, Assistant, Cloud Function and Node Red. By the end of the project we will learn best practices of combining Watson Services, and how they can build interactive retrieval system with discovery + assistant.

- Project Requirements: Python, IBM Cloud, IBM Watson, Node-Red
- Functional Requirements: IBM Cloud
- Technical Requirements: Python, Watson AI, ML
- Software Requirements: Watson Assistant, Watson Discovery, IBM Cloud Functions
- Project Deliverables: Intelligent Customer Help Desk With Smart Document Understanding
- Project Duration: 30 Days

1.2. Purpose:

The typical customer care chatbot can answer simple questions, such as store locations and hours, directions, and maybe even making appointments. When a question falls outside of the scope of the pre-determined question set, the option is typically to tell the customer the question isn't valid or offer to speak to a real person.

In this project, there will be another option. If the customer question is about the operation of a device, the application shall pass the question onto Watson Discovery Service, which has been pre-loaded with the device's owners manual. So now, instead of "Would you like to speak to a customer

representative?” we can return relevant sections of the owners manual to help solve our customers’ problems.

To take it a step further, the project shall use the Smart Document Understanding feature of Watson Discovery to train it on what text in the owners manual is important and what is not. This will improve the answers returned from the queries.

Scope Of Work:

- Create a customer care dialog skill in Watson Assistant
- Use Smart Document Understanding to build an enhanced Watson Discovery collection
- Create an IBM Cloud Functions web action that allows Watson Assistant to post queries to Watson Discovery
- Build a web application with integration to all these services & deploy the same on IBM Cloud Platform

2.Literature Survey

2.1. Existing Problem:

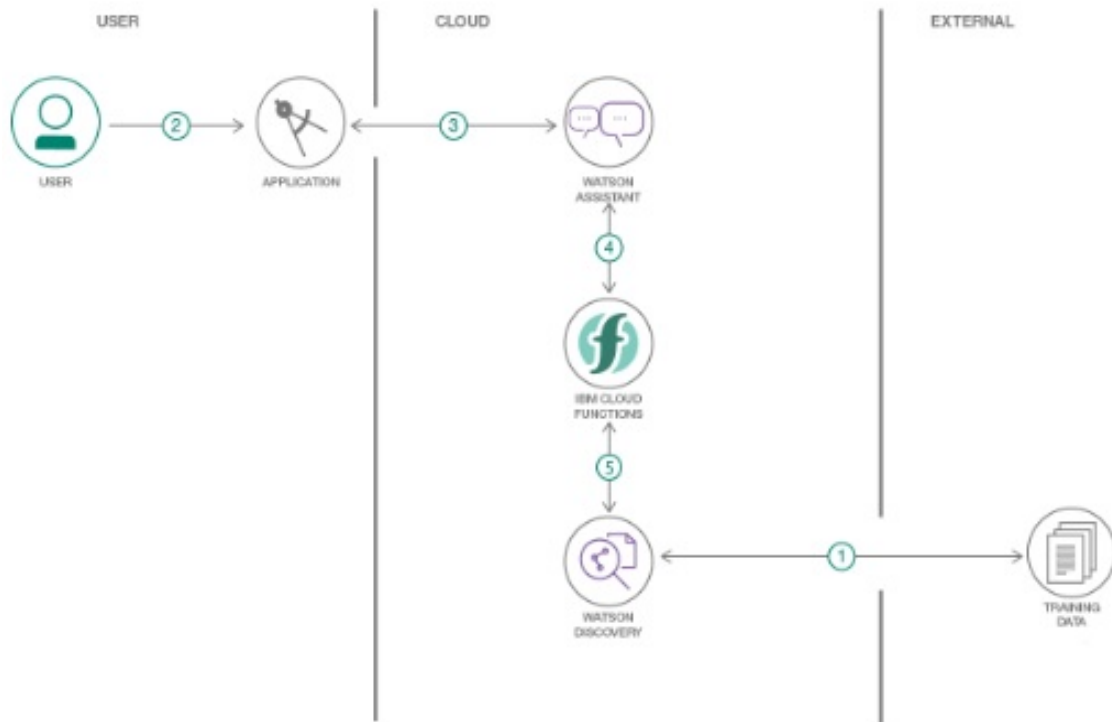
Generally chatbots are equipped with a certain set of questions such as store locations and operating hours, directions and booking appointments. If the question or the user input lies out of the scope of the pre-existing question set, the chatbot returns an error message asking the user to rephrase the question or ask the user if they would like to speak to a customer support executive. An efficient chatbot should minimise the volume of customer support redirections to the representatives.

2.2. Proposed Solution:

For the above-mentioned problem, we have to put a virtual agent in chatbot so it can understand the queries that are posted by customers. The virtual agent should be trained from some insight based on company backgrounds, working hours, store locations and product related information. In this project, I used Watson Discovery to achieve the above solution.

3.Theoretical Analysis

3.1. Flow Diagram :



1. The document is annotated using Watson Discovery SDU
2. The user interacts with the backend server via the app UI. The frontend app UI is a chatbot that engages the user in a conversation.
3. Dialog between the user and backend server is coordinated using a Watson Assistant dialog skill.
4. If the user asks a product operation question, a search query is passed to a predefined IBM Cloud Functions action.
5. The Cloud Functions action will query the Watson Discovery service and return the results.

3.2. Software/ Hardware Designing:

1. Create IBM Cloud services
2. Configure Watson Discovery
3. Create IBM Cloud Functions action
4. Configure Watson Assistant
5. Get IBM Cloud services credentials and add to .env file
6. Run the application

4Experimental Investigation

4.1. Create IBM Cloud Services:

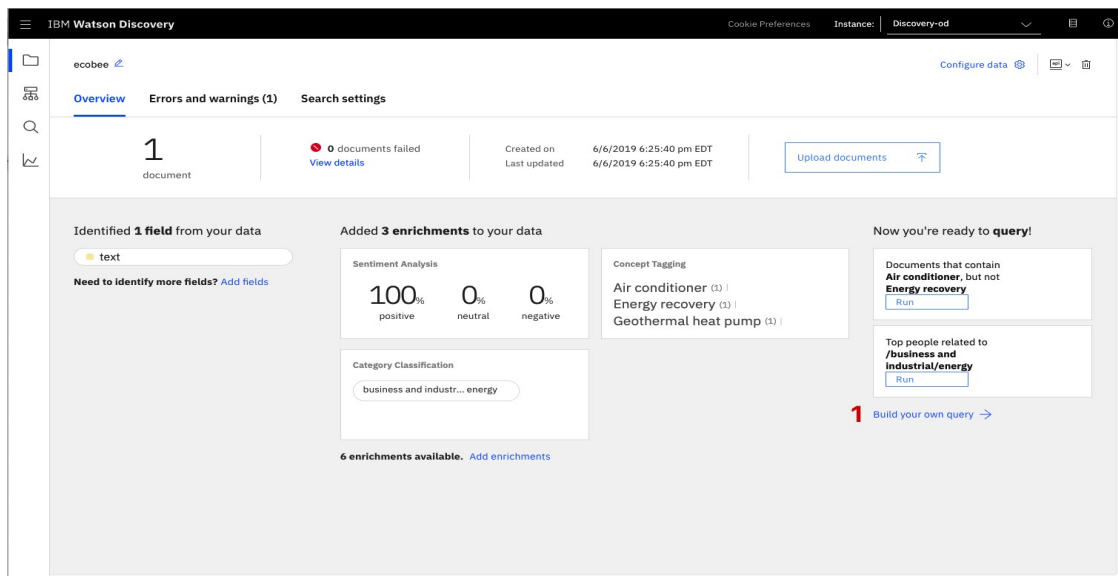
Create the following services:

- Watson Discovery
- Watson Assistant
- Node-Red Dashboard

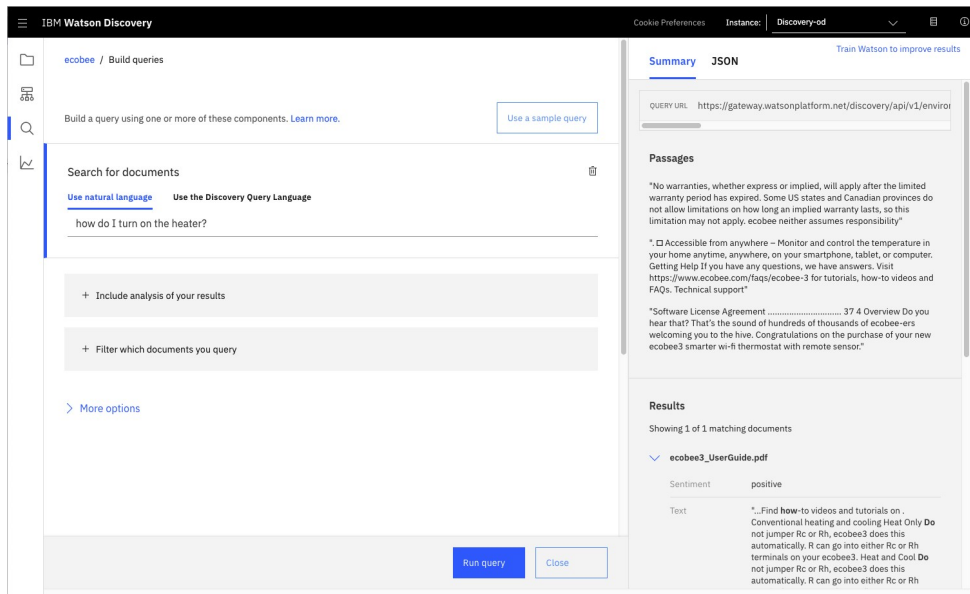
4.2. Configure Watson Discovery:

Import the document

Launch the Watson Discovery tool and create a new data collection by selecting the Upload your own data option. Give the data collection a unique name. When prompted, select and upload the ecobee3_UserGuide.pdf file located in the data directory of your local repo. The Ecobee is a popular residential thermostat that has a wifi interface and multiple configuration options. Before applying SDU to our document, lets do some simple queries on the data so that we can compare it to results found after applying SDU.



Click the Build your own query [1] button.

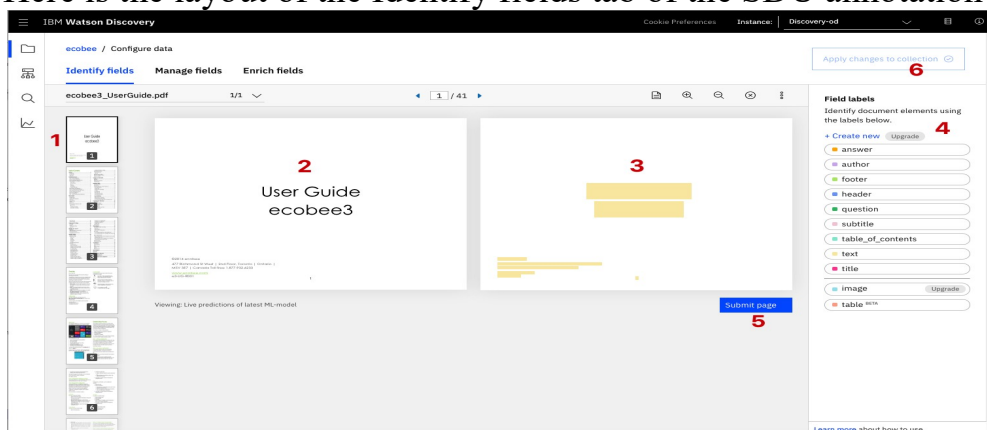


Enter queries related to the operation of the thermostat and view the results. As you will see, the results are not very useful, and in some cases, not even related to the question.

Annotate with SDU

Now let's apply SDU to our document to see if we can generate some better query responses.

From the Discovery collection panel, click the Configure data button (located in the top right corner) to start the SDU process. Here is the layout of the Identify fields tab of the SDU annotation panel:



The goal is to annotate all of the pages in the document so Discovery can learn what text is important, and what text can be ignored.

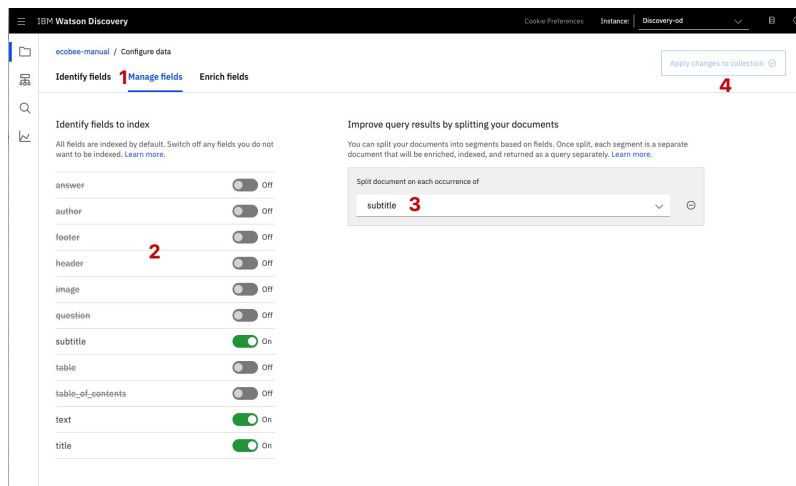
- [1] is the list of pages in the manual. As each is processed, a green check mark will appear on the page.
- [2] is the current page being annotated.
- [3] is where you select text and assign it a label.
- [4] is the list of labels you can assign to the page text.
- Click [5] to submit the page to Discovery.
- Click [6] when you have completed the annotation process.

As you go through the annotations one page at a time, Discovery is learning and should start automatically updating the upcoming pages. Once you get to a page that is already correctly annotated, you can stop, or simply click Submit [5] to acknowledge it is correct. The more pages you annotate, the better the model will be trained.

For this specific owner's manual, at a minimum, it is suggested to mark the following:

- The main title page as title
- The table of contents (shown in the first few pages) as table_of_contents
- All headers and sub-headers (typed in light green text) as a subtitle
- All page numbers as footers
- All warranty and licensing information (located in the last few pages) as a footer
- All other text should be marked as text.

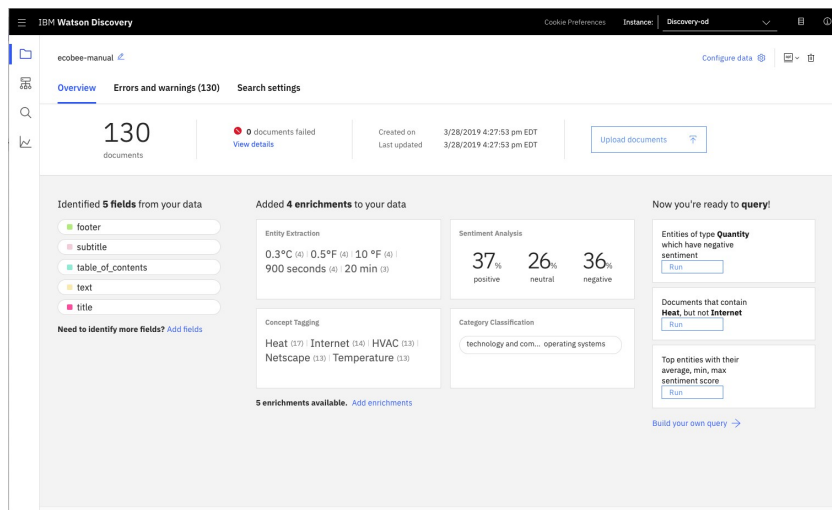
Once you click the Apply changes to collection button [6], you will be asked to reload the document. Choose the same owner's manual .pdf document as before. Next, click on the Manage fields [1] tab.



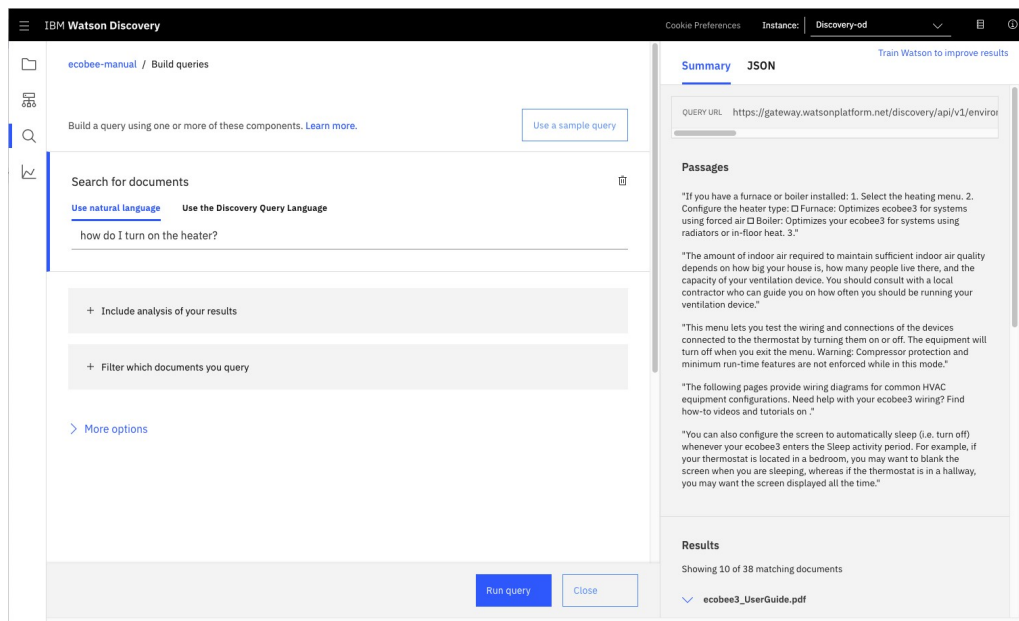
- [2] Here is where you tell Discovery which fields to ignore. Using the on/off buttons, turn off all labels except subtitles and text.
- [3] is telling Discovery to split the document apart, based on subtitle.
- Click [4] to submit your changes.

Once again, you will be asked to reload the document.

Now, as a result of splitting the document apart, your collection will look very different:



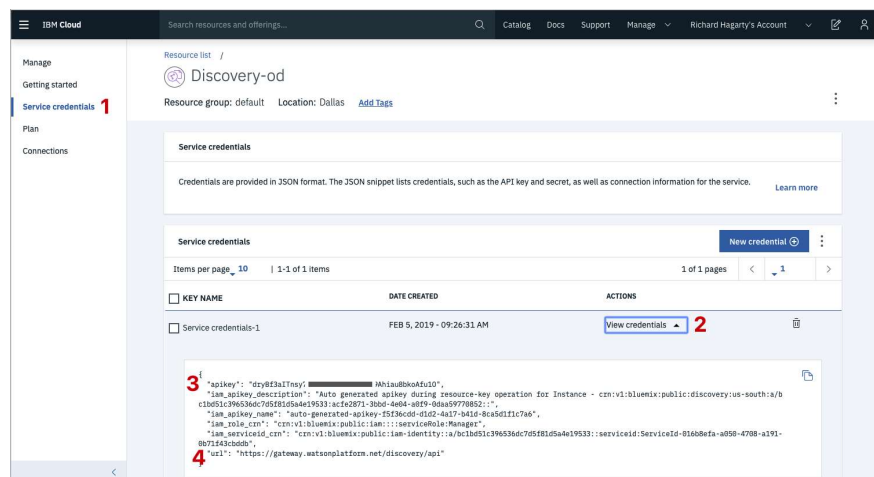
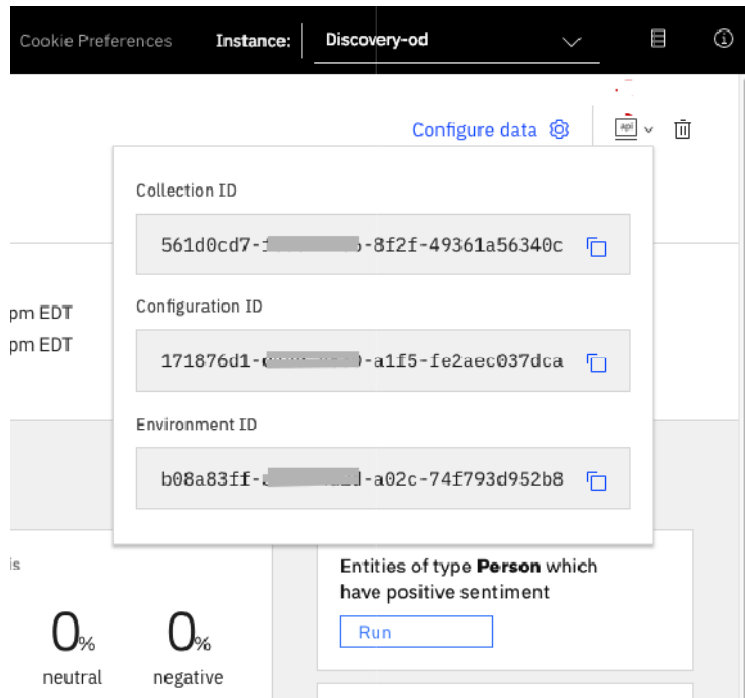
Return to the query panel (click Build your own query) and see how much better the results are.



Store credentials for future use

In upcoming steps, you will need to provide the credentials to access your Discovery collection. The values can be found in the following locations.

The Collection ID and Environment ID values can be found by clicking the dropdown button [1] located at the top right side of your collection panel:

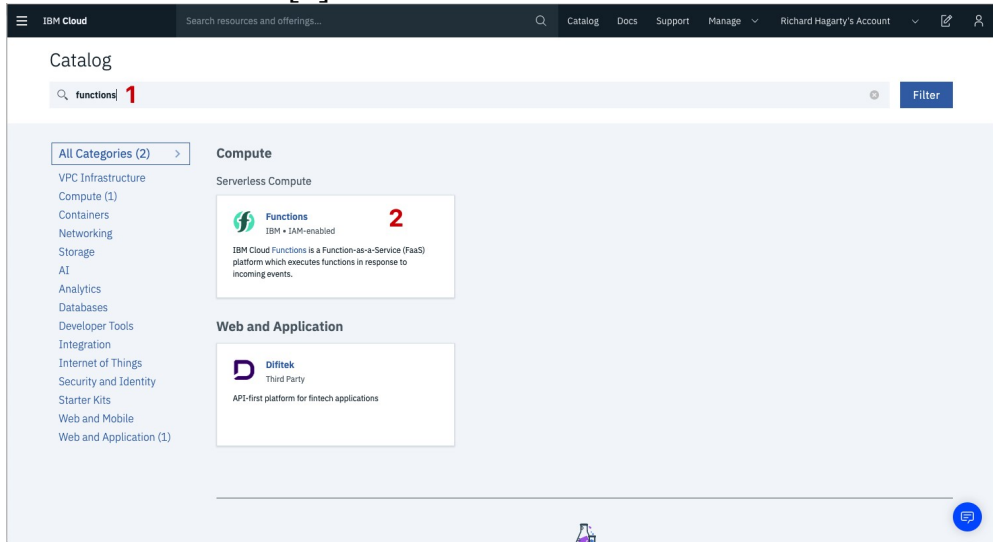


Click the View credentials [2] drop-down menu to view the IAM apikey [3] and URL endpoint [4] for your service.

4.3. Create IBM Cloud Functions action

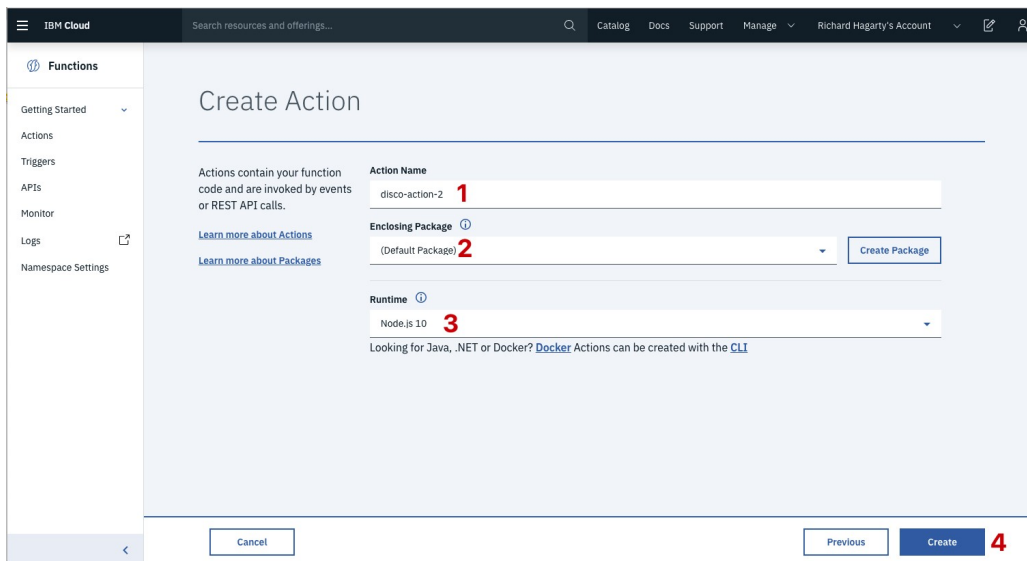
Now let's create the web action that will make queries against our Discovery collection.

Start the IBM Cloud Functions service by selecting Create Resource from the IBM Cloud dashboard. Enter functions as the filter [1], then select the Functions card [2]:

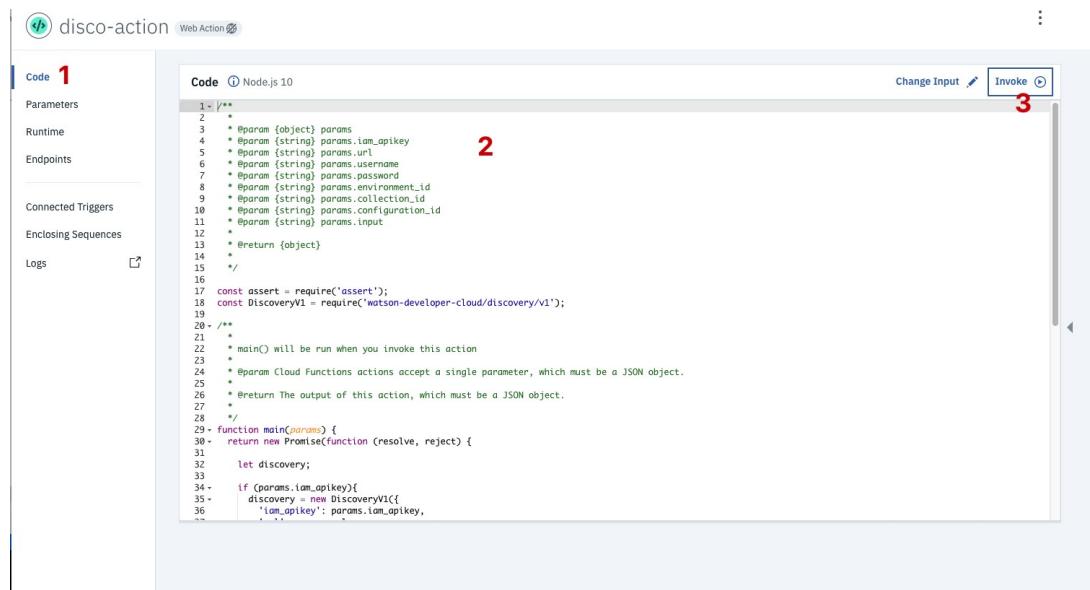


From the Functions main panel, click on the Actions tab. Then click on Create. From the Create panel, select the Create Action option.

On the Create Action panel, provide a unique Action Name [1], keep the default package [2], and select the Node.js 10 [3] runtime. Click the Create button [4] to create the action.



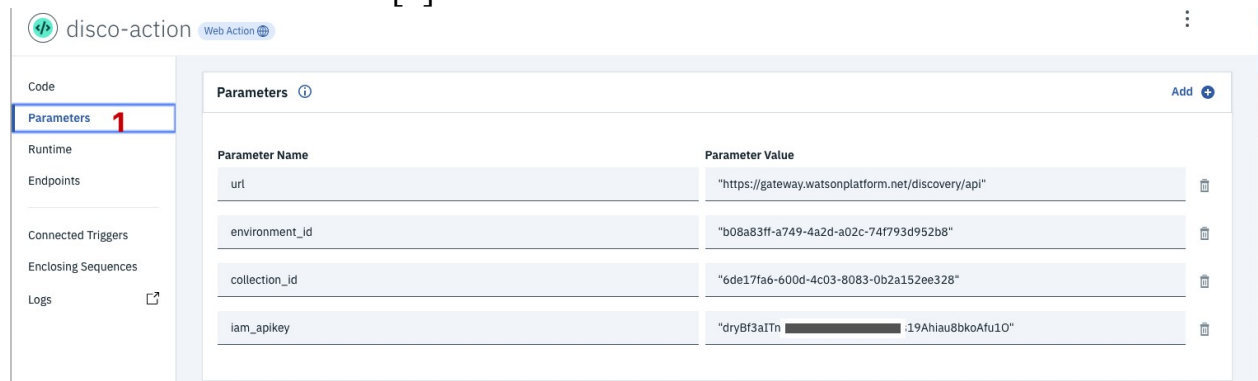
Once your action is created, click on the Code tab [1]:



In the code editor window [2], cut and paste in the code from the disco-action.js file found in the actions directory of your local repo. The code is pretty straight-forward - it simply connects to the Discovery service, makes a query against the collection, then returns the response.

If you press the Invoke button [3], it will fail due to credentials not being defined yet. We'll do this next.

Select the Parameters tab [1]:



Add the following keys:

- url
- environment_id
- collection_id
- iam_apikey

For values, please use the values associated with the Discovery service you created in the previous step.

Note: Make sure to enclose your values in double quotes.

Now that the credentials are set, return to the Code panel and press the Invoke button again. Now you should see actual results returned from the Discovery service:

The screenshot shows the IBM Cloud Functions console for the 'disco-action' function. The 'Code' panel on the left displays the Node.js code for the function. The 'Invoke' button is visible. The 'Activations' panel on the right shows the results of a successful invocation.

```
1- /**
2-  *
3-  * @param {object} params
4-  * @param {string} params.iam_apikey
5-  * @param {string} params.url
6-  * @param {string} params.username
7-  * @param {string} params.password
8-  * @param {string} params.environment_id
9-  * @param {string} params.collection_id
10-  * @param {string} params.configuration_id
11-  * @param {string} params.input
12-  *
13-  * @return {object}
14-  */
15- */
16-
17- const assert = require('assert');
18- const DiscoveryV1 = require('watson-developer-cloud/discovery/v1');
19-
20- /**
21-  *
22-  * main() will be run when you invoke this action
23-  *
24-  * @param Cloud Functions actions accept a single parameter, which must be a JSON object.
25-  *
26-  * @return The output of this action, which must be a JSON object.
27-  */
28- */
29- function main(params) {
30-   return new Promise(function (resolve, reject) {
31-
32-     let discovery;
33-
34-     if (params.iam_apikey) {
35-       discovery = new DiscoveryV1({
36-         'iam_apikey': params.iam_apikey,
37-         'url': params.url,
38-         'version': '2019-03-25'
39-       });
40-     }
41-   });
42- }
```

Activations

disco-action 1050 ms 6/6/2019, 10:45:14

Activation ID:
e1bfe0ff21544e95b5c0ff21549e85a1

Results:

```
{
  "matching_results": 14,
  "passages": [],
  "results": [
    {
      "enriched_text": {
        "categories": [
          {
            "label": "/technology and computing/operating systems",
            "score": 0.842265
          },
          {
            "label": "/technology and computing/hardware/computer",
            "score": 0.835879
          },
          {
            "label": "/technology and computing/hardware/computer peripherals/computer monitors",
            "score": 0.832254
          }
        ],
        "concepts": [
          {
            "dbpedia_resource": "http://dbpedia.org/resource/IPhone",
            "relevance": 0.917306,
            "text": "IPhone"
          },
          {
            "dbpedia_resource": "http://dbpedia.org/resource/Personal_digital_assistant",
            "relevance": 0.887088,
            "text": "Personal digital assistant"
          }
        ]
      }
    }
  ]
}
```

Next, go to the Endpoints panel [1]:

The screenshot shows the IBM Cloud Functions console for the 'disco-action' function, specifically the 'Endpoints' panel. It displays the 'Web Action' configuration and the 'REST API' endpoints.

Web Action

2 ☒ **Enable as Web Action** Allow your Cloud Functions actions to handle HTTP events. Learn more about Web Actions.

☐ **Raw HTTP handling** When enabled your Action receives requests in plain text instead of a JSON body

HTTP METHOD	AUTH	URL
ANY	Public	3 https://us-south.functions.cloud.ibm.com/api/v1/web/IBM%20Cloud%20Storage_DSX-journey-2/default/disco-action

REST API

HTTP METHOD	AUTH	URL
POST	API-KEY	https://us-south.functions.cloud.ibm.com/api/v1/namespaces/IBM%20Cloud%20Storage_DSX-journey-2/actions/disco-action

CURL

```
4 curl -u API-KEY -X POST https://us-south.functions.cloud.ibm.com/api/v1/namespaces/IBM%20Cloud%20Storage_DSX-journey-2/actions/disco-action?blocking=true
```

Click the checkbox for Enable as Web Action [2]. This will generate a public endpoint URL [3].

Take note of the URL value [3], as this will be needed by Watson Assistant in a future step.

To verify you have entered the correct Discovery parameters, execute the provided curl command [4]. If it fails, re-check your parameter values.

NOTE: An IBM Cloud Functions service will not show up in your dashboard resource list. To return to your defined Action, you will need to access Cloud Functions by selecting Create Resource from the main dashboard panel (as shown at the beginning of this step).

4.5. Configure Watson Assistant

As shown below, launch the Watson Assistant tool and create a new dialog skill. Select the Use sample skill option as your starting point.

This dialog skill contains all of the nodes needed to have a typical call center conversation with a user.

Add new intent

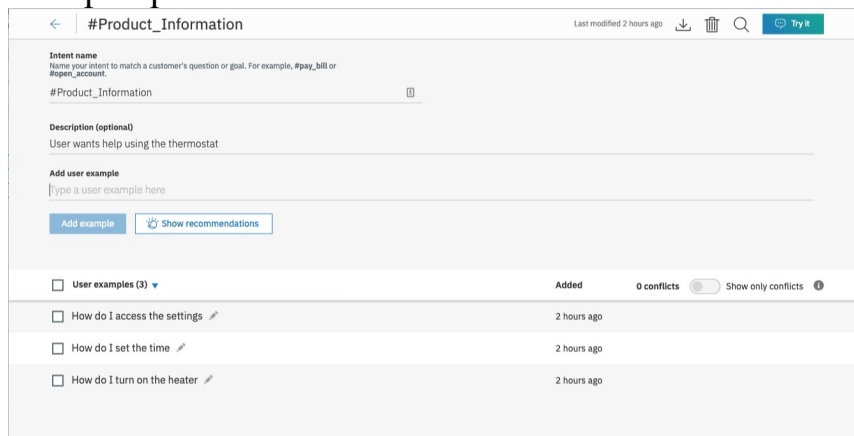
The default customer care dialog does not have a way to deal with any questions involving outside resources, so we will need to add this.

Create a new intent that can detect when the user is asking about operating the Ecobee thermostat.

From the Customer Care Sample Skill panel, select the Intents tab.

Click the Create intent button.

Name the intent #Product_Information, and at a minimum, enter the following example questions to be associated with it.



The screenshot shows the Watson Assistant 'Intents' tab for a skill named '#Product_Information'. The interface includes a header with a back arrow, the intent name, and a 'Try It' button. Below the header, there are fields for 'Intent name' (pre-filled with '#Product_Information'), 'Description (optional)' (pre-filled with 'User wants help using the thermostat'), and 'Add user example' (with a text input field and 'Add example' and 'Show recommendations' buttons). At the bottom, there is a table of 'User examples' with columns for checkboxes, the example text, and the time added. The table shows three examples: 'How do I access the settings', 'How do I set the time', and 'How do I turn on the heater', all added '2 hours ago'.

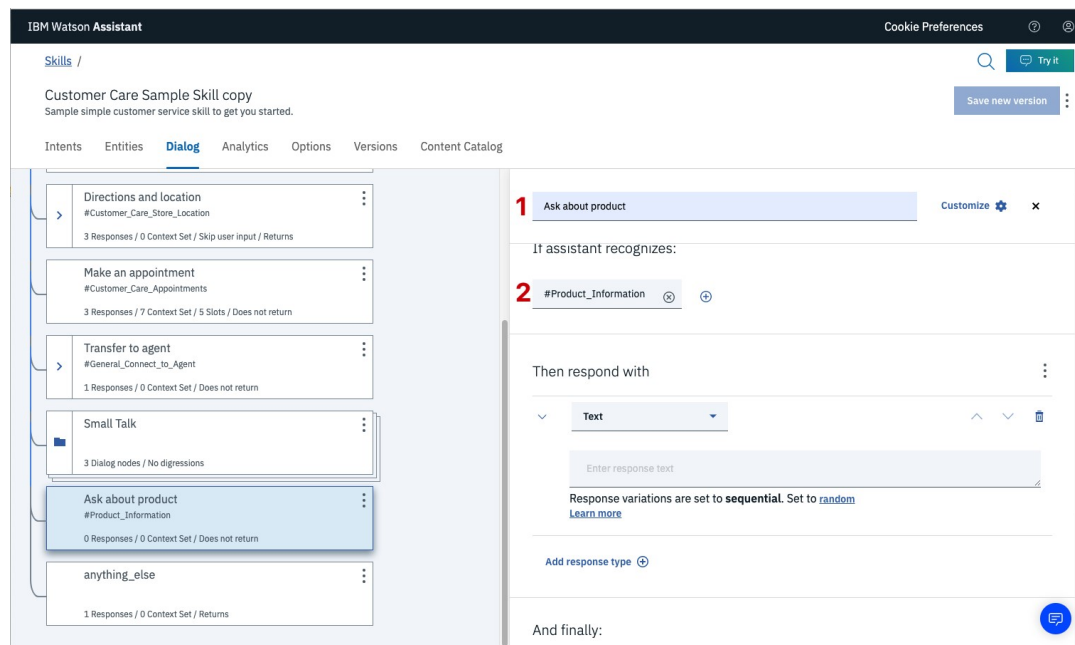
<input type="checkbox"/>	User examples (3) ▼	Added	0 conflicts	Show only conflicts ⓘ
<input type="checkbox"/>	How do I access the settings ↗	2 hours ago		
<input type="checkbox"/>	How do I set the time ↗	2 hours ago		
<input type="checkbox"/>	How do I turn on the heater ↗	2 hours ago		

Create new dialog node

Now we need to add a node to handle our intent. Click on the Dialog [1] tab, then click on the drop down menu for the Small Talk node [2], and select the Add node below [3] option.

The screenshot shows the IBM Watson Assistant interface. At the top, there's a dark blue header with "IBM Watson Assistant". Below it, a breadcrumb trail shows "Skills /". The main title is "Customer Care Sample Skill copy" with a subtitle "Sample simple customer service skill to get you started." Below this is a navigation bar with tabs: "Intents", "Entities", "1 Dialog" (highlighted with a red '1'), "Analytics", "Options", "Versions", and "Content Catalog". The main area displays a list of dialog nodes. The nodes are: "Directions and location" (with a blue arrow icon), "Make an appointment", "Transfer to agent" (with a blue arrow icon), "Small Talk" (with a blue square icon), and "anything_else". Each node shows its name, a unique identifier, and statistics like "3 Responses / 0 Context Set / Skip user input / Returns". A red '2' points to the three-dot menu icon next to the "Small Talk" node. A context menu is open, showing options: "Add node to folder", "Add node above", "Add node below" (highlighted with a blue bar and a red '3'), "Add folder", "Move", "Duplicate", "Jump to", and "Delete".

Name the node "Ask about product" [1] and assign it our new intent [2].

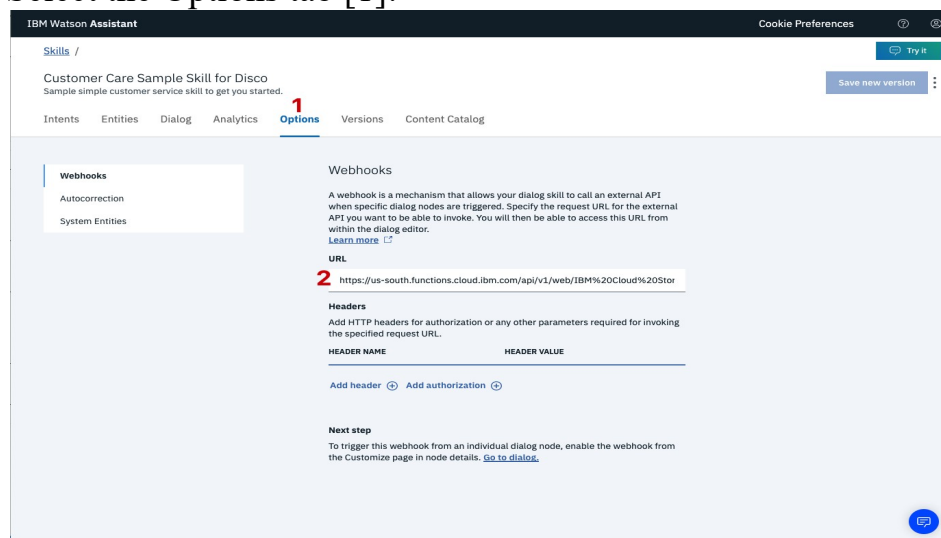


This means that if Watson Assistant recognizes a user input such as "how do I set the time?", it will direct the conversation to this node.

Enable webhook from Assistant

Set up access to our WebHook for the IBM Cloud Functions action you created in Step #4.

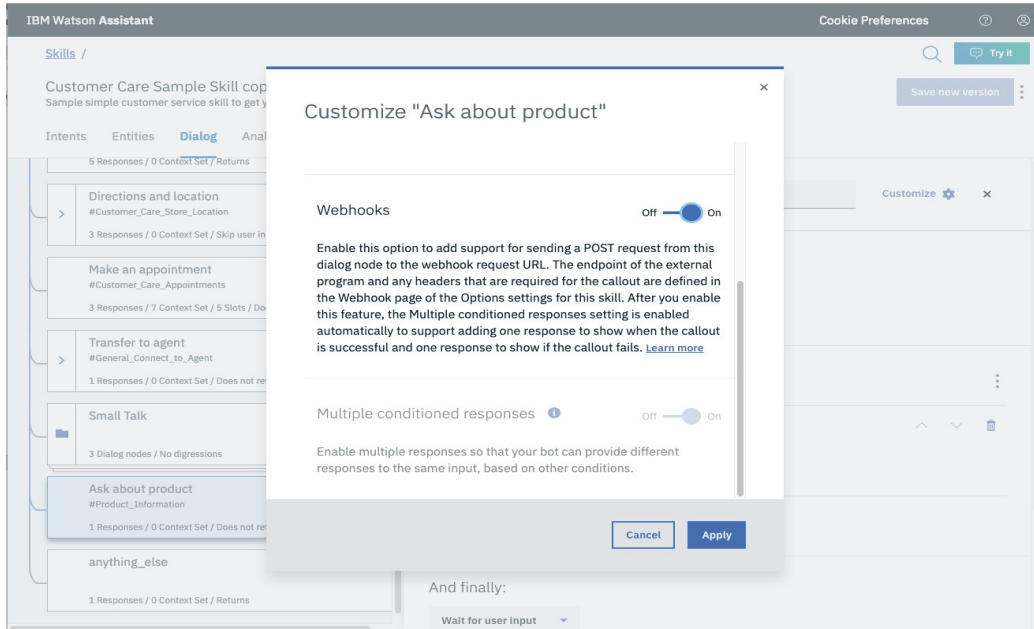
Select the Options tab [1]:



Enter the public URL endpoint for your action [2].

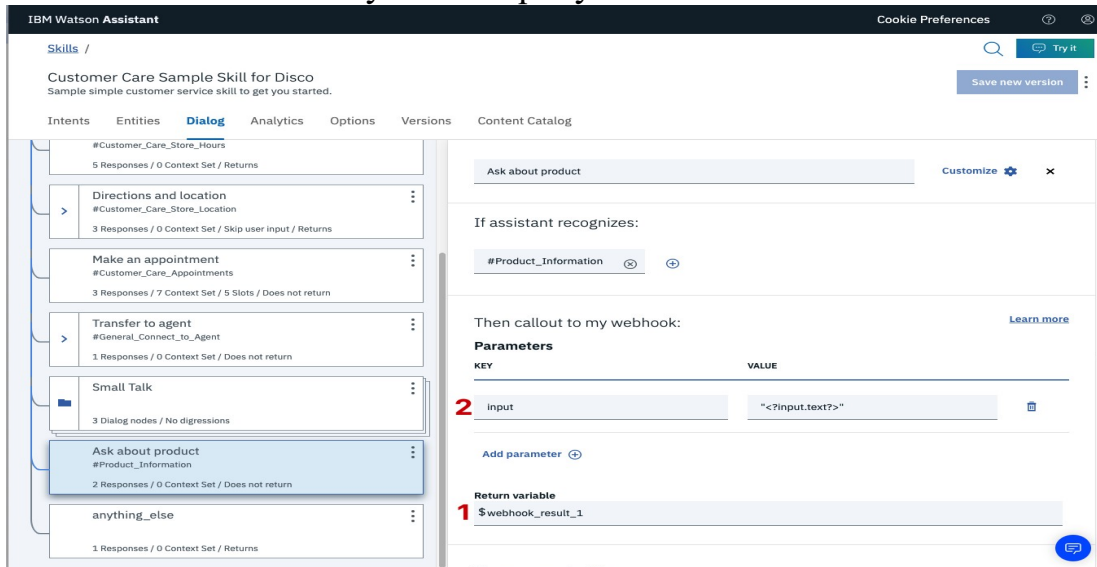
Important: Add .json to the end of the URL to specify the result should be in JSON format.

Return to the Dialog tab, and click on the Ask about product node. From the details panel for the node, click on Customize, and enable Webhooks for this node:



Click Apply.

The dialog node should have a Return variable [1] set automatically to \$webhook_result_1. This is the variable name you can use to access the result from the Discovery service query.



You will also need to pass in the users question via the parameter input [2]. The key needs to be set to the value:

```
"<?input.text?>"
```

If you fail to do this, Discovery will return results based on a blank query.

Optionally, you can add these responses to aid in debugging:

Return variable

```
$webhook_result_1
```

Then respond with

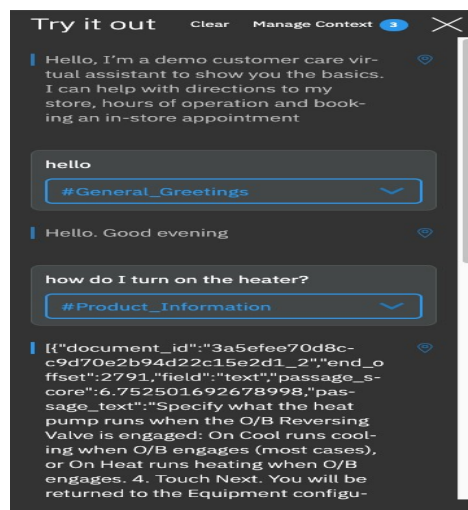
	IF ASSISTANT RECOGNIZES	RESPOND WITH		
1	<pre>\$webhook_result_1</pre>	<pre>\$webhook_result_1</pre>	⚙️	🗑️
2	<pre>anything_else</pre>	<pre>Try again later</pre>	⚙️	🗑️

[Add response](#) (+)

Test in Assistant Tooling

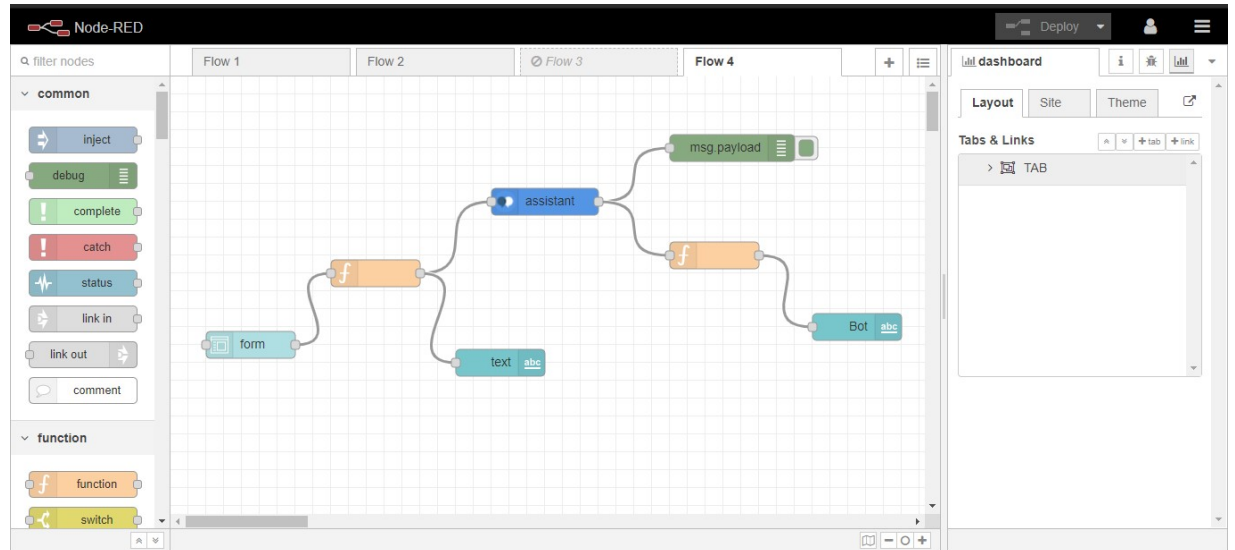
From the Dialog panel, click the Try it button located at the top right side of the panel.

Enter some user input:

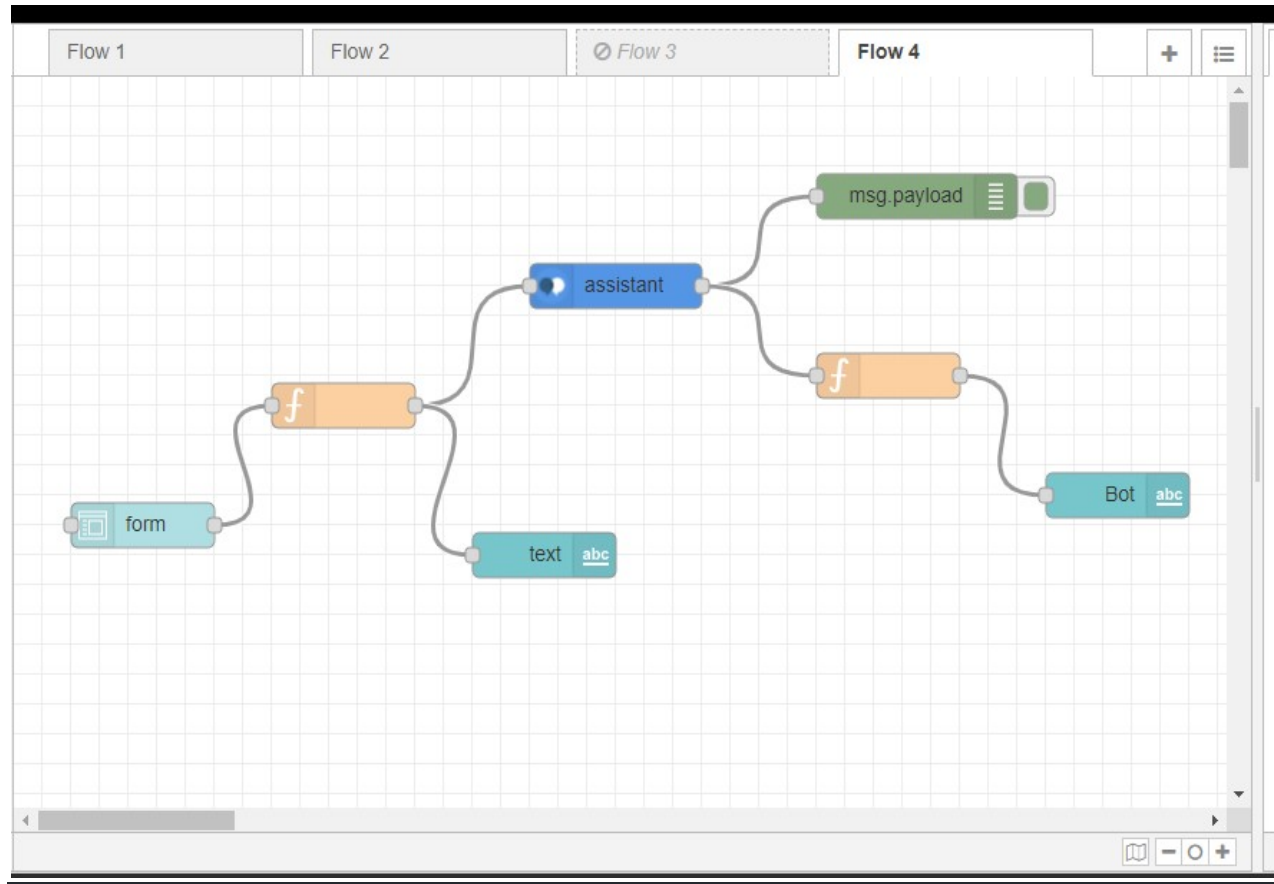


4.6.Create a Node-RED flow :

Use Node-RED to create a UI and to link all the services.



5. Flow Chart



6.Test Results

TAB

Query Input

Enter the Name
Hemant Joshi

Enter the Query
Hello

SUBMIT

CANCEL

textHello

Answers

BotHello. How can I help this evening?

TAB

Query Input

Enter the Name
Hemant Joshi

Enter the Query
How do I turn on the heater

SUBMIT

CANCEL

textHow do I turn on the heater

Answers

Bot

You can customize the brightness of your ecobee3's screen. The brightness for both the active and standby screens can be configured independently. You can also configure the screen to automatically sleep (i.e. turn off) whenever your ecobee3 enters the Sleep activity period. For example, if your thermostat is located in a bedroom, you may want to blank the screen when you are sleeping, whereas if the thermostat is in a hallway, you may want the screen displayed all the time. On Thermostat: 1. Select Main Menu > Settings > Preferences 2. Select Screen brightness. 3. Adjust the values of the Active and Standby screen brightness. 4. Select Screen sleeps when I sleep if you want to make the screen blank during the Sleep activity period. If you have a furnace or boiler installed: 1. Select the heating menu. 2. Configure the heater type: ☐ Furnace: Optimizes ecobee3 for systems using forced air ☐ Boiler: Optimizes your ecobee3 for systems using radiators or in-floor heat. 3. Touch Next. You will be returned to the Equipment configuration menu. This menu lets you test the wiring and connections of the devices connected to the thermostat by turning them on or off. The equipment will turn off when you exit the menu. Warning: Compressor protection and minimum run-time features are not enforced while in this mode.

TAB

Query Input

Enter the Name
Hemant Joshi

Enter the Query
When the store will open

SUBMIT

CANCEL

text

When the store will open

Answers

Our hours are Monday to Friday 10am to 8pm and Friday and Saturday 11am to 6pm.

Bot

7.Advantages And Disadvantages

Advantages:

- Companies can use these to decrease the work flow to the representatives.
- Reduce the number of reps.
- Cost Efficient.
- Decrease in the number of calls diverted to representatives.
- Less work load on employees.

Disadvantages:

- Sometimes the chatbot misleads the customers.
- The discovery returns wrong results when not properly configured.
- Giving same answer for different sentiments.
- Sometimes is unable to connect the customer sentiments and intents.

8.Application

- It can be deployed in popular social media applications like Facebook, Slack and Telegram.
- Chatbot can be deployed at any website to clear the basic doubts of the customer.

9.Conclusion

By following the above-mentioned steps, we can create a basic chatbot which can help us to answer the basic questions of the customer or user related to location of the office, working hours and the information about the product. We successfully create the intelligent helpdesk smart chatbot using Watson Assistant, Watson Cloud Function, Watson Discovery and Node-Red.

10.Future Scope

We can import the pre-built node-red flow and can improve our UI, moreover we can make a data base and use it to show the recent chats to the customer. We can also improve the results of discovery by enriching it with more fields and doing the Smart Data Annotation more accurately. We can get the premium version to increase the scope of our chatbot in terms of the calla and requests. We can also include Watson text to audio and Speech to text services to access the chatbot handsfree. These are few of the future scopes which are possible.

11.References

- https://www.ibm.com/cloud/architecture/tutorials/cognitive_discovery
- <https://cloud.ibm.com/docs/assistant?topic=assistant-getting-started>
- <https://developer.ibm.com/recipes/tutorials/how-to-create-a-watson-chatbot-on-nodered/>
- <http://www.iotgyan.com/learning-resource/integration-of-watson-assistant-to-node-red>
- <https://github.com/IBM/watson-discovery-sdu-with-assistant>
- <https://www.youtube.com/watch?v=Jpr3wVH3FVA>