Project Report

Name : DHANUSH M (dhanush18245@gmail.com)

Title : Intelligent Customer Help Desk WithSmart Document Understanding

Category : Artificial Intelligence/ Machine Learning

Internship at smartinternz.com@2020

INTRODUCTION

Overview

We will be designing an application that leverages multiple Watson AIServices (Discovery, Assistant, Cloud function and Node Red). By the end of the project, we'll learn best practices of combining Watson services, and how they can be used to build interactive information retrieval systems with Discovery + Assistant.

- ➢ Project Requirements: Python, IBM Cloud, IBM Watson
- ➢ Functional Requirements: IBM cloud
- ➢ Technical Requirements: AI, ML, WATSON AI, PYTHON
- ➢ Software Requirements: Watson assistant, Watson discovery.
- ➢ Project Deliverables: Smartinternz Internship
- ➢ Project Team: Dhanush
- ➢ Project Duration:19 days

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 predetermined question set, the option is typically to tell the customer the question isn't valid or offer to speak to a real person. The purpose of this project is to build a customer helping chatbot such that if the customer question is about the operation of a device, the application shall pass the question onto WatsonDiscovery Service, which has been pre-loaded with the device's owners manual. So, insteadof "Would you like to speak to a customer representative?" we can return relevant sections of the owners manual to help solve our customers' problems.

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.

LITERATURE SURVEY

Existing problem

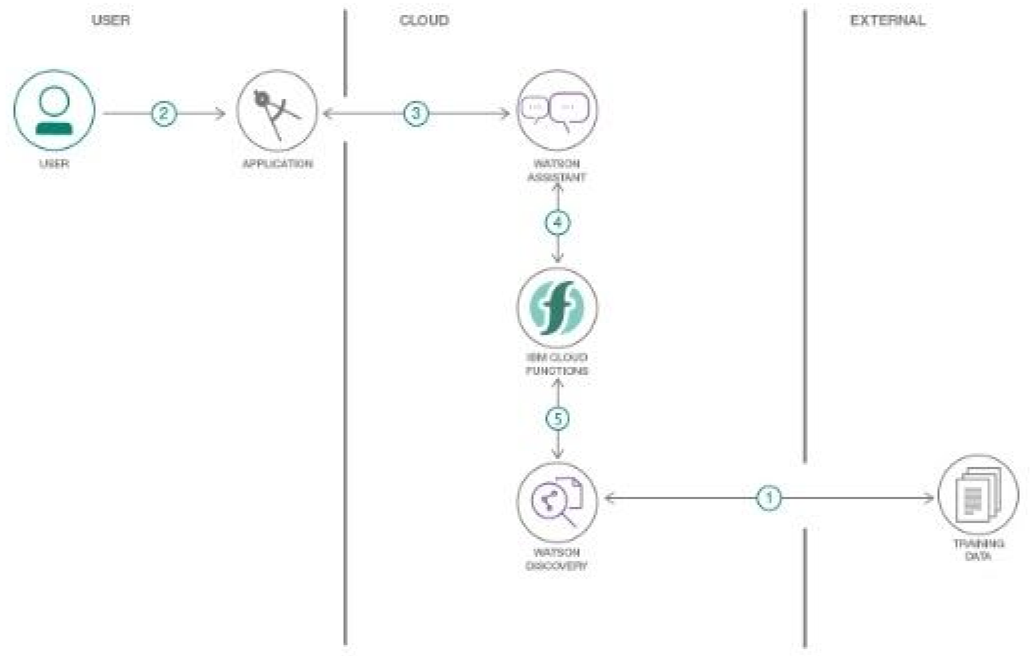
Generally Chatbots means getting input from users and getting only response questions and for some questions the output from bot will be like " try again", "I don't understand", "will you repeat again", and so on... and directs customer to customer agent but a good customer Chatbot should minimize involvement of customer agent to chat with customer to clarify his/her doubts. So to achieve this we should include a virtual agent in chatbot so that it will take care of real involvement of the customer agent and the customer can clarify his doubts with fast chatbots.

Proposed solution

For the above problem we are able to put a virtual agent in chatbot so it can understand the queries that are posted by customers. The virtual agent should train from some insight records based company background so it can answer queries supported by the merchandise or associated with the company. In other words, some styles of manual will be accustomed to train the bot using AI. Here I'm using Watson Discovery as a tool for implementing AI and getting trained by the owner's manual.

THEORETICAL ANALYSIS

Block/Flow Diagram



1. The document is annotated using Watson Discovery Smart Document Understanding
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.

Hardware / Software designing

1. Create IBM Cloud services
2. Configure Watson Discovery
3. Create IBM Cloud Functions action
4. Configure Watson Assistant
5. Create flow and configure node
6. Deploy and run Node Red app

EXPERIMENTAL INVESTIGATIONS

Create IBM Cloud services

Create the following services:

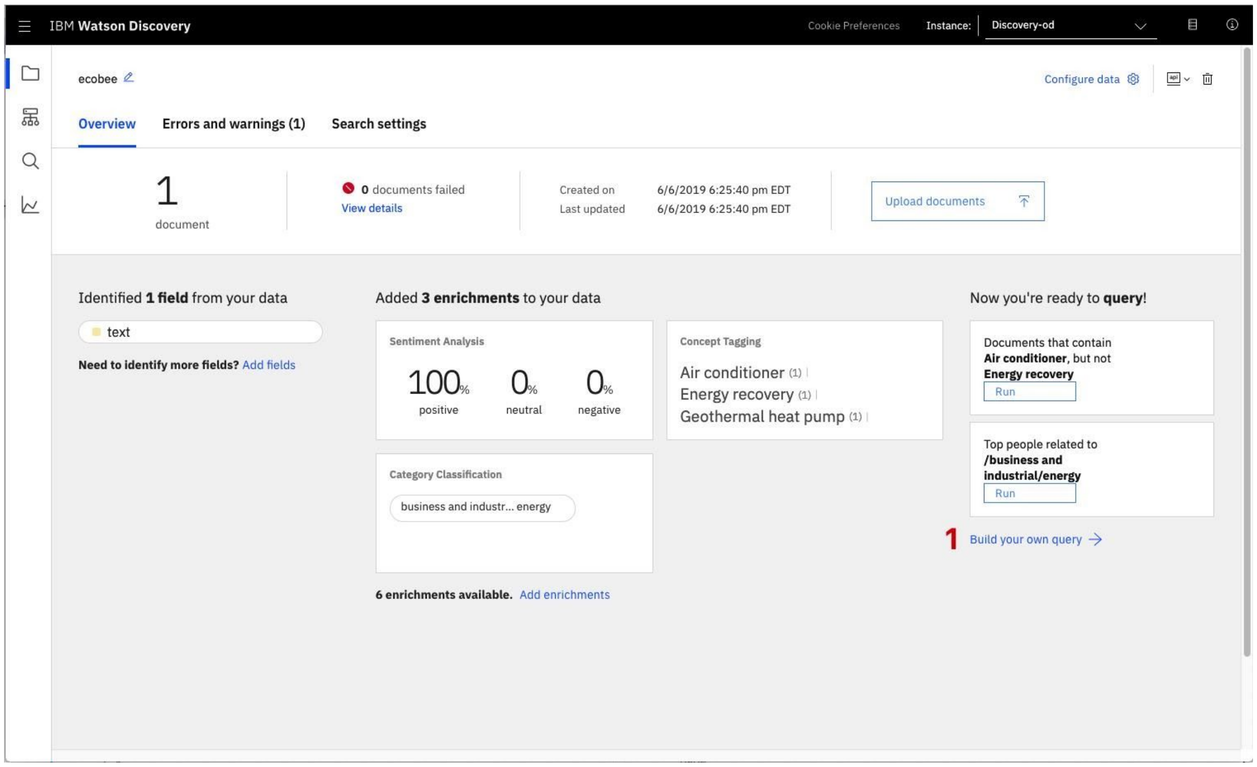
- Watson Discovery
- Watson Assistant
- IBM cloud function
- Node Red

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, let's do some simple queries on the data so that we can compare it to results found after applying SDU.



1 document

0 documents failed
View details

Created on 6/6/2019 6:25:40 pm EDT
Last updated 6/6/2019 6:25:40 pm EDT

Identified **1 field** from your data

text

Need to identify more fields? Add fields

Added **3 enrichments** to your data

Sentiment Analysis: 100% positive, 0% neutral, 0% negative

Concept Tagging: Air conditioner (1), Energy recovery (1), Geothermal heat pump (1)

Category Classification: business and industr... energy

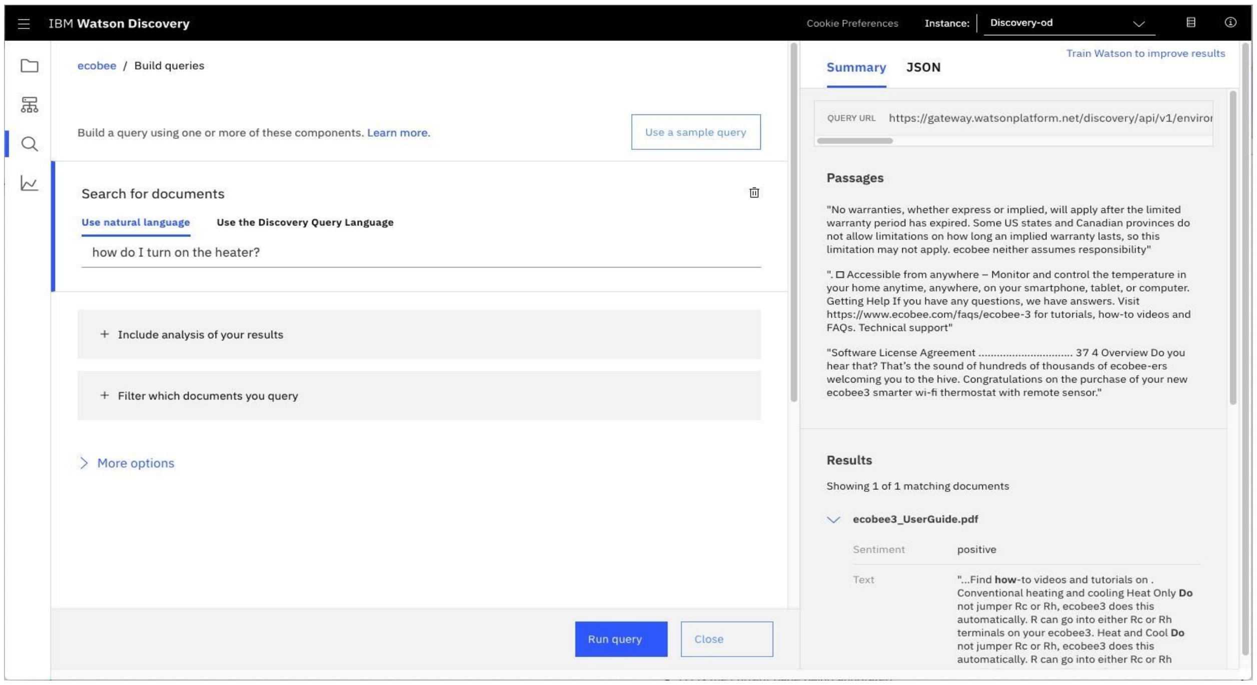
6 enrichments available. Add enrichments

Now you're ready to **query**!

Documents that contain **Air conditioner**, but not **Energy recovery**

Top people related to **/business and industrial/energy**

Click the Build your own query [1] button.



Build a query using one or more of these components. Learn more.

Search for documents

how do I turn on the heater?

QUERY URL https://gateway.watsonplatform.net/discovery/api/v1/enviror

Passages

"No warranties, whether express or implied, will apply after the limited warranty period has expired. Some US states and Canadian provinces do not allow limitations on how long an implied warranty lasts, so this limitation may not apply. ecobee neither assumes responsibility"

". ☐ Accessible from anywhere – Monitor and control the temperature in your home anytime, anywhere, on your smartphone, tablet, or computer. Getting Help If you have any questions, we have answers. Visit https://www.ecobee.com/faqs/ecobee-3 for tutorials, how-to videos and FAQs. Technical support"

"Software License Agreement .... 37 4 Overview Do you hear that? That's the sound of hundreds of thousands of ecobee-ers welcoming you to the hive. Congratulations on the purchase of your new ecobee3 smarter wi-fi thermostat with remote sensor."

Results

Showing 1 of 1 matching documents

ecobee3_UserGuide.pdf

Sentiment: positive

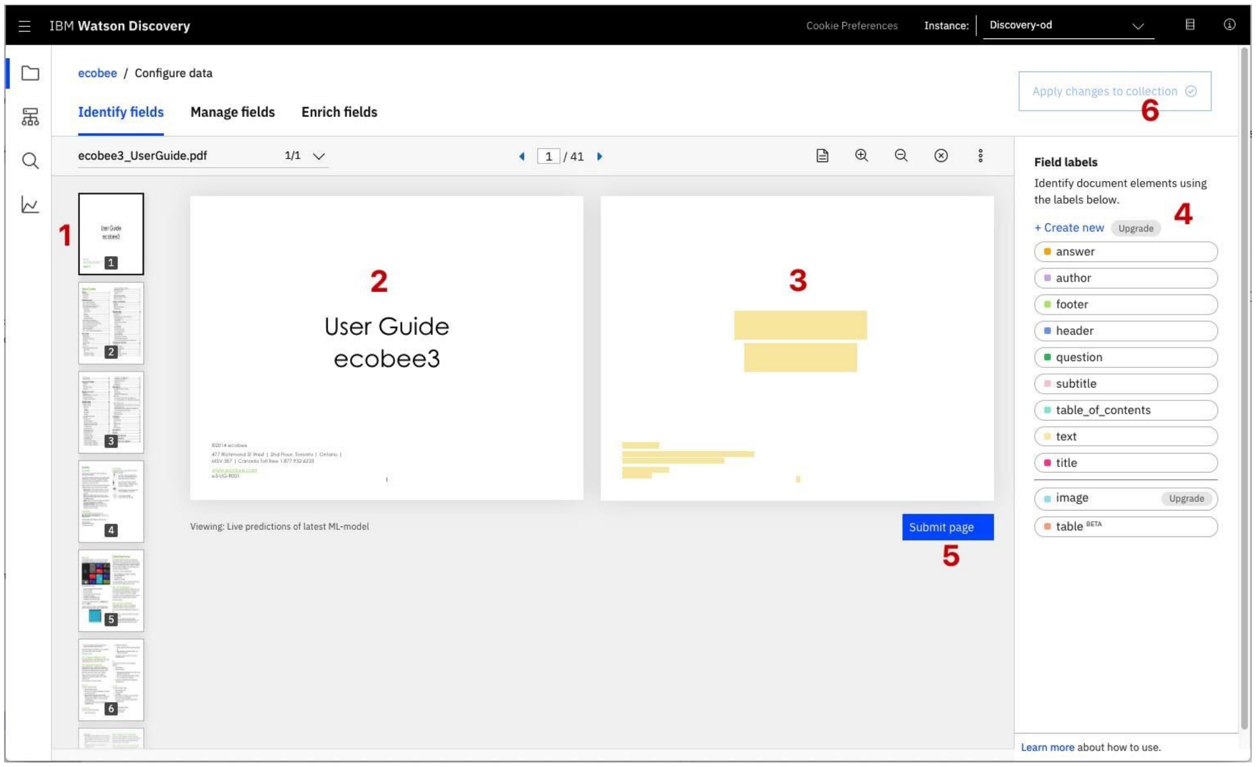
Text: "...Find **how**-to videos and tutorials on . Conventional heating and cooling Heat Only **Do** not jumper Rc or Rh, ecobee3 does this automatically. R can go into either Rc or Rh terminals on your ecobee3. Heat and Cool **Do** not jumper Rc or Rh, ecobee3 does this automatically. R can go into either Rc or Rh

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 subheaders (typed in light green text) as a subtitle

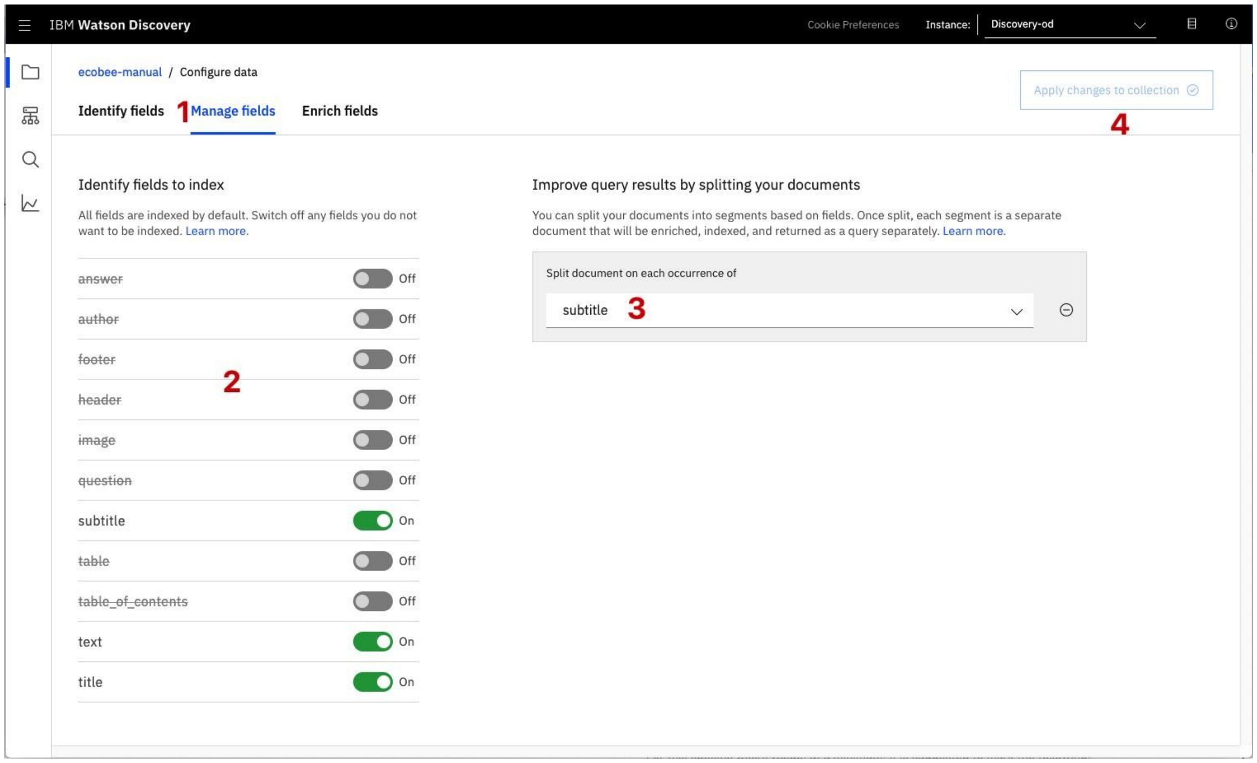
All page numbers as footers

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.



Identify fields to index

All fields are indexed by default. Switch off any fields you do not want to be indexed. Learn more.

~~answer~~	Off
~~author~~	Off
~~footer~~	Off
~~header~~	Off
~~image~~	Off
~~question~~	Off
subtitle	On
~~table~~	Off
~~table_of_contents~~	Off
text	On
title	On

Improve query results by splitting your documents

You can split your documents into segments based on fields. Once split, each segment is a separate document that will be enriched, indexed, and returned as a query separately. Learn more.

Split document on each occurrence of: subtitle

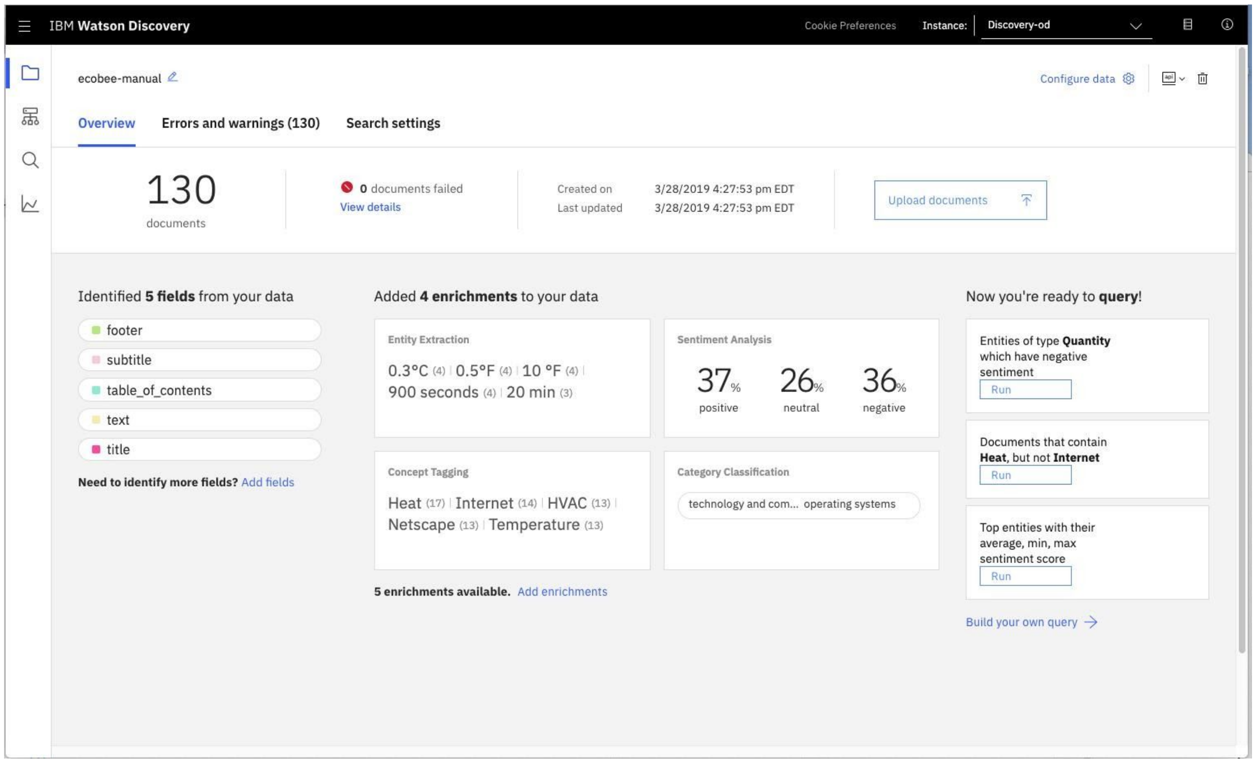
[2] Here is where you tell Discovery which fields to ignore. Using the on/off buttons, turn off all labels except subtitles and text.

[3] is telling Discovery to split the document apart, based on subtitles.

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:



130 documents

0 documents failed

Created on 3/28/2019 4:27:53 pm EDT
Last updated 3/28/2019 4:27:53 pm EDT

Identified **5 fields** from your data

- footer
- subtitle
- table_of_contents
- text
- title

Need to identify more fields? Add fields

Added **4 enrichments** to your data

Entity Extraction
0.3°C (4) | 0.5°F (4) | 10 °F (4) | 900 seconds (4) | 20 min (3)

Sentiment Analysis
37% positive 26% neutral 36% negative

Concept Tagging
Heat (17) | Internet (14) | HVAC (13) | Netscape (13) | Temperature (13)

Category Classification
technology and com... operating systems

5 enrichments available. Add enrichments

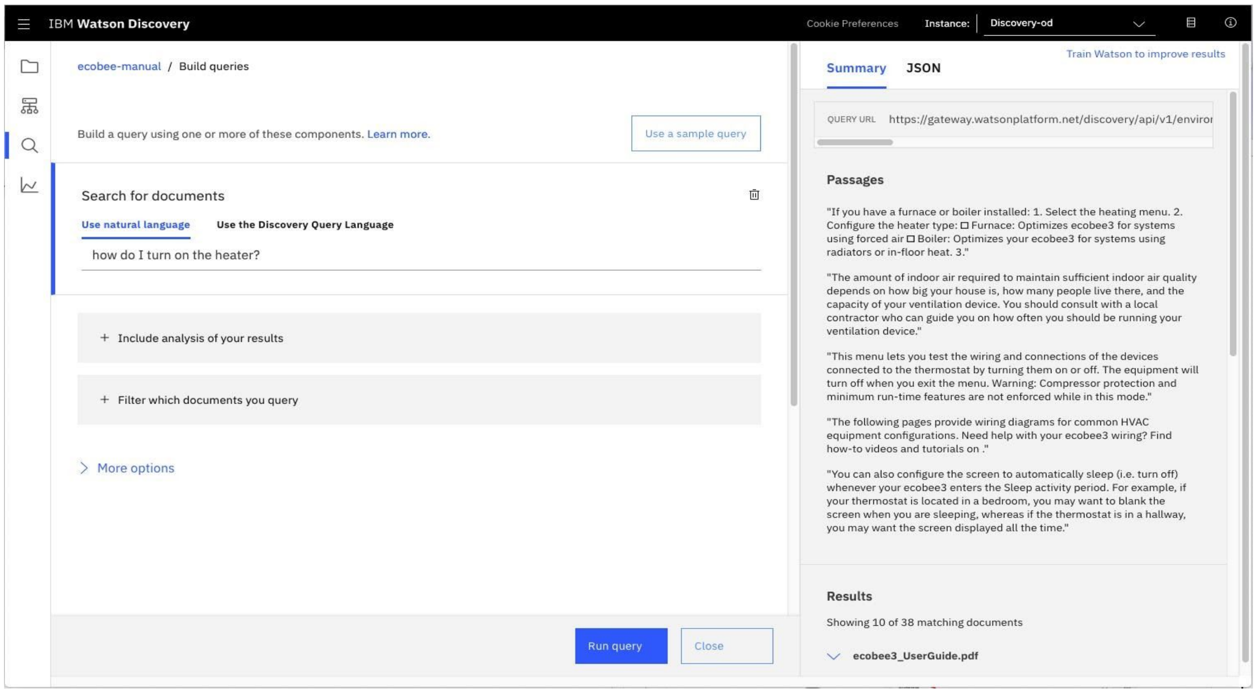
Now you're ready to **query**!

Entities of type **Quantity** which have negative sentiment

Documents that contain **Heat**, but not **Internet**

Top entities with their average, min, max sentiment score

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


IBM Watson Discovery
Cookie Preferences
Instance: Discovery-od
ecobee-manual / Build queries
Summary
JSON
Train Watson to improve results
Use a sample query
Use natural language
Use the Discovery Query Language
Include analysis of your results
Filter which documents you query
More options
Run query
Close

Build a query using one or more of these components. Learn more.

Search for documents

how do I turn on the heater?

QUERY URL https://gateway.watsonplatform.net/discovery/api/v1/enviror

Passages

"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."

"The amount of indoor air required to maintain sufficient indoor air quality depends on how big your house is, how many people live there, and the capacity of your ventilation device. You should consult with a local contractor who can guide you on how often you should be running your ventilation device."

"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."

"The following pages provide wiring diagrams for common HVAC equipment configurations. Need help with your ecobee3 wiring? Find how-to videos and tutorials on ."

"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."

Results

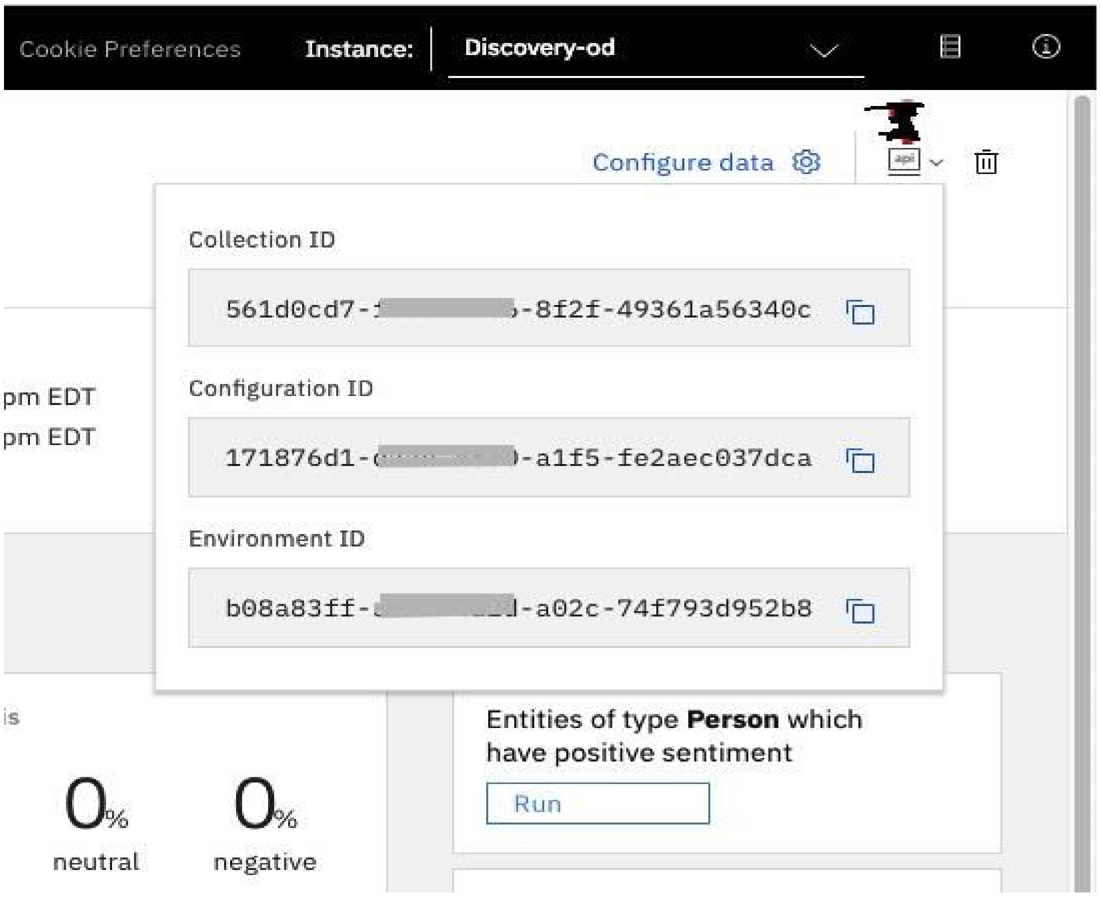
Showing 10 of 38 matching documents

ecobee3_UserGuide.pdf

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:



Collection ID

`561d0cd7-[redacted]-8f2f-49361a56340c`

Configuration ID

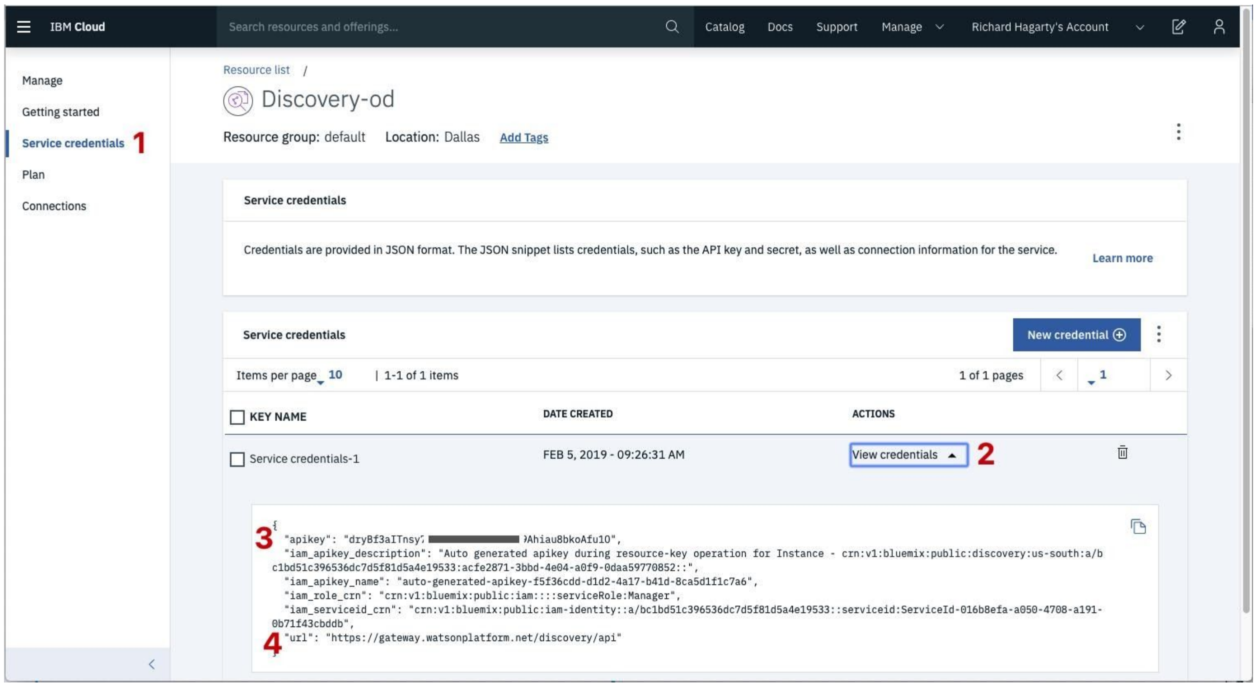
`171876d1-[redacted]-a1f5-fe2aec037dca`

Environment ID

`b08a83ff-[redacted]-a02c-74f793d952b8`

Entities of type **Person** which have positive sentiment

For credentials, return to the main panel of your Discovery service, and click the Service credentials [1] tab:



Resource group: default Location: Dallas Add Tags

Service credentials

Credentials are provided in JSON format. The JSON snippet lists credentials, such as the API key and secret, as well as connection information for the service.

Service credentials

Items per page 10 | 1-1 of 1 items 1 of 1 pages 1

KEY NAME	DATE CREATED	ACTIONS
Service credentials-1	FEB 5, 2019 - 09:26:31 AM	View credentials

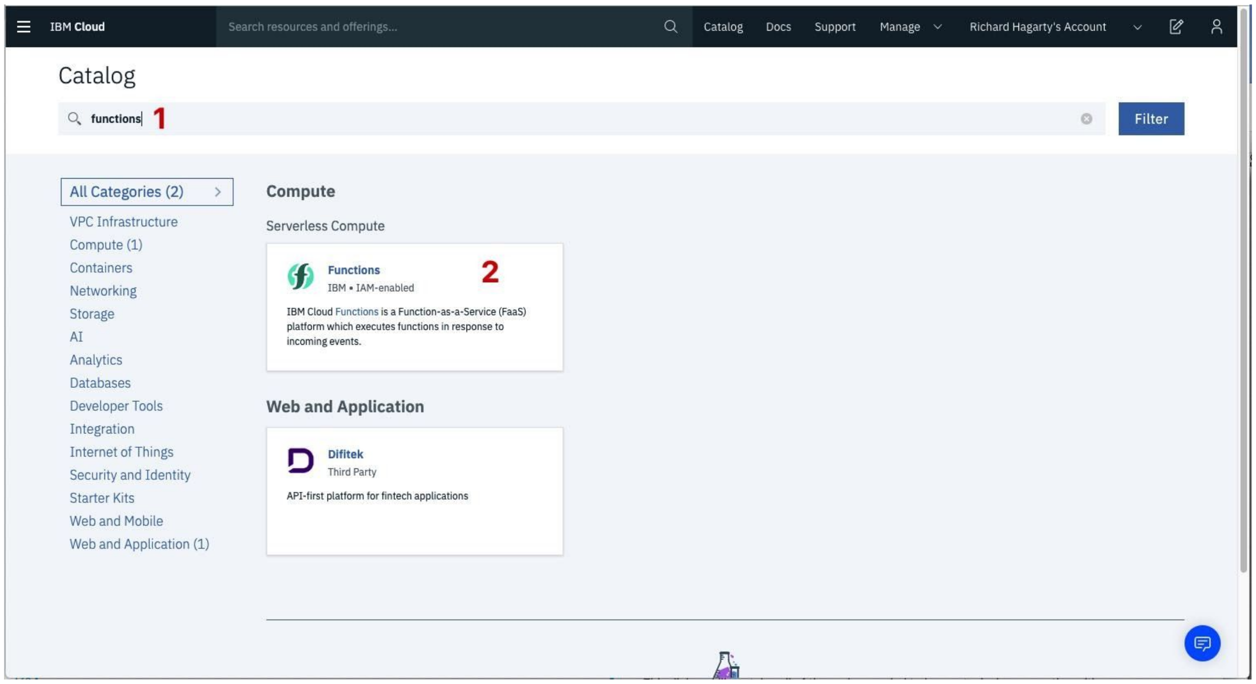
```
{
  "apikey": "dryBf3aITnsy7[redacted]?Ahiau8bkoAfu10",
  "iam_apikey_description": "Auto generated apikey during resource-key operation for Instance - crn:v1:bluemix:public:discovery:us-south:a/b
c1bd51c396536dc7d5f81d5a4e19533:acfe2871-3bbd-4e04-a0f9-0daa59770852::",
  "iam_apikey_name": "auto-generated-apikey-f5f36cdd-d1d2-4a17-b41d-8ca5d1f1c7a6",
  "iam_role_crn": "crn:v1:bluemix:public:iam::::serviceRole:Manager",
  "iam_serviceid_crn": "crn:v1:bluemix:public:iam-identity::a/bc1bd51c396536dc7d5f81d5a4e19533::serviceid:ServiceId-016b8efa-a050-4708-a191-
0b71f43cbddb",
  "url": "https://gateway.watsonplatform.net/discovery/api"
}
```

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

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]:



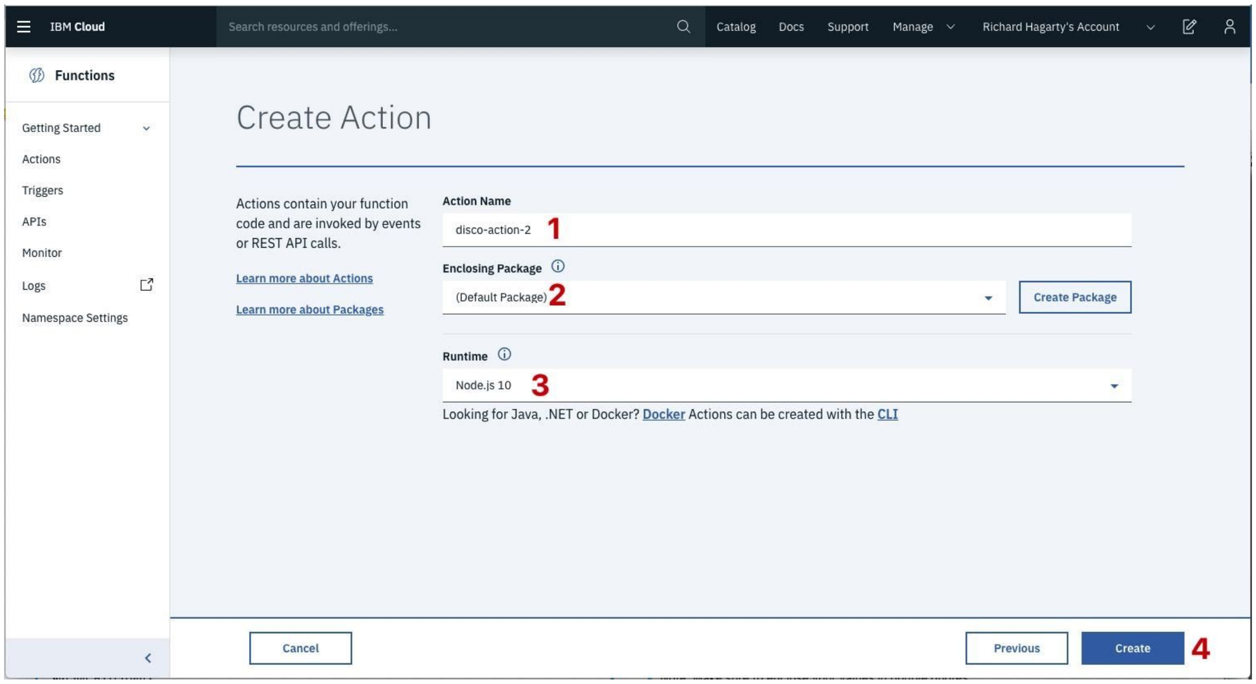
IBM Cloud Functions is a Function-as-a-Service (FaaS) platform which executes functions in response to incoming events.

API-first platform for fintech applications

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.



Create Action

Actions contain your function code and are invoked by events or REST API calls.

Learn more about Actions

Learn more about Packages

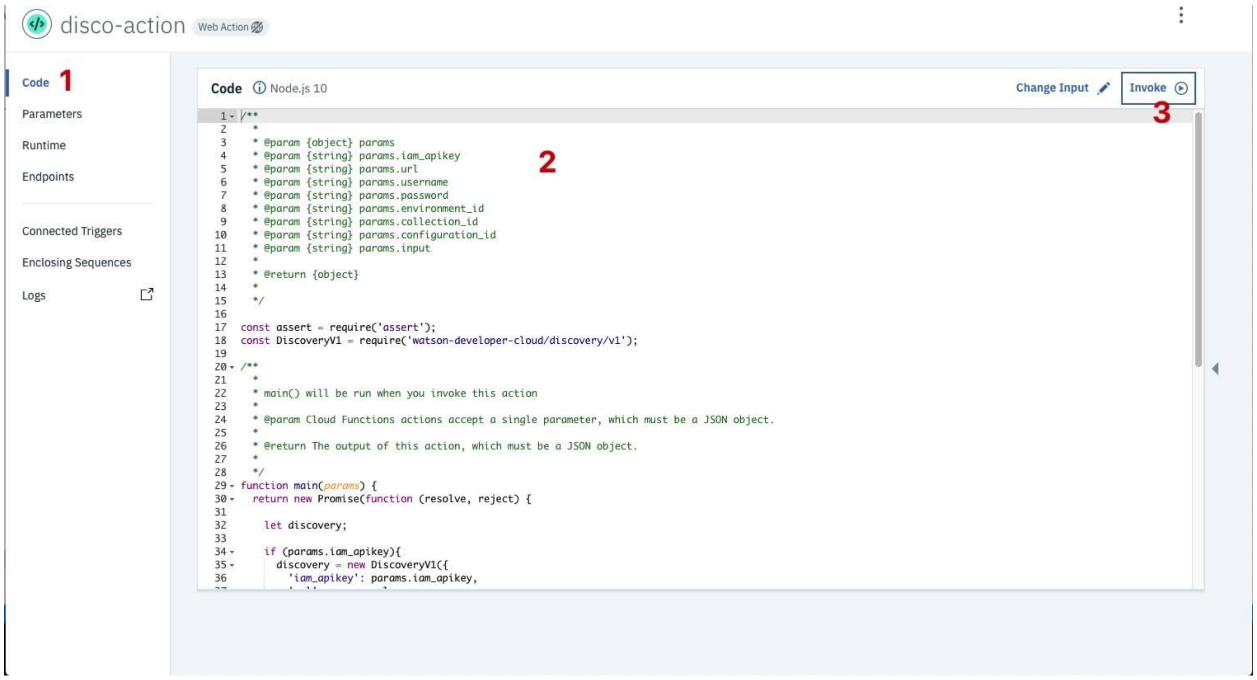
Action Name: disco-action-2 1

Enclosing Package: (Default Package) 2

Runtime: Node.js 10 3

Looking for Java, .NET or Docker? Docker Actions can be created with the CLI

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



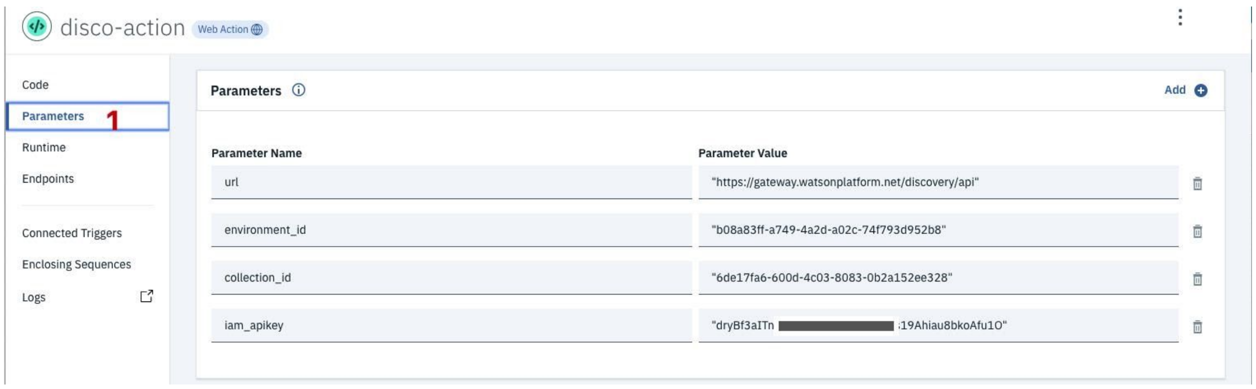
Code Node.js 10

```
/**
  *
  * @param {object} params
  * @param {string} params.iam_apikey
  * @param {string} params.url
  * @param {string} params.username
  * @param {string} params.password
  * @param {string} params.environment_id
  * @param {string} params.collection_id
  * @param {string} params.configuration_id
  * @param {string} params.input
  *
  * @return {object}
  *
  */

const assert = require('assert');
const DiscoveryV1 = require('watson-developer-cloud/discovery/v1');

/**
  *
  * main() will be run when you invoke this action
  *
  * @param Cloud Functions actions accept a single parameter, which must be a JSON object.
  *
  * @return The output of this action, which must be a JSON object.
  *
  */
function main(params) {
  return new Promise(function (resolve, reject) {

    let discovery;

    if (params.iam_apikey){
      discovery = new DiscoveryV1({
        'iam_apikey': params.iam_apikey,
```

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]:



Parameter Name	Parameter Value
url	"https://gateway.watsonplatform.net/discovery/api"
environment_id	"b08a83ff-a749-4a2d-a02c-74f793d952b8"
collection_id	"6de17fa6-600d-4c03-8083-0b2a152ee328"
iam_apikey	"dryBf3aITn [redacted] :19Ahiau8bkoAfu1O"

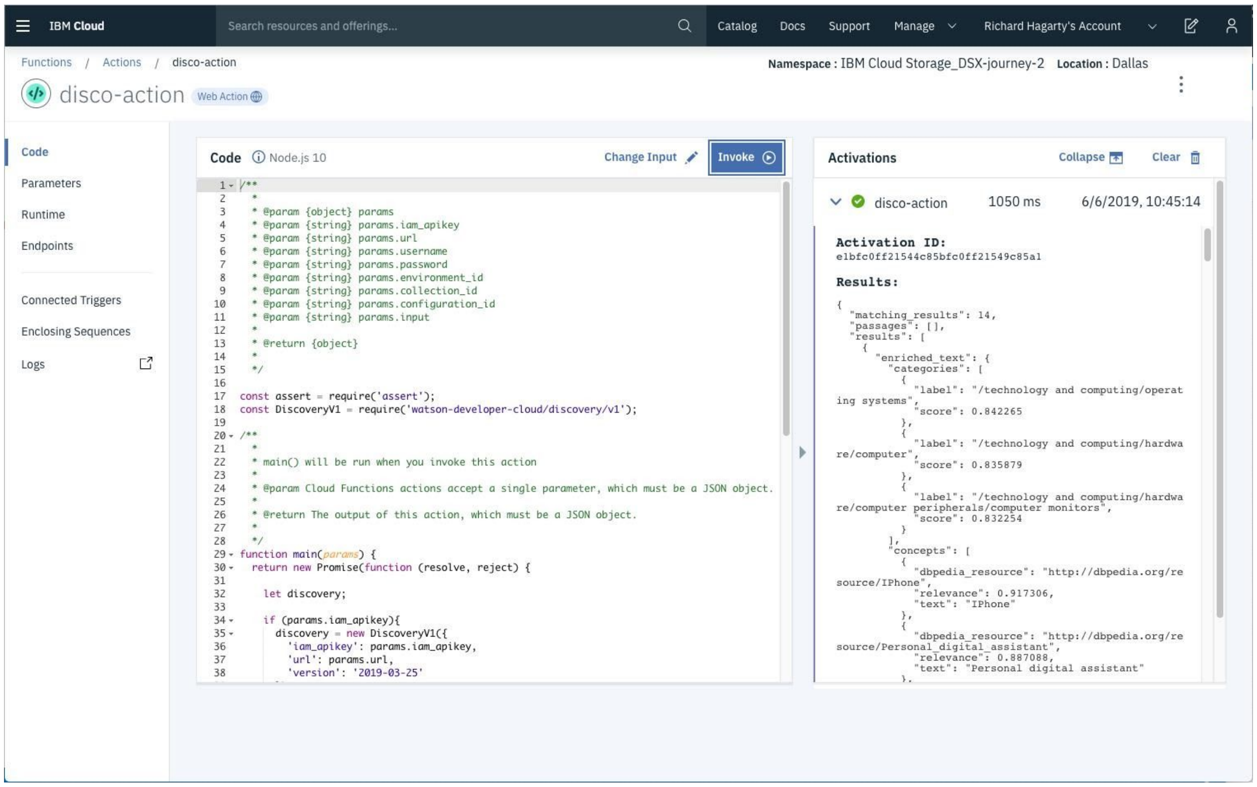
Add the following keys:

➢ url
➢ environment_id
➢ collection_id
➢ iam_apikey

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

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:



IBM Cloud · Search resources and offerings... · Catalog · Docs · Support · Manage · Richard Hagarty's Account

Functions / Actions / disco-action

Namespace : IBM Cloud Storage_DSX-journey-2 **Location :** Dallas

disco-action Web Action

Code · Parameters · Runtime · Endpoints · Connected Triggers · Enclosing Sequences · Logs

Code Node.js 10 · Change Input · Invoke

```
/**
  *
  * @param {object} params
  * @param {string} params.iam_apikey
  * @param {string} params.url
  * @param {string} params.username
  * @param {string} params.password
  * @param {string} params.environment_id
  * @param {string} params.collection_id
  * @param {string} params.configuration_id
  * @param {string} params.input
  *
  * @return {object}
  *
  */

const assert = require('assert');
const DiscoveryV1 = require('watson-developer-cloud/discovery/v1');

/**
  *
  * main() will be run when you invoke this action
  *
  * @param Cloud Functions actions accept a single parameter, which must be a JSON object.
  *
  * @return The output of this action, which must be a JSON object.
  *
  */
function main(params) {
  return new Promise(function (resolve, reject) {

    let discovery;

    if (params.iam_apikey){
      discovery = new DiscoveryV1({
        'iam_apikey': params.iam_apikey,
        'url': params.url,
        'version': '2019-03-25'
```

Activations · Collapse · Clear

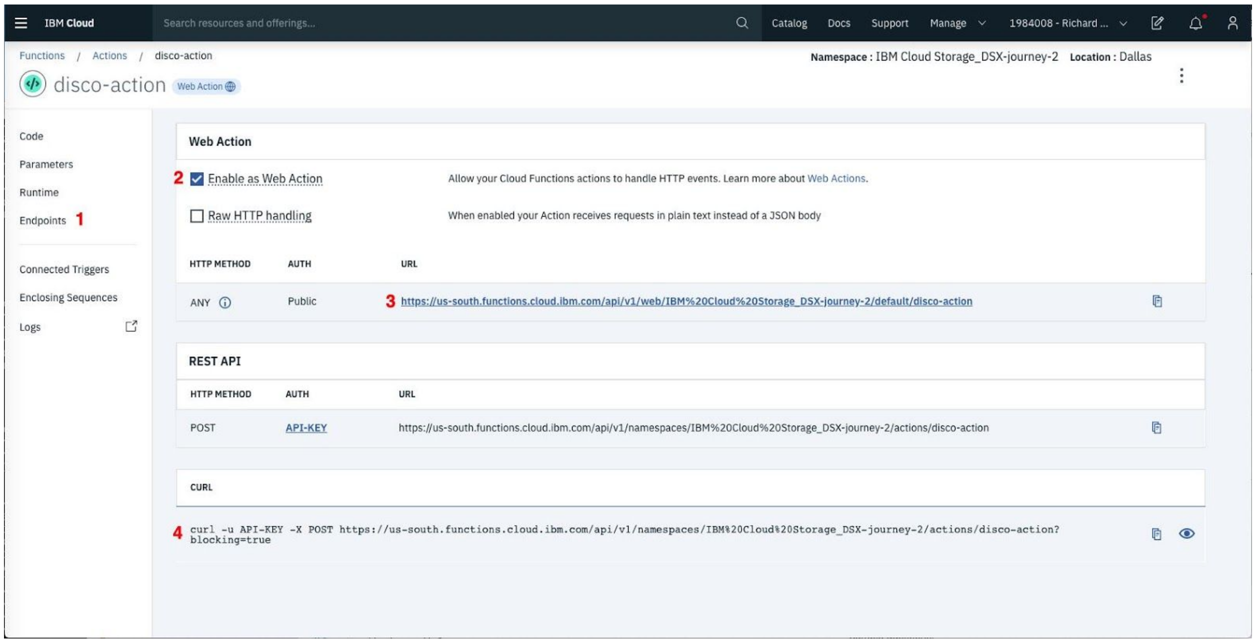
disco-action 1050 ms 6/6/2019, 10:45:14

Activation ID:
e1bfc0ff21544c85bfc0ff21549c85a1

Results:

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

Next, go to the Endpoints panel [1]:



IBM Cloud · Search resources and offerings... · Catalog · Docs · Support · Manage · 1984008 - Richard ...

Functions / Actions / disco-action

Namespace : IBM Cloud Storage_DSX-journey-2 **Location :** Dallas

disco-action Web Action

Code · Parameters · Runtime · Endpoints 1 · Connected Triggers · Enclosing Sequences · Logs

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.

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

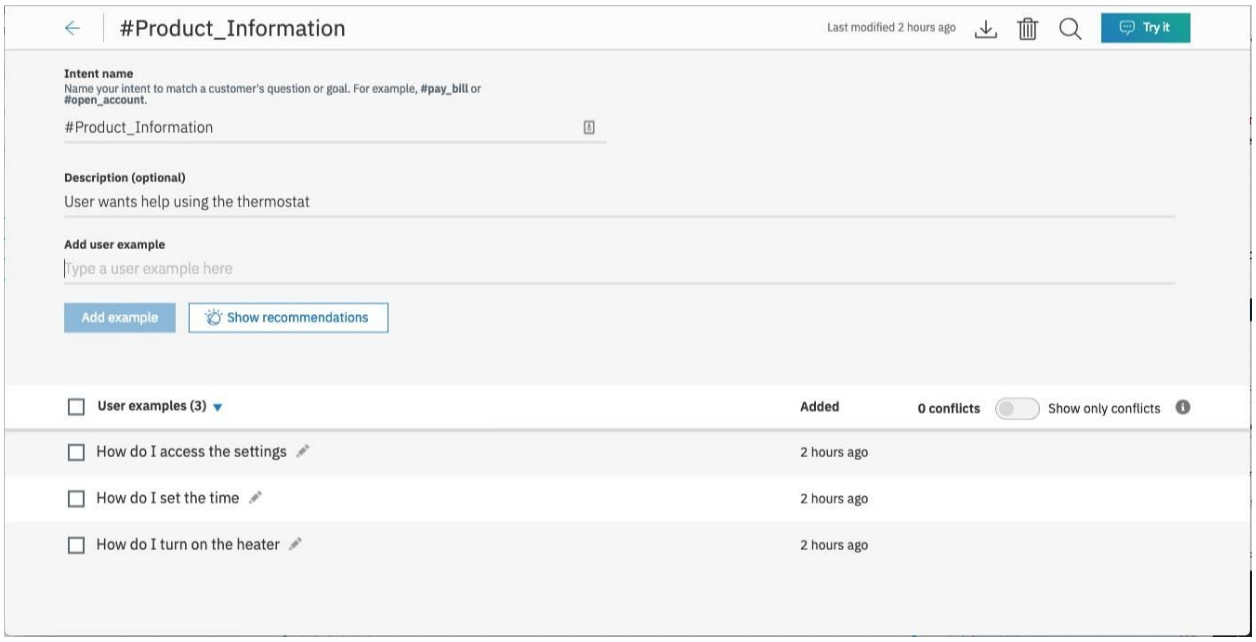
involving outside resources, so we will need to add this.

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.



Intent name
Name your intent to match a customer's question or goal. For example, **#pay_bill** or **#open_account**.

#Product_Information

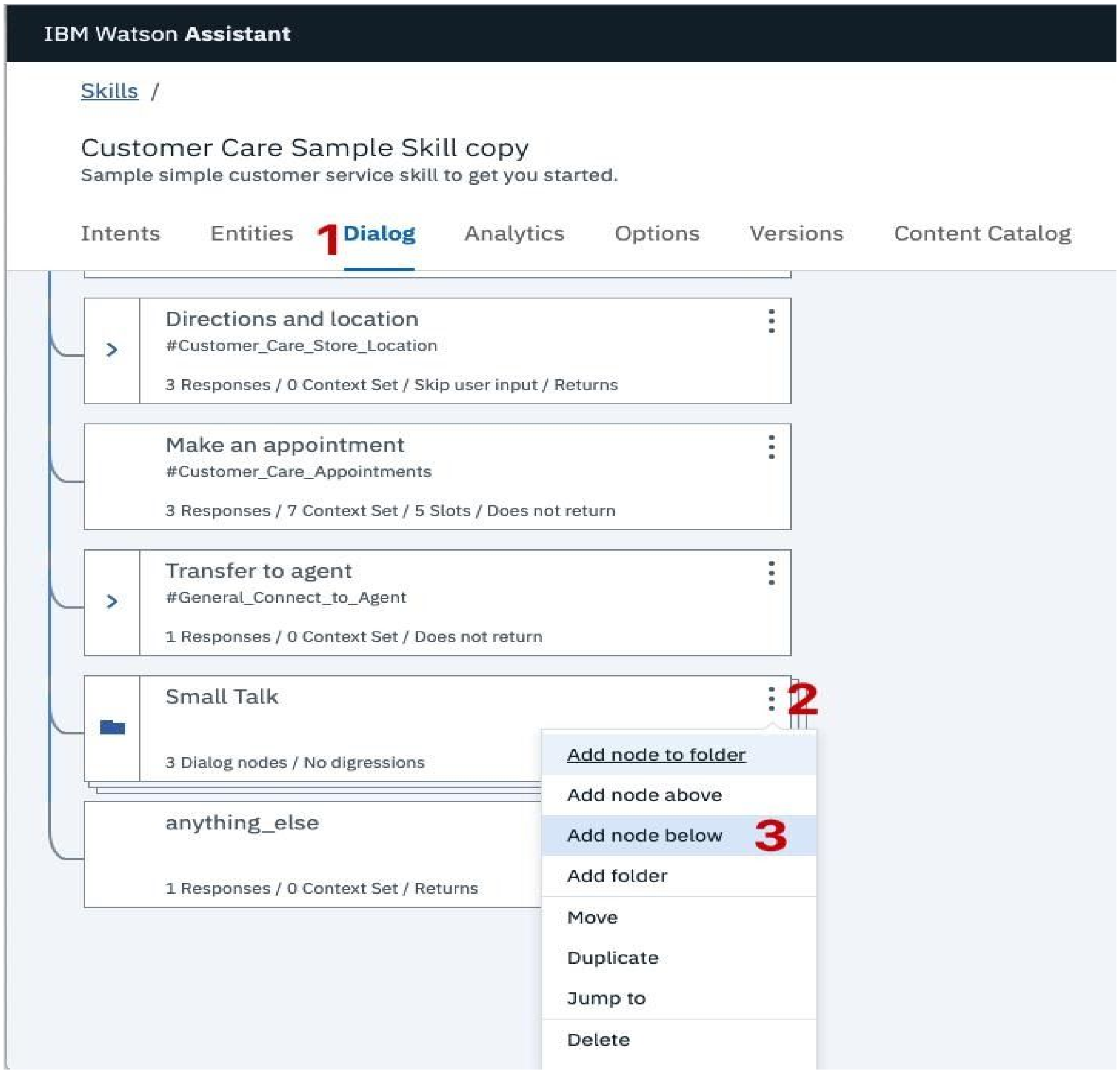
Description (optional)
User wants help using the thermostat

Add user example
Type a user example here

User examples (3)	Added
How do I access the settings	2 hours ago
How do I set the time	2 hours ago
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.


IBM Watson Assistant

Skills /

Customer Care Sample Skill copy
Sample simple customer service skill to get you started.

Directions and location
#Customer_Care_Store_Location
3 Responses / 0 Context Set / Skip user input / Returns

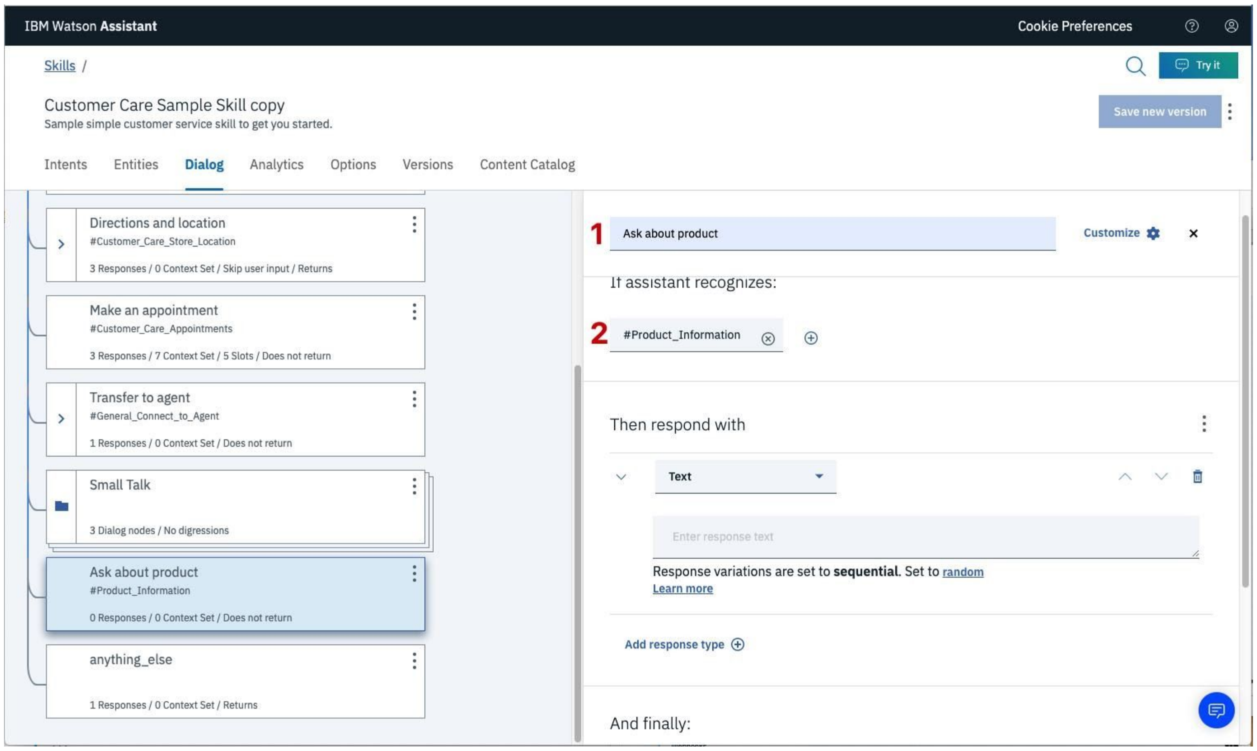
Make an appointment
#Customer_Care_Appointments
3 Responses / 7 Context Set / 5 Slots / Does not return

Transfer to agent
#General_Connect_to_Agent
1 Responses / 0 Context Set / Does not return

Small Talk
3 Dialog nodes / No digressions

anything_else
1 Responses / 0 Context Set / Returns

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



Customer Care Sample Skill copy
Sample simple customer service skill to get you started.

Directions and location
#Customer_Care_Store_Location
3 Responses / 0 Context Set / Skip user input / Returns

Make an appointment
#Customer_Care_Appointments
3 Responses / 7 Context Set / 5 Slots / Does not return

Transfer to agent
#General_Connect_to_Agent
1 Responses / 0 Context Set / Does not return

Small Talk
3 Dialog nodes / No digressions

Ask about product
#Product_Information
0 Responses / 0 Context Set / Does not return

anything_else
1 Responses / 0 Context Set / Returns

1 Ask about product

2 #Product_Information

Enter response text

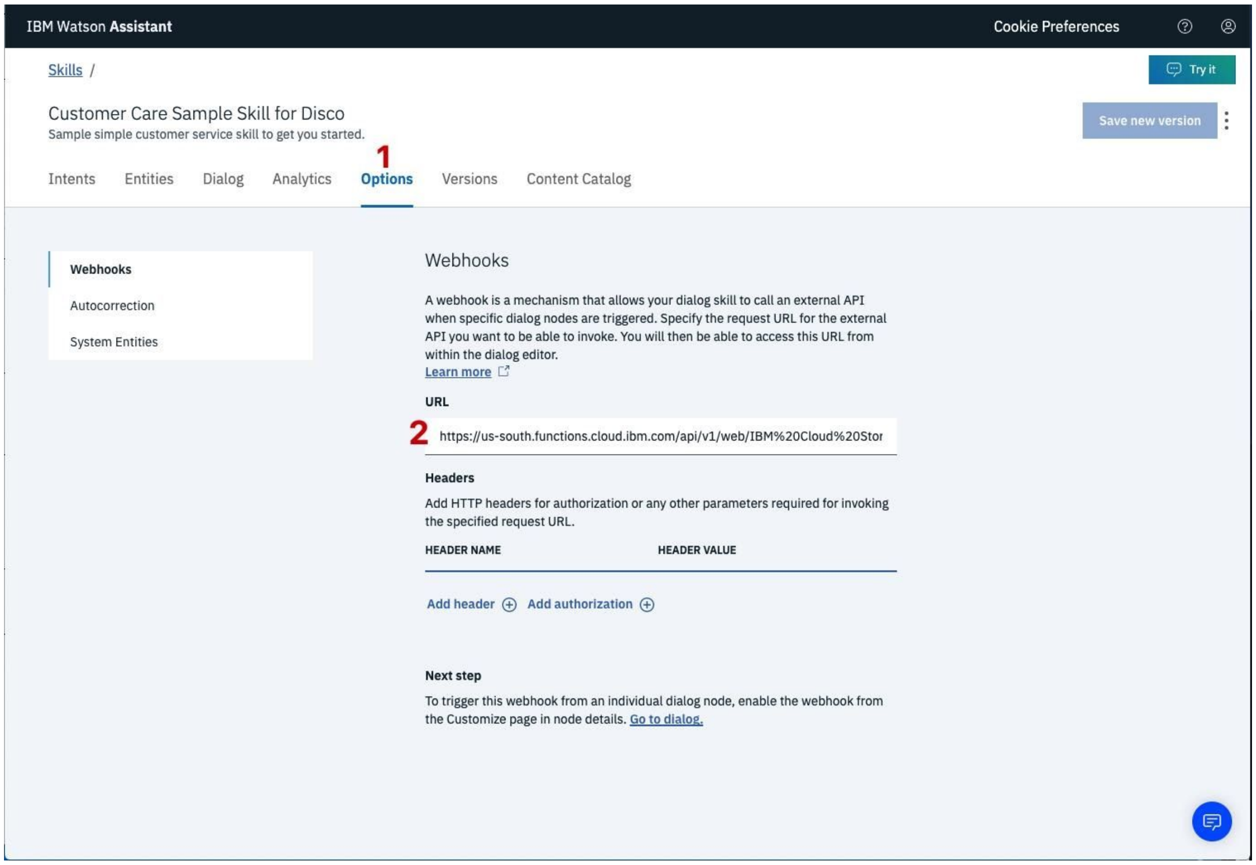
Response variations are set to **sequential**. Set to random
Learn more

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]:



Customer Care Sample Skill for Disco

Sample simple customer service skill to get you started.

Webhooks

A webhook is a mechanism that allows your dialog skill to call an external API when specific dialog nodes are triggered. Specify the request URL for the external API you want to be able to invoke. You will then be able to access this URL from within the dialog editor.
Learn more

URL

https://us-south.functions.cloud.ibm.com/api/v1/web/IBM%20Cloud%20Stor

Headers

Add HTTP headers for authorization or any other parameters required for invoking the specified request URL.

HEADER NAME	HEADER VALUE

Next step

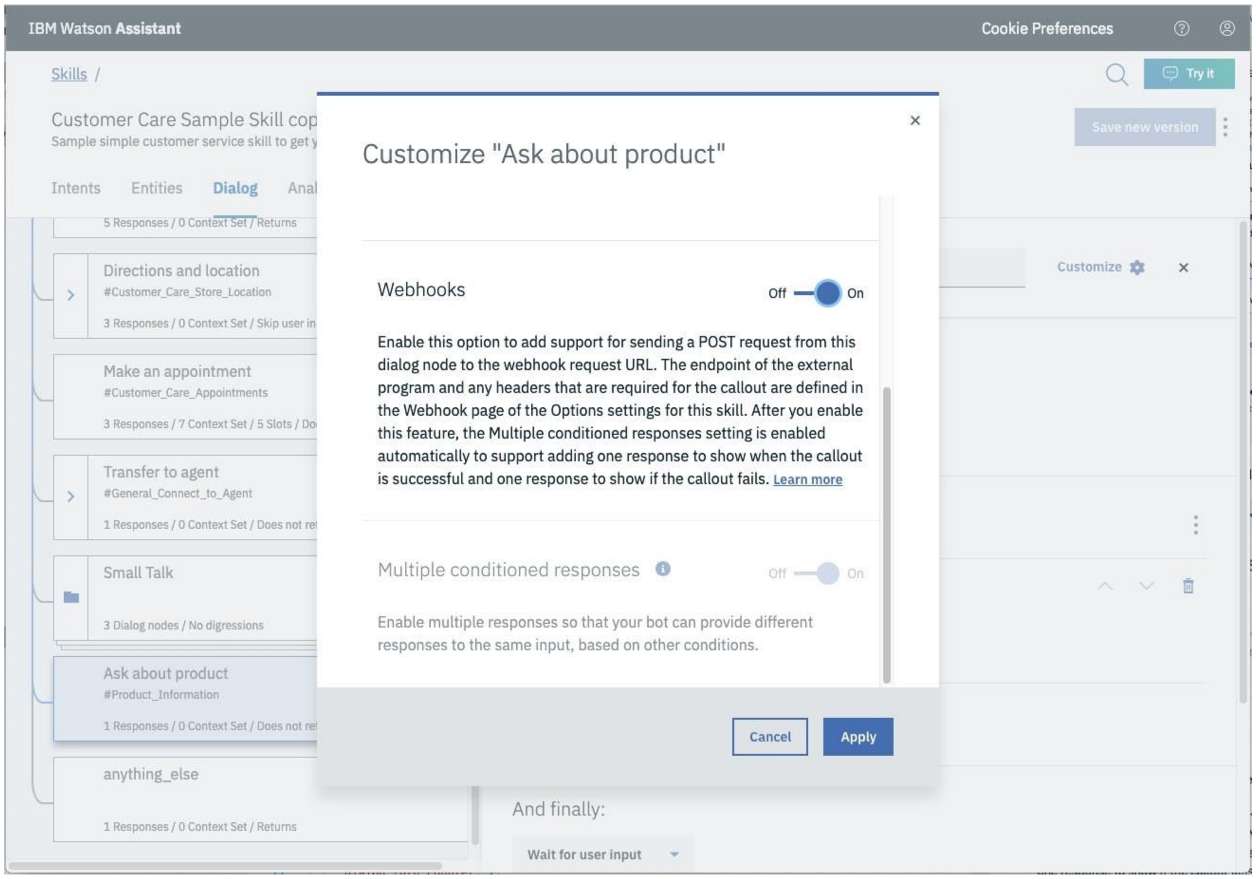
To trigger this webhook from an individual dialog node, enable the webhook from the Customize page in node details. Go to dialog.

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:


IBM Watson Assistant

Customize "Ask about product"

Webhooks Off [✓ On]

Enable this option to add support for sending a POST request from this dialog node to the webhook request URL. The endpoint of the external program and any headers that are required for the callout are defined in the Webhook page of the Options settings for this skill. After you enable this feature, the Multiple conditioned responses setting is enabled automatically to support adding one response to show when the callout is successful and one response to show if the callout fails. Learn more

Multiple conditioned responses Off On

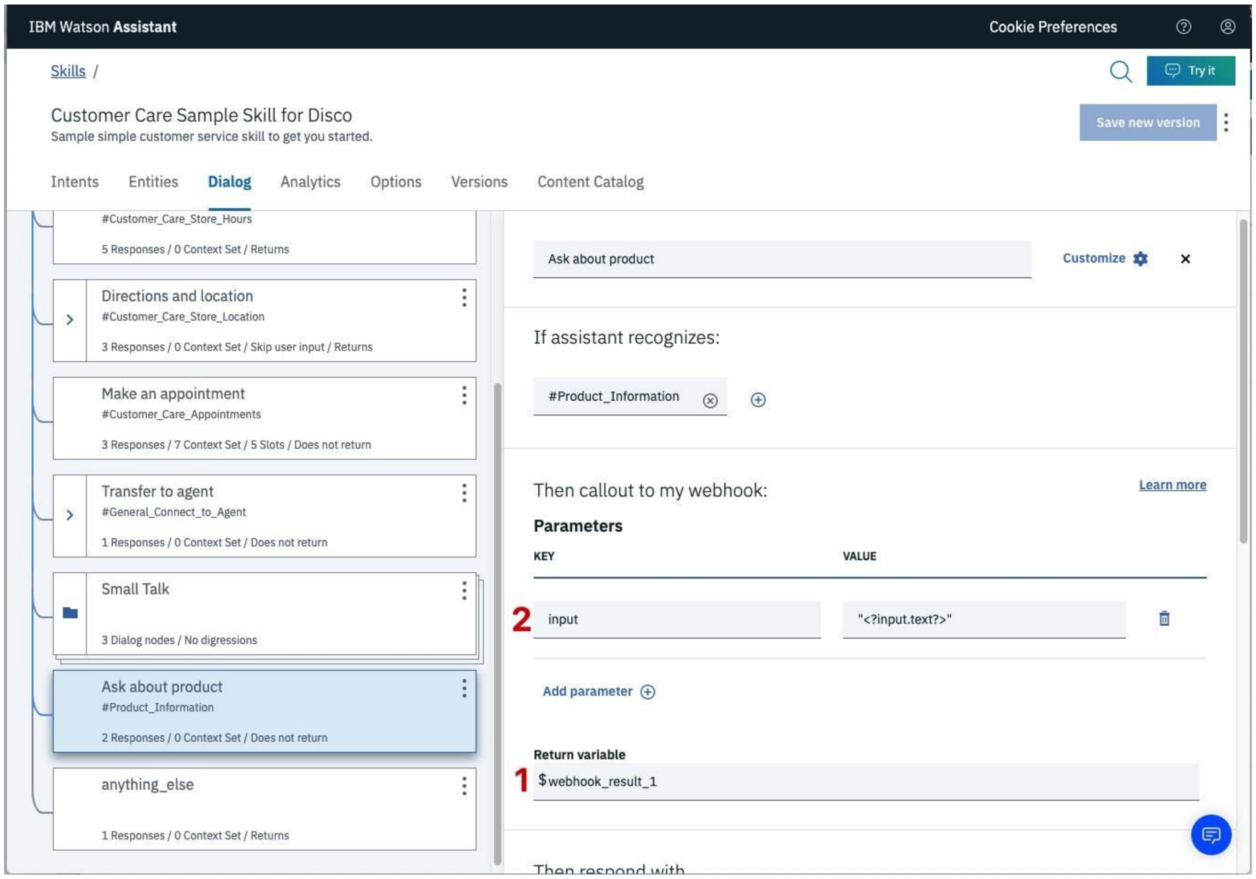
Enable multiple responses so that your bot can provide different responses to the same input, based on other conditions.

Cookie Preferences
Skills /
Try it
Customer Care Sample Skill cop
Sample simple customer service skill to get y
Save new version
Intents
Entities
Dialog
Anal
5 Responses / 0 Context Set / Returns
Directions and location
#Customer_Care_Store_Location
3 Responses / 0 Context Set / Skip user in
Make an appointment
#Customer_Care_Appointments
3 Responses / 7 Context Set / 5 Slots / Do
Transfer to agent
#General_Connect_to_Agent
1 Responses / 0 Context Set / Does not re
Small Talk
3 Dialog nodes / No digressions
Ask about product
#Product_Information
1 Responses / 0 Context Set / Does not re
anything_else
1 Responses / 0 Context Set / Returns
Customize
Cancel
Apply
And finally:
Wait for user input

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.


IBM Watson **Assistant** Cookie Preferences

Skills /

Customer Care Sample Skill for Disco
Sample simple customer service skill to get you started.

Try it | Save new version

Intents | Entities | **Dialog** | Analytics | Options | Versions | Content Catalog

#Customer_Care_Store_Hours
5 Responses / 0 Context Set / Returns

Directions and location
#Customer_Care_Store_Location
3 Responses / 0 Context Set / Skip user input / Returns

Make an appointment
#Customer_Care_Appointments
3 Responses / 7 Context Set / 5 Slots / Does not return

Transfer to agent
#General_Connect_to_Agent
1 Responses / 0 Context Set / Does not return

Small Talk
3 Dialog nodes / No digressions

Ask about product
#Product_Information
2 Responses / 0 Context Set / Does not return

anything_else
1 Responses / 0 Context Set / Returns

Ask about product | Customize

If assistant recognizes:

#Product_Information

Then callout to my webhook: | Learn more

Parameters

KEY	VALUE
2 input	"<?input.text?>"

Add parameter

Return variable

1 $webhook_result_1

Then respond with

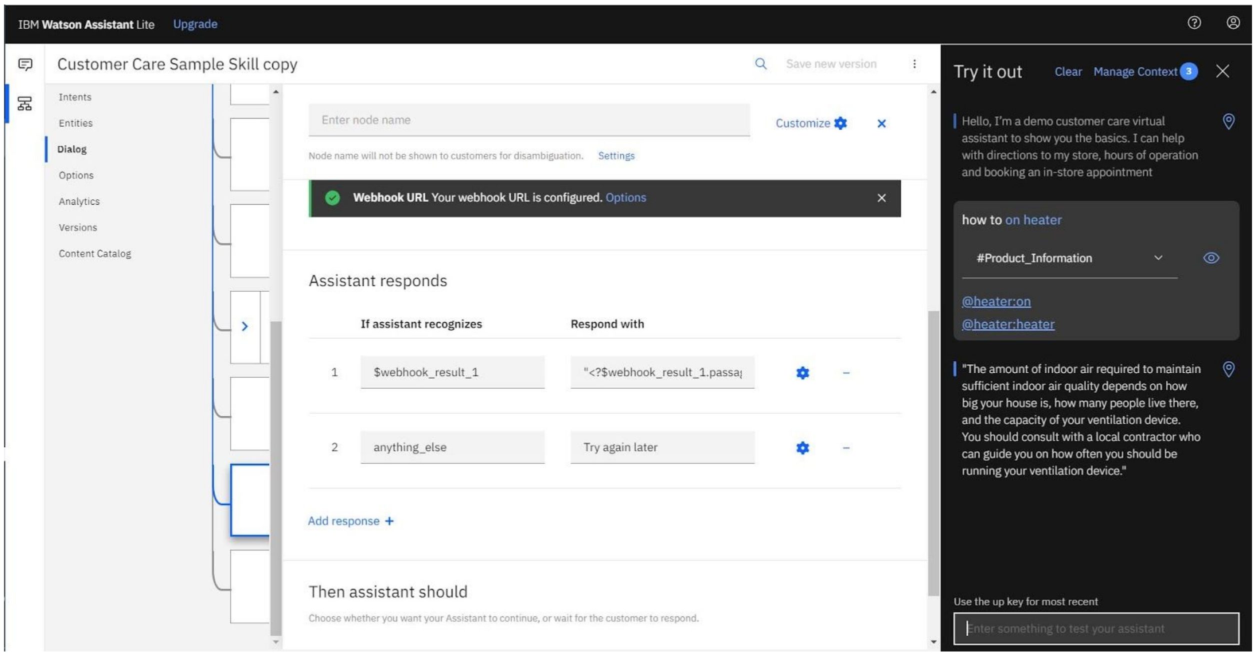
You will also need to pass in the users question via the parameter input [2]. The key needs to be set to the value:

"<?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:

Add Add “<?webhook_result_1.passages[0].passage_text?>” in response within the Assistant responds block as shown below.


IBM Watson Assistant Lite Upgrade

Customer Care Sample Skill copy

Intents
Entities
Dialog
Options
Analytics
Versions
Content Catalog

Enter node name

Node name will not be shown to customers for disambiguation. Settings

Webhook URL Your webhook URL is configured. Options

Assistant responds

	If assistant recognizes	Respond with
1	$webhook_result_1	"<?$webhook_result_1.passa
2	anything_else	Try again later

Add response +

Then assistant should

Choose whether you want your Assistant to continue, or wait for the customer to respond.

Try it out

Hello, I'm a demo customer care virtual assistant to show you the basics. I can help with directions to my store, hours of operation and booking an in-store appointment

how to on heater

#Product_Information

@heater:on
@heater:heater

"The amount of indoor air required to maintain sufficient indoor air quality depends on how big your house is, how many people live there, and the capacity of your ventilation device. You should consult with a local contractor who can guide you on how often you should be running your ventilation device."

Use the up key for most recent

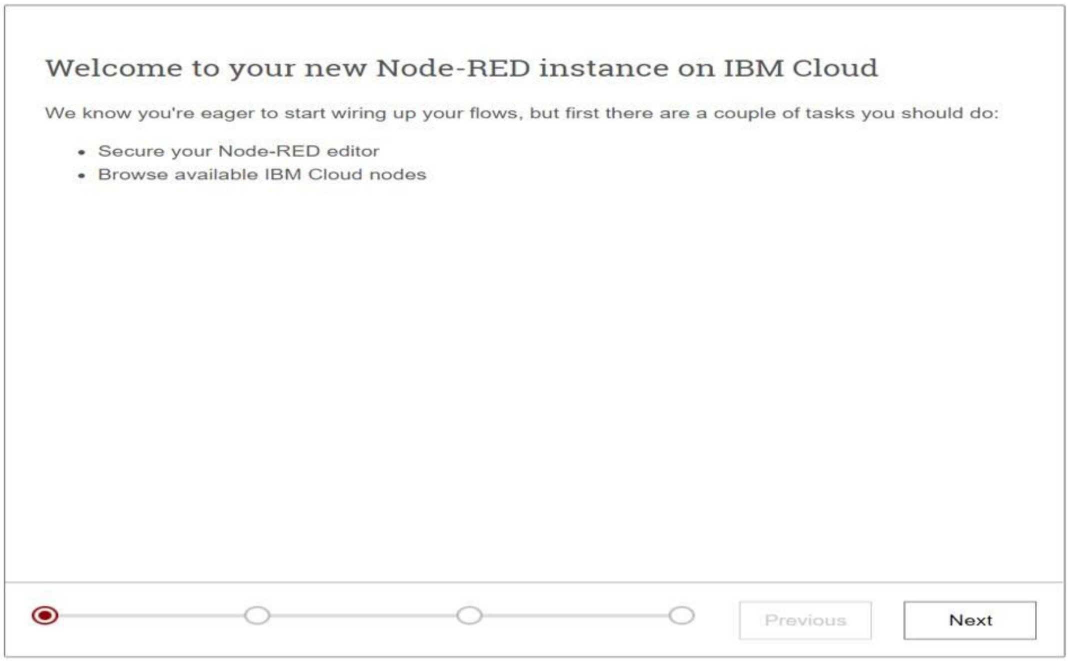
Enter something to test your assistant

Creation of Node-RED in IBM cloud

- ➢ Step-1: Login to IBM and go to the catalog
- ➢ Step-2: Search for node-red and select “Node-RED Starter “ Service
- ➢ Step-3: Enter the Unique name and click on create a button

  Note: Your Node-red service is starting

- ➢ Step – 5: We have to configure Node red for the first time. Click on next to continue

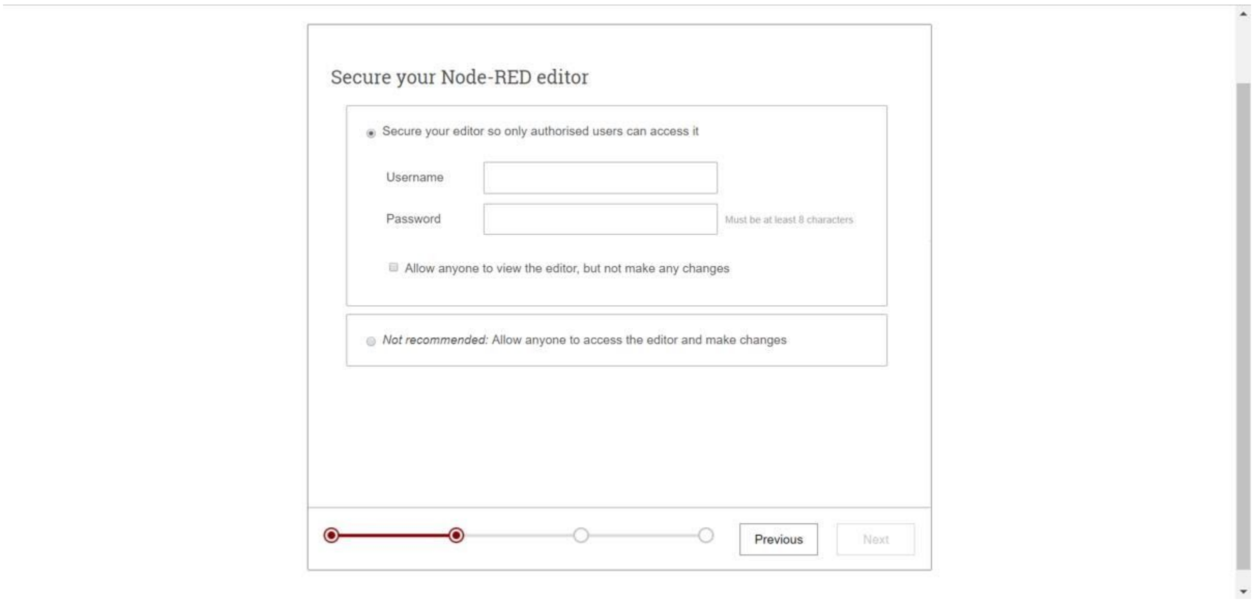


Welcome to your new Node-RED instance on IBM Cloud

We know you're eager to start wiring up your flows, but first there are a couple of tasks you should do:

- Secure your Node-RED editor
- Browse available IBM Cloud nodes

➢ Step – 6: Secure your node red editor by giving a username and password and click on Next



Secure your Node-RED editor

(●) Secure your editor so only authorised users can access it

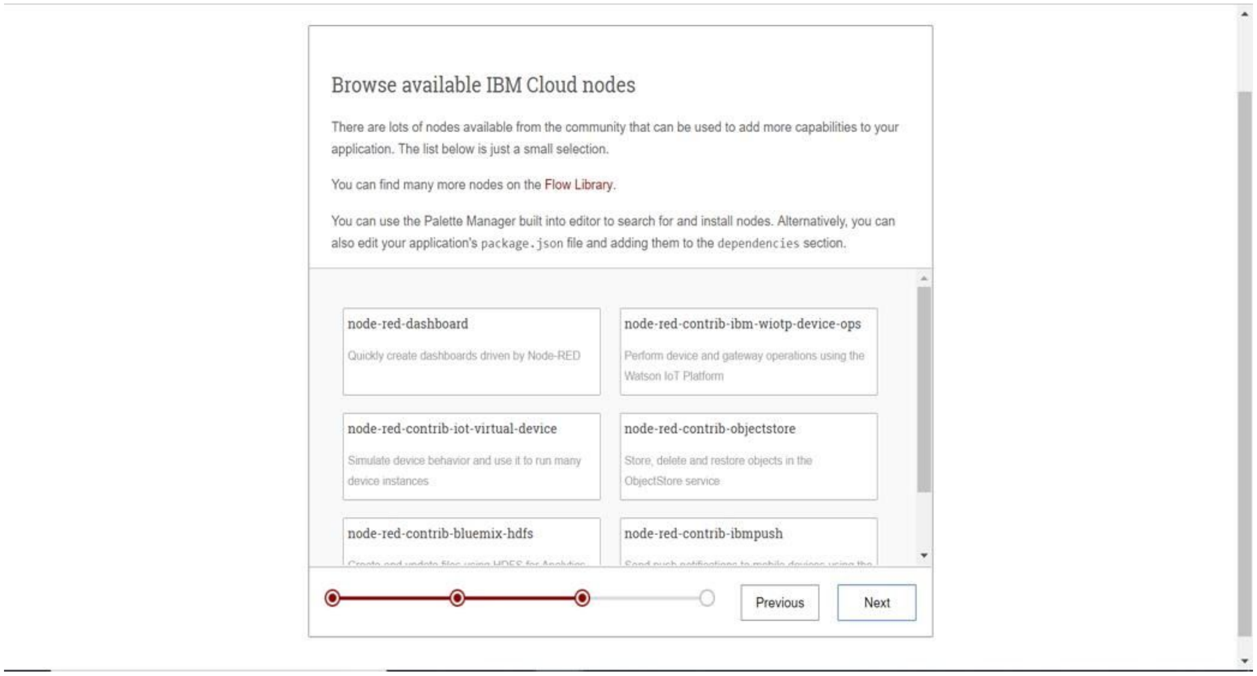
Username

Password — Must be at least 8 characters

☐ Allow anyone to view the editor, but not make any changes

( ) *Not recommended:* Allow anyone to access the editor and make changes

➢ Step – 7: Click Next to continue



Browse available IBM Cloud nodes

There are lots of nodes available from the community that can be used to add more capabilities to your application. The list below is just a small selection.

You can find many more nodes on the Flow Library.

You can use the Palette Manager built into editor to search for and install nodes. Alternatively, you can also edit your application's `package.json` file and adding them to the `dependencies` section.

node-red-dashboard
Quickly create dashboards driven by Node-RED

node-red-contrib-ibm-wiotp-device-ops
Perform device and gateway operations using the Watson IoT Platform

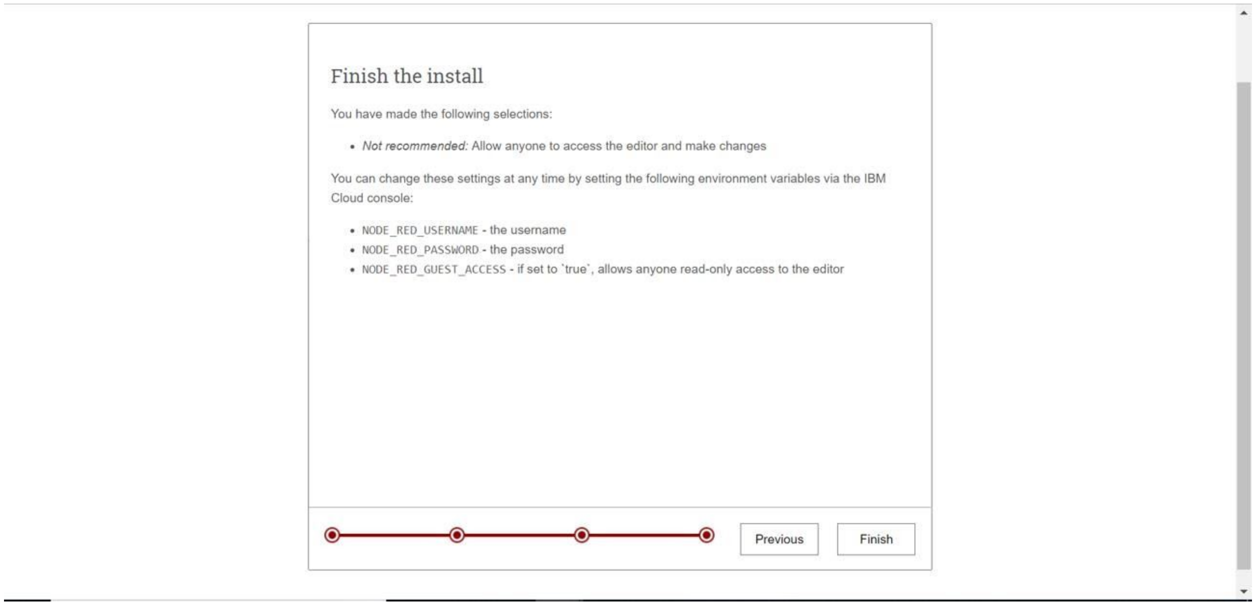
node-red-contrib-iot-virtual-device
Simulate device behavior and use it to run many device instances

node-red-contrib-objectstore
Store, delete and restore objects in the ObjectStore service

node-red-contrib-bluemix-hdfs

node-red-contrib-ibmpush

➢ Step – 8: Click Finish



Finish the install

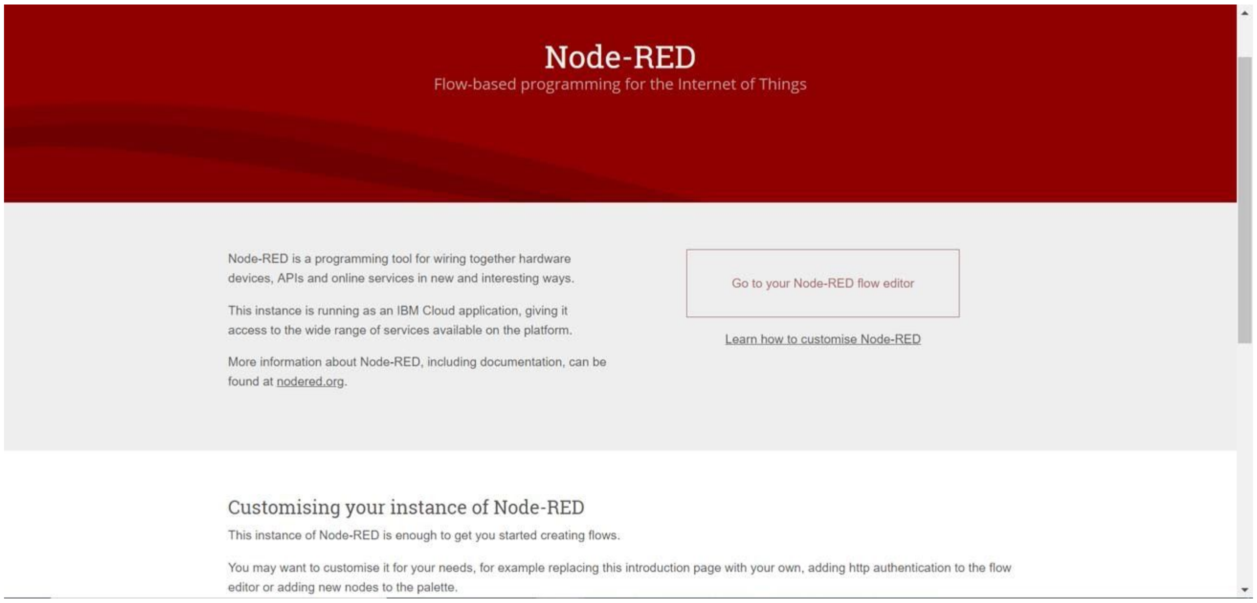
You have made the following selections:

- *Not recommended:* Allow anyone to access the editor and make changes

You can change these settings at any time by setting the following environment variables via the IBM Cloud console:

- `NODE_RED_USERNAME` - the username
- `NODE_RED_PASSWORD` - the password
- `NODE_RED_GUEST_ACCESS` - if set to `true`, allows anyone read-only access to the editor

➢ Step – 9: Click on Go to Node-Red flow editor to launch the flow editor



Node-RED

Flow-based programming for the Internet of Things

Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways.

This instance is running as an IBM Cloud application, giving it access to the wide range of services available on the platform.

More information about Node-RED, including documentation, can be found at nodered.org.

Go to your Node-RED flow editor

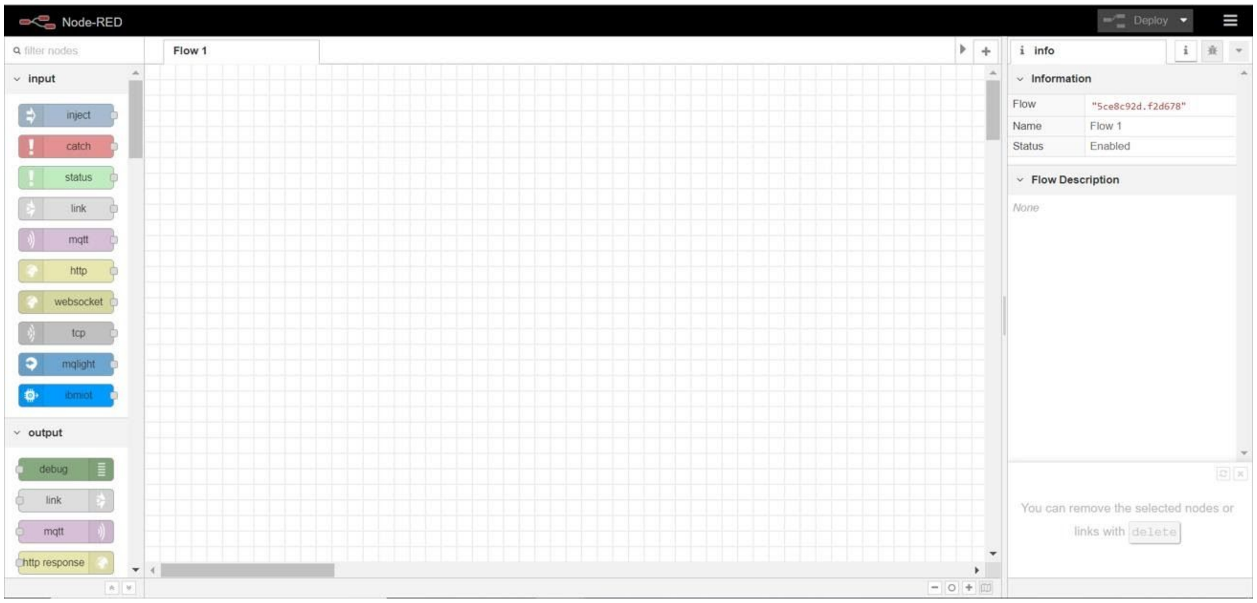
Learn how to customise Node-RED

Customising your instance of Node-RED

This instance of Node-RED is enough to get you started creating flows.

You may want to customise it for your needs, for example replacing this introduction page with your own, adding http authentication to the flow editor or adding new nodes to the palette.

➢ Node red editor has various nodes with the respective functionality



Flow	"5ce8c92d.f2d678"
Name	Flow 1
Status	Enabled

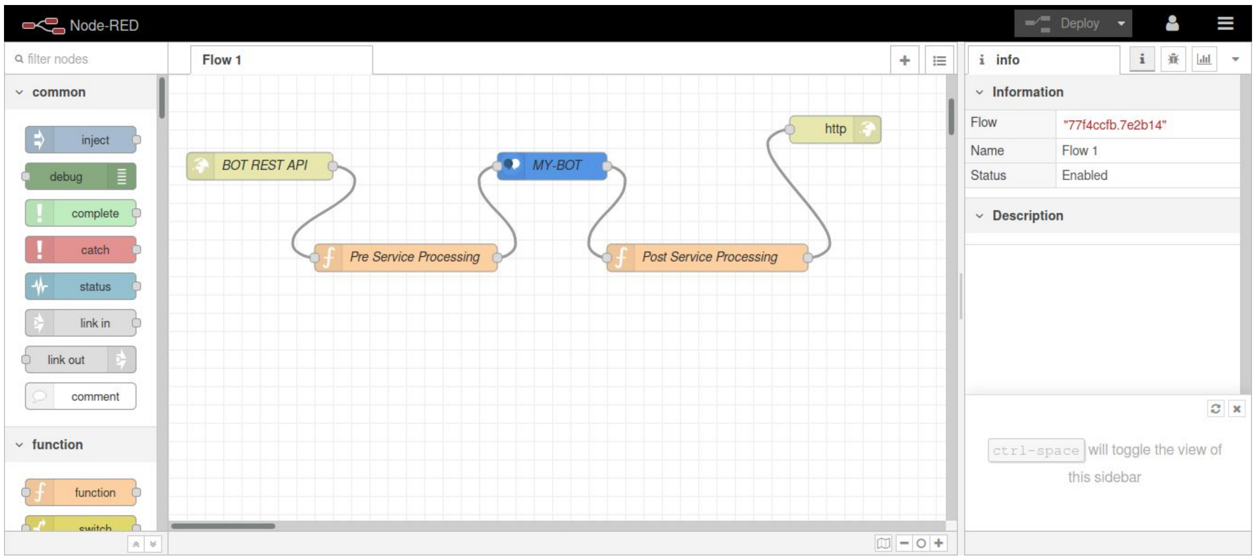
None

You can remove the selected nodes or links with delete

Integration of watson assistant in Node-RED

➢ Double-click on the Watson assistant node

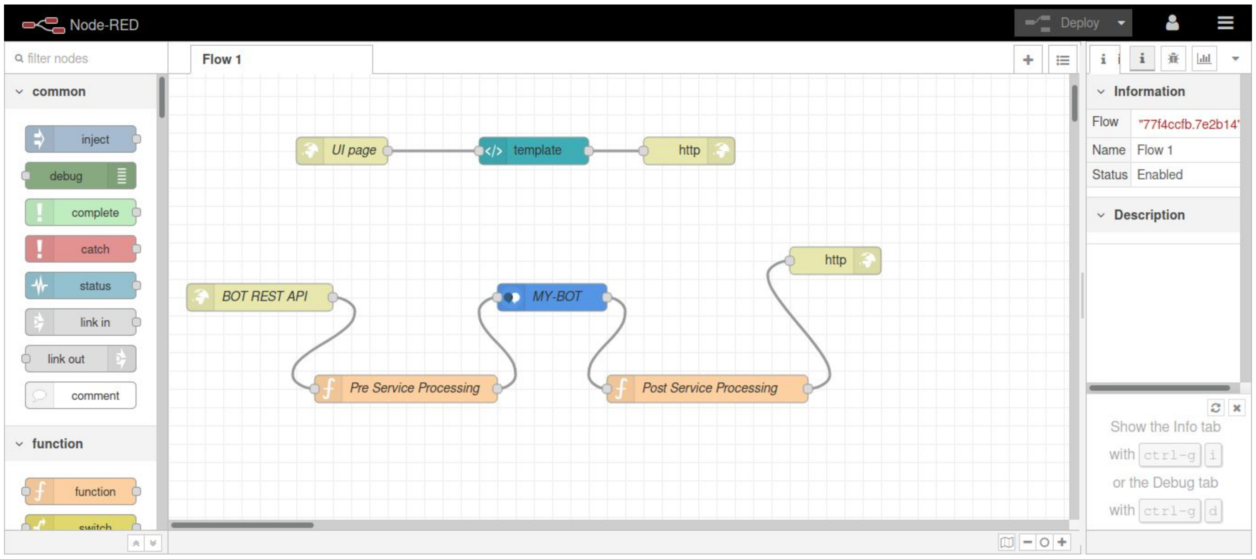
- ➢ Give a name to your node and enter the username, password and workspace id of your Watson assistant service
- ➢ After entering all the information click on Done
- ➢ Drag a http-in node, http-response node into the workspace
- ➢ Drag two functions node, configure those to perform the pre and post processing of the input and output of the watson assistant node
- ➢ Connect the http-in, preprocessing, watson assistant, post processing and http-response nodes to form an API
- ➢ Connect the nodes as shown below and click on Deploy
- ➢ Drag the function node to parse the JSON data and get the bot response
- ➢ Connect the nodes as shown below and click on Deploy



Flow	"77f4ccfb.7e2b14"
Name	Flow 1
Status	Enabled

We are done integrating Watson assistant service to Node-red. In the next lab, we will create a web application using Node-red for the chatbot. For creating a web application UI we need “dashboard“ nodes which should be installed manually.

- ➢ Go to navigation pane and click on manage palette
- ➢ Click on install
- ➢ Search for “node-red-dashboard” and click on install and again click on install on the prompt
- ➢ The following message indicates dashboard nodes are installed, close the manage palette
- ➢ Drag a http-in, template and http-response node
- ➢ Make a request to the API and display the response in the web dashboard in the template node
- ➢ Connect those nodes



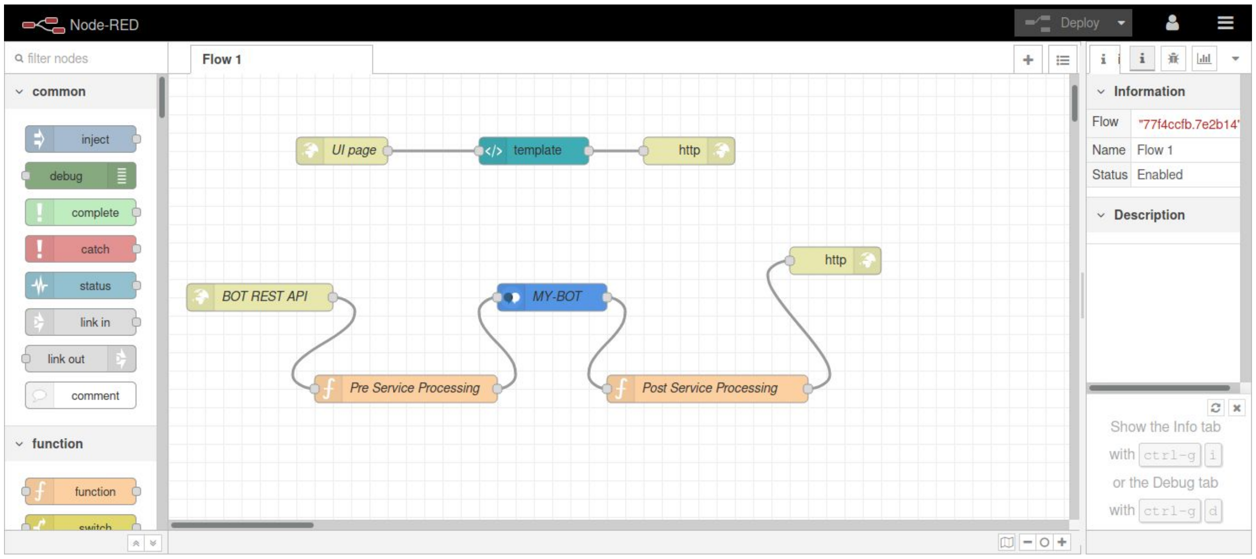
- ➢ Click on Deploy

FLOWCHART

1.Create flow and configure node:

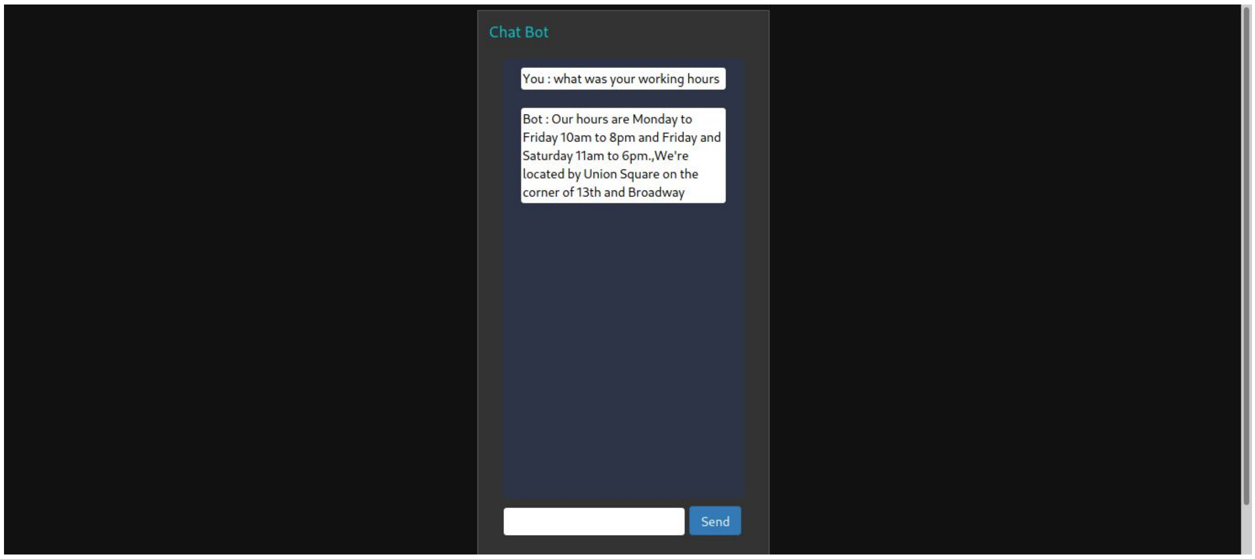
First go to manage the palette and install the dashboard. Now,Create the flow with the help of following node:

- ➢ Assistant
- ➢ Function
- ➢ Http-in
- ➢ Http-response
- ➢ Function
- ➢ Template



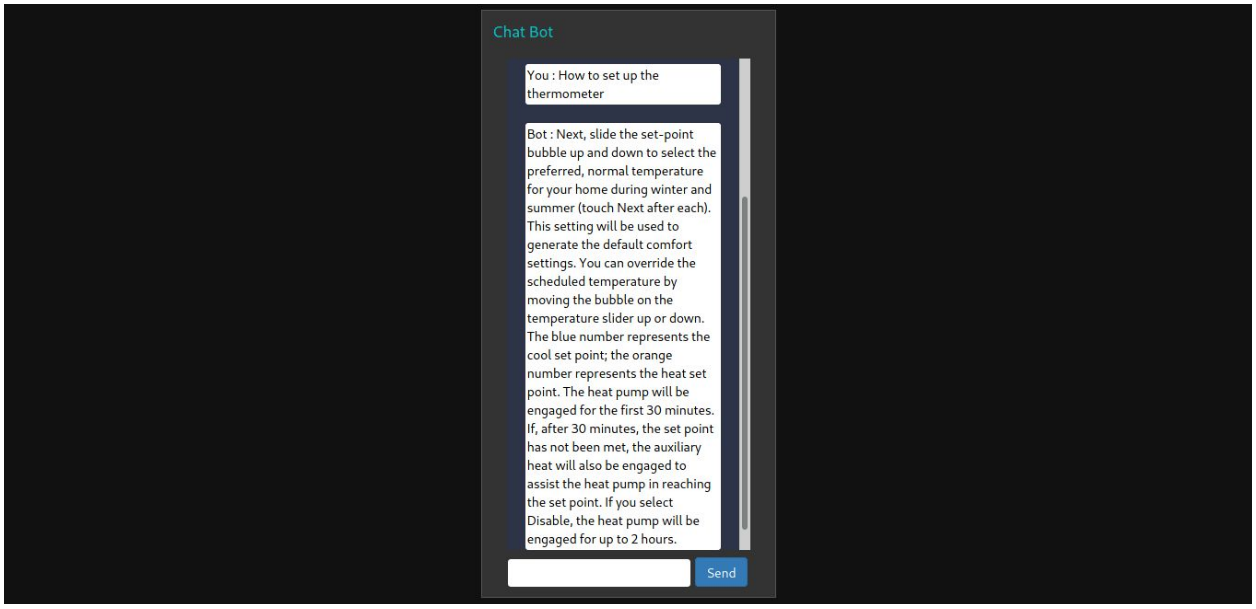
RESULTS

Finally our Node-RED dashboard integrates all the components and displayed in the Dashboard UI by typing URL- https://node-red-iztmp.eu-gb.mybluemix.net/ui in browser



You : what was your working hours

Bot : Our hours are Monday to Friday 10am to 8pm and Friday and Saturday 11am to 6pm.,We're located by Union Square on the corner of 13th and Broadway



You : How to set up the thermometer

Bot : Next, slide the set-point bubble up and down to select the preferred, normal temperature for your home during winter and summer (touch Next after each). This setting will be used to generate the default comfort settings. You can override the scheduled temperature by moving the bubble on the temperature slider up or down. The blue number represents the cool set point; the orange number represents the heat set point. The heat pump will be engaged for the first 30 minutes. If, after 30 minutes, the set point has not been met, the auxiliary heat will also be engaged to assist the heat pump in reaching the set point. If you select Disable, the heat pump will be engaged for up to 2 hours.

ADVANTAGES & DISADVANTAGES

Advantages:

➢ Companies can deploy chatbots to rectify simple and general human queries .

- ➢ Reduces man power
- ➢ Cost efficient
- ➢ No need to divert calls to customer agents and customer agents can look at other works.

Disadvantages:

- ➢ Some times chatbot can mislead customers
- ➢ Giving the same answer for different sentiments.
- ➢ Sometimes cannot connect to customer sentiments and intentions

APPLICATIONS

- ➢ It can deploy in popular social media applications like facebook,slack,telegram.
- ➢ Chatbot can deploy any website to clarify basic doubts of viewer

CONCLUSION

By doing the above procedure and all we successfully created an Intelligent help desk smart chatbot using Watson assistant, Watson discovery, Node-RED and cloud-functions.

FUTURE SCOPE

We can include watson studio text to speech and speech to text services to access the chatbot handsfree. This is one of the future scope of this project.

BIBLIOGRAPHY

APPENDIX

NODE-RED FLOW

[{"id":"77f4ccfb.7e2b14","type":"tab","label":"Flow 1","disabled":false,"info":""},{"id":"5120c7ff.d8f8d8","type":"http in","z":"77f4ccfb.7e2b14","name":"BOT REST API","url":"/botchat","method":"post","upload":false,"swaggerDoc":"","x":100,"y":340,"wires":[["66875061.7c44d"]]},{"id":"66875061.7c44d","type":"function","z":"77f4ccfb.7e2b14","name":"Pre Service Processing","func":"\n// stash away incoming data\nmsg.mydata = {};\nmsg.mydata.messagein = msg.req.body.msgdata;\nmsg.payload = msg.mydata.messagein;\n\nmsg.params = { \"context\": msg.req.body.context};\n\nreturn msg;","outputs":1,"noerr":0,"x":260,"y":440,"wires":[["e92cb8bc.22e0c8"]]},{"id":"e92cb8bc.22e0c8","type":"watson-conversation-v1","z":"77f4ccfb.7e2b14","name":"MY-BOT","workspaceid":"399ad9e1-f0d1-42a0-9365-569523b88706","multiuser":false,"context":true,"empty-payload":false,"service-endpoint":"https://api.eu-gb.assistant.watson.cloud.ibm.com/instances/7d6a44ec-a644-49a0-b4b1-2c51c09af9c9","timeout":"","optout-learning":false,"x":420,"y":340,"wires":[["4c17e4d0.0ef5ac"]]},{"id":"4c17e4d0.0ef5ac","type":"function","z":"77f4ccfb.7e2b14","name":"Post Service Processing","func":"msg.mydata.messageout = msg.payload;\n\nmsg.payload = {};\nmsg.payload.botresponse = msg.mydata;\n\nreturn msg;","outputs":1,"noerr":0,"x":590,"y":440,"wires":[["529bb52c.eafc3c"]]},{"id":"529bb52c.eafc3c","type":"http response","z":"77f4ccfb.7e2b14","name":"","statusCode":"","headers":{},"x":730,"y":300,"wires":[]},{"id":"d2ab7769.9c5f88","type":"ui_template","z":"77f4ccfb.7e2b14","group":"1e888e0.de7d272","name":"","order":1,"width":0,"height":0,"format":"<html>\n <head>\n   <meta charset=\"utf-8\">\n   <meta http-equiv=\"X-UA-Compatible\" content=\"IE=edge\">\n   <meta name=\"viewport\" content=\"width=device-width, initial-scale=1\">\n   <title>\n\t My BOT\n\t</title>\n\t<link rel=\"stylesheet\"\n   type=\"text/css\"\n href=\"https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css\" />\n </head>\n <body>\n\n   <div class=\"container\" style=\"overflow-x: hidden;\n   \n background-color:#2E3448;\n   min-height:80vh;\n   width:44vh;\n position:relative;\n   border-radius:5px;\n   margin-bottom:3px;\">\n   <div id=\"no-script\"class=\"bg-info\">\n   This application needs JavaScript enabled in your browser!\n </div>\n   <div id=\"id_contextdump\"></div>\n   \n   <div id=id_botchathistory

```
style=\"margin-top:5px;padding:5px;\">\n\t  </div>\n    </div>\n    <div style=\"bottom:0px;\n\t
display:flex;\n\t            width:100%;\n\t            margin-bottom:8px;\n\t            margin-top:5px;\n\t
height:32px;\">\n\t      <form>\n           <!--<label for=\"id_chattext\">Your Input: </label>-->\n           <input
style=\"border:None;\n                      margin:0px;\n                      Padding:2px;\n
background-color:white;\n                      height:30px;\n                      border-radius:3px;\n
margin-right:5px;\n                      margin-left:22px;\n                      margin-top:2px;\n
margin-bottom:8px;\n                      width:33vh;\n                      color:black;\" onfocus=\"this.value=''\"
type=\"text\" name=\"chattext\" id=\"id_chattext\">\n\t      </form>\n\t      <button class=\"btn
btn-primary\" onclick=\"javascript:onChatClick()\">Send</button>\n\t  </div>\n    <script
type=\"text/javascript\" src=\"https://code.jquery.com/jquery-2.1.4.min.js\"></script>\n    <script
src=\"https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js\"></script>\n\n    <script
type=\"text/javascript\">\n    \n      $(document).ready(function() {\n         javascriptCheck();\n
\t$('#id_contextdump').hide();\n      });\n\n      // if javascript is enabled on the browser then can\n      //
remove the warning message\n      function javascriptCheck() {\n        $('#no-script').remove();\n      }\n
\n      function createNewDiv(who, message) {\n        console.log('002-001');  \n        var txt = who + ' : ' +
message;\n        return $('<div
style=\"background-color:white;border-radius:3px;color:black;padding:2px;\"></div><br>').text(txt);\n
}\n\n      function chat(person, txt) {\n        $('#id_botchathistory').append(createNewDiv(person, txt));\n
}    \n      \n      function processOK(response) {\n        console.log('003-001');\n        //
console.log(response);\n        // console.log(response.botresponse.messageout);\n
console.log(response.botresponse.messageout.output.text[0]);\n
if(response.botresponse.messageout.output.text[0][0]==\"[\"){\n            var obj =
JSON.parse(response.botresponse.messageout.output.text[0]);\n            console.log(obj);\n            var i;\n
var result = \" \";\n            for(i=0;i<obj.length;i=i+1){\n                result += obj[i].passage_text + '    ';\n
}\n            console.log(result)\n            chat('Bot',result);\n        }\n        //
console.log(response.botresponse.messageout.context.webhook_result_1.passages[0].passage_text);\n
else{chat('Bot', response.botresponse.messageout.output.text); }\n        \n        //
console.log(response.botresponse.messageout.output.text[0].passage_text);\n
$('#id_contextdump').data('convContext', response.botresponse.messageout.context);\n      }\n      \n
function processNotOK() {\n        chat('Error', 'Error whilst attempting to talk to Bot');\n      }\n      \n
function invokeAjax(message) {\n        var contextdata = $('#id_contextdump').data('convContext');\n
// console.log('checking stashed context data');\n        // console.log(\"hi\"+contextdata);\n        \n  \n
//var ajaxData = \"msgdata=\" + message;\n        var ajaxData = {};\n        ajaxData.msgdata = message;\n
if (contextdata) {\n          ajaxData.context = contextdata;    \n        }\n\n        $.ajax({\n          type: 'POST',\n
```

url: 'https://node-red-iztmp.eu-gb.mybluemix.net/botchat',\n   data: ajaxData,\n   success: processOK,\n   error: processNotOK\n   });\n   }\n   \n   // User has entered some text.\n function onChatClick() {\n   // console.log('001-001');\n   var txt = $('#id_chattext').val();\n   // console.log(\"Hi\"+txt); txt- value given to the bot\n   chat('You', txt); \n   invokeAjax(txt);\n   // console.log('001-002');\n   }\n   \n   \n   </script>\n </body>\n</html>","storeOutMessages":true,"fwdInMessages":true,"resendOnRefresh":true,"templateScope":"local","x":400,"y":180,"wires":[["74a3eb34.6aa534"]]},{"id":"8db5345c.1b2fd8","type":"http in","z":"77f4ccfb.7e2b14","name":"UI page","url":"/ui","method":"get","upload":false,"swaggerDoc":"","x":190,"y":180,"wires":[["d2ab7769.9c5f88"]]},{"id":"74a3eb34.6aa534","type":"http response","z":"77f4ccfb.7e2b14","name":"","statusCode":"","headers":{},"x":570,"y":180,"wires":[]},{"id":"1e888e0.de7d272","type":"ui_group","z":"","name":"Chat Bot","tab":"2a456389.998e6c","order":1,"disp":true,"width":"6","collapse":false},{"id":"2a456389.998e6c","type":"ui_tab","z":"","name":"Home","icon":"dashboard","disabled":false,"hidden":false}]

Cloud function Node.js 10 code for discovery integration webhook generation:

```
/**

  *

  * @param {object} params

  * @param {string} params.iam_apikey

  * @param {string} params.url

  * @param {string} params.username

  * @param {string} params.password

  * @param {string} params.environment_id

  * @param {string} params.collection_id

  * @param {string} params.configuration_id

  * @param {string} params.input
```

```
 *
 * @return {object}
 *
 */


const assert = require('assert');
const DiscoveryV1 = require('watson-developer-cloud/discovery/v1');


/**
 *
 * main() will be run when you invoke this action
 *
 * @param Cloud Functions actions accept a single parameter, which must be a JSON object.
 *
 * @return The output of this action, which must be a JSON object.
 *
 */
function main(params) {
 return new Promise(function (resolve, reject) {


   let discovery;
```

```
if (params.iam_apikey){
  discovery = new DiscoveryV1({
    'iam_apikey': params.iam_apikey,
    'url': params.url,
    'version': '2019-03-25'
  });
}
else {
  discovery = new DiscoveryV1({
    'username': params.username,
    'password': params.password,
    'url': params.url,
    'version': '2019-03-25'
  });
}

discovery.query({
  'environment_id': params.environment_id,
  'collection_id': params.collection_id,
  'natural_language_query': params.input,
  'passages': true,
  'count': 3,
      'passages_count': 3
    }, function(err, data) {
      if (err) {
        return reject(err);
      }
      return resolve(data);
    });
  });
}
```