Intelligent Customer Help Desk With Smart Document Understand

Name: GARIMELLA SRI KRISHNA PRIYA

Category: Machine Learning

Mail id: garimellasripriya@gmail.com

college: SRM

Internship at smartinternz.com@2020

1. INTRODUCTION

a. Overview: Here we are going to prepare a chat bot by the use of IBM cloud services and the Watson services like assistant and discovery.

Services we are using in making chat bot are Discovery , Assistant, Cloud function and Node Red. By the end of the project, we'll learn how to make chat bot and best practices of integration of Watson services, and how they can build interactive information retrieval systems with Discovery and Assistant with the combination of functions and integrated to node red.

i. **Project Requirements**: Python, IBM Cloud, IBMWatson

ii. **Functional Requirements**: IBMcloud

iii. **Technical Requirements**: AI,ML,WATSONAI,PYTHON

iv. **Software Requirements**: Watson assistant, Watsondiscovery,node-red.

v. **Project Deliverables**: Smartinternz Intership

vi. **Project Team**: Garimella Sri Krishna Priya

vii. **Project Duration**:19days

b. Purpose:

The typical customer care chat bot 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 per-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 per-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 customer.

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.

i. Scope of Work

1. Create a customer care dialog skill in WatsonAssistant.

2. Use Smart Document Understanding to build an enhanced Watson Discovery collection.

3. Create an IBM Cloud Functions web action that allows Watson Assistant to post queries to WatsonDiscovery.

4. Build a web application with integration to all these services & deploy the same on IBM CloudPlatform.

2.LITERATURESURVEY

a. Existing problem:

A good customer Chat bot should minimize involvement of customer agent to chat with customer to clarify his/her doubts. So to achieve this we should include an virtual agent in chat bot so that it will take care of real involvement of customer agent and customer can clarifies his doubts with fast chat bots. In general we can say that the chat bot is the platform where the user can clarify his various doubts about the product which he/she want to purchase and they have the full rights to know about the product. Here we have to clarify the user doubts without the help of the agent and if the chat bot can't answer the queries the they can contact to the agent or any other person.

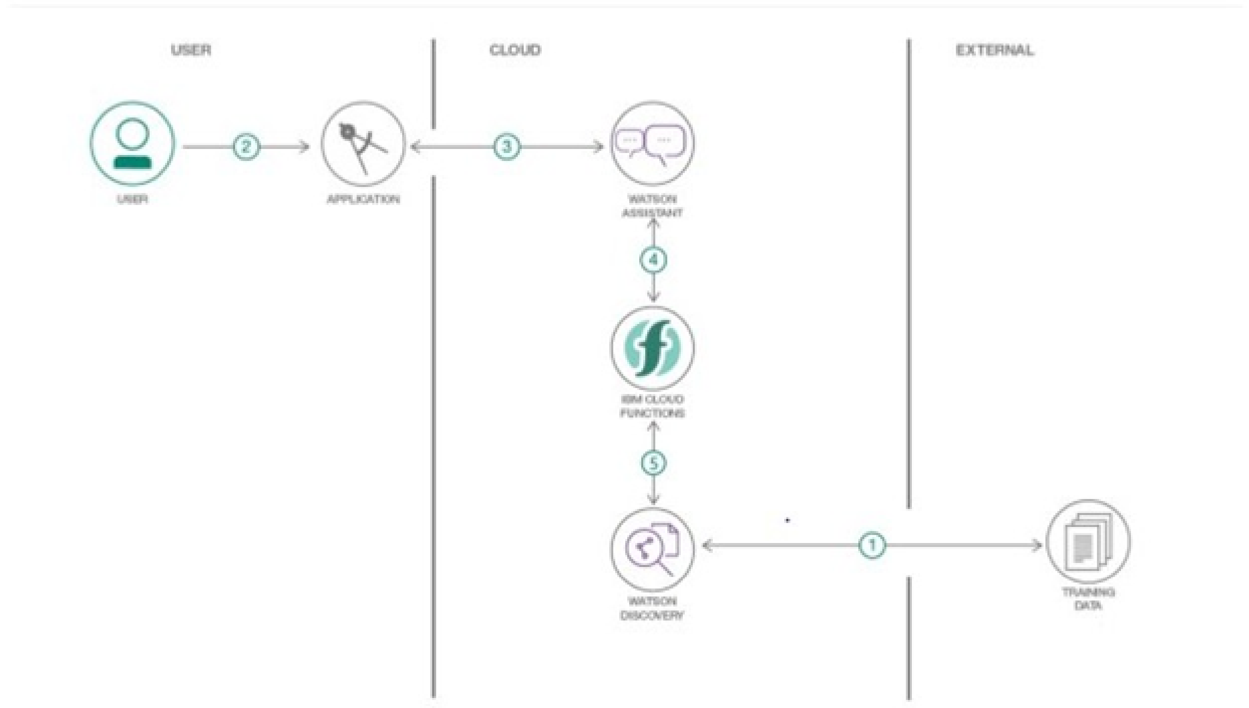
b. Proposed solution:

For the above problem to get solved we have to put an virtual agent in chat bot so it can understand the queries that are posted by customers. The virtual agent should trained from some insight records based company background so it can answer queries based on the product or related to company. In this project I have used Watson Discovery to achieve the above solution. And later including Assistant and Discovery on Node-RED.

We use function to integrate the discovery and then the assistant and then the integration of all these are done in node-red. The discovery contains the manuals where we train them and then with the help of functions we integrate it to assistant.

3. THEORITICAL ANALYSIS

Block/FlowDiagram



1. The document is annotated using Watson Discovery SDU.

2. The user interacts with the backend server via the app UI. The front end app UI is a chat bot 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

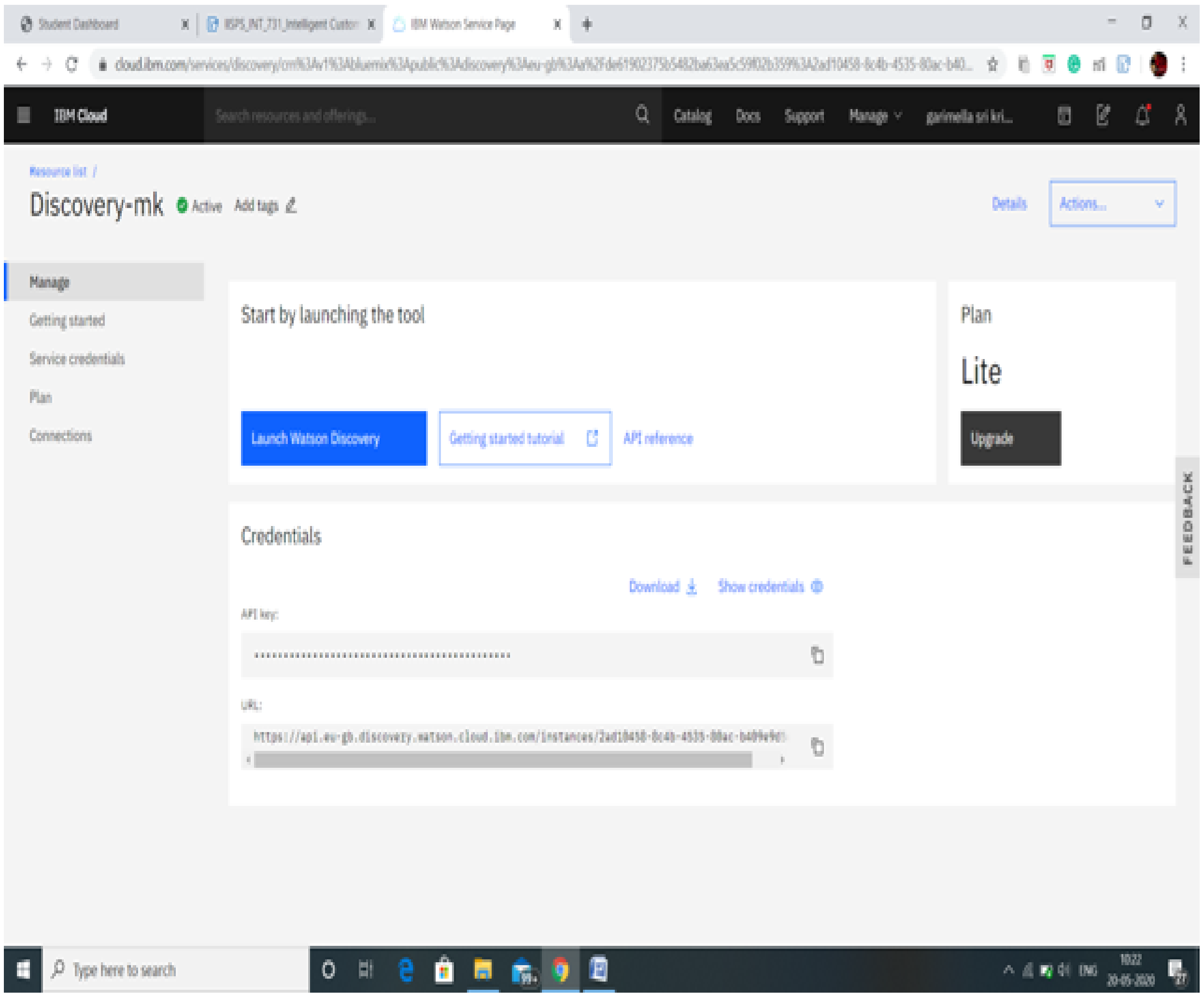
6. Deploy and run Node Red app.

4.EXPERIMENTAL INVESTIGATIONS

1.Create IBM Cloud services

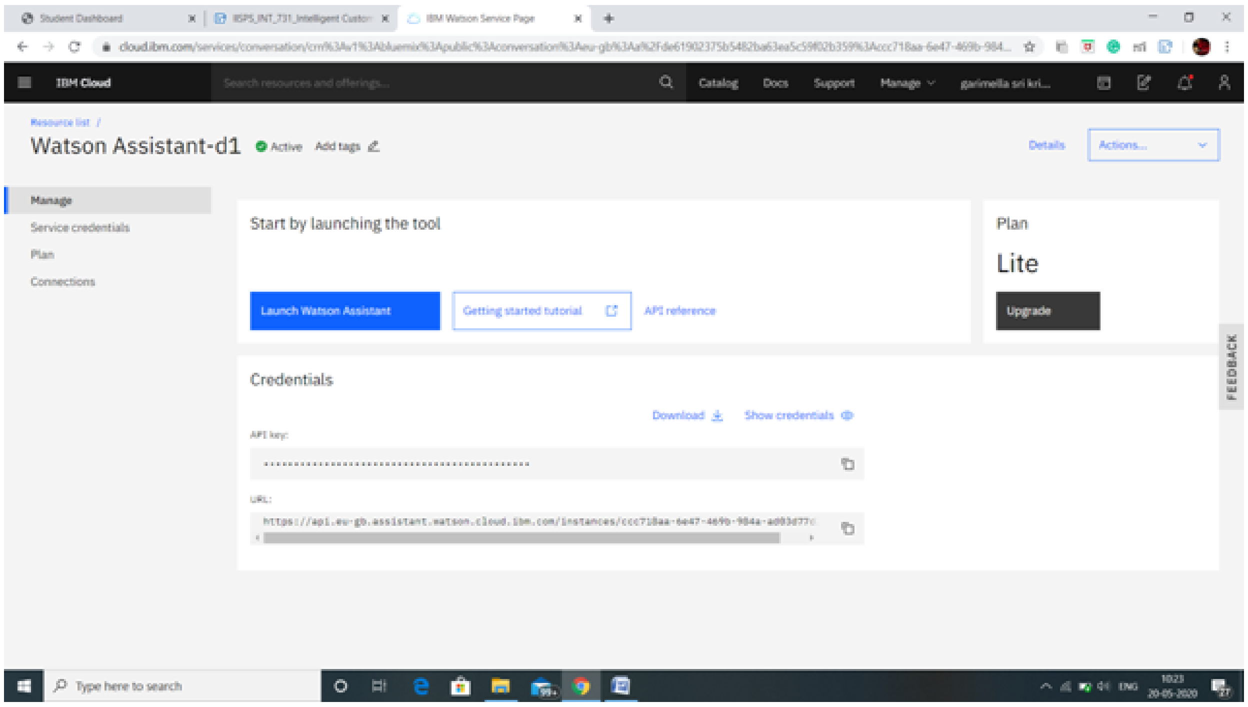
Create the following services:

a. WatsonDiscovery



```
https://api.eu-gb.discovery.watson.cloud.ibm.com/instances/2ad10458-8c4b-4535-80ac-b409e9d
```

b. WatsonAssistant



Resource list /

Watson Assistant-d1 Active Add tags

Manage
Service credentials
Plan
Connections

Start by launching the tool

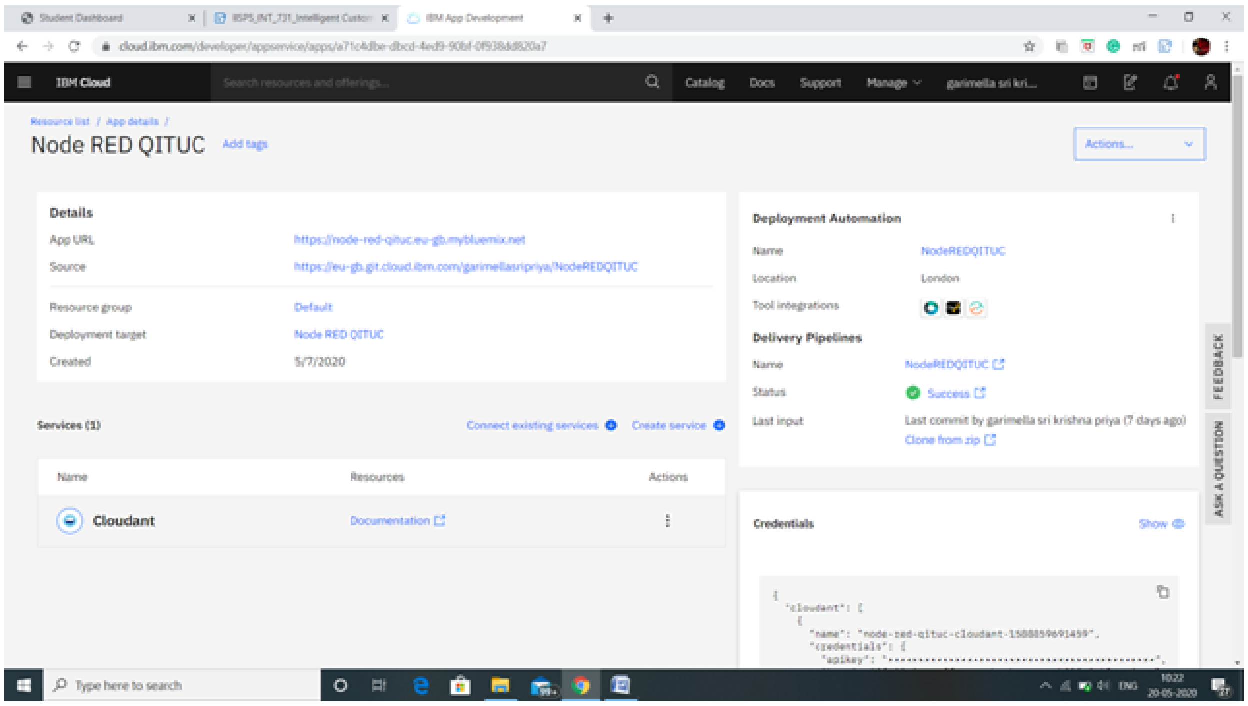
Plan
Lite

Credentials

API key:
```
*******************************************
```

URL:
```
https://api.eu-gb.assistant.watson.cloud.ibm.com/instances/ccc718aa-6e47-469b-984a-ad03d77c
```

c.NodeRed



Resource list / App details /

Node RED QITUC Add tags

Details	
App URL	https://node-red-qituc.eu-gb.mybluemix.net
Source	https://eu-gb.git.cloud.ibm.com/garimellasripriya/NodeREDQITUC
Resource group	Default
Deployment target	Node RED QITUC
Created	5/7/2020

Services (1)

Name	Resources	Actions
Cloudant	Documentation	

Deployment Automation

Name	NodeREDQITUC
Location	London
Tool integrations	

Delivery Pipelines

Name	NodeREDQITUC
Status	Success
Last input	Last commit by garimella sri krishna priya (7 days ago) Clone from zip

Credentials

```
{
  "cloudant": [
    {
      "name": "node-red-qituc-cloudant-1588859691459",
      "credentials": {
        "apikey": "********************************************",
```

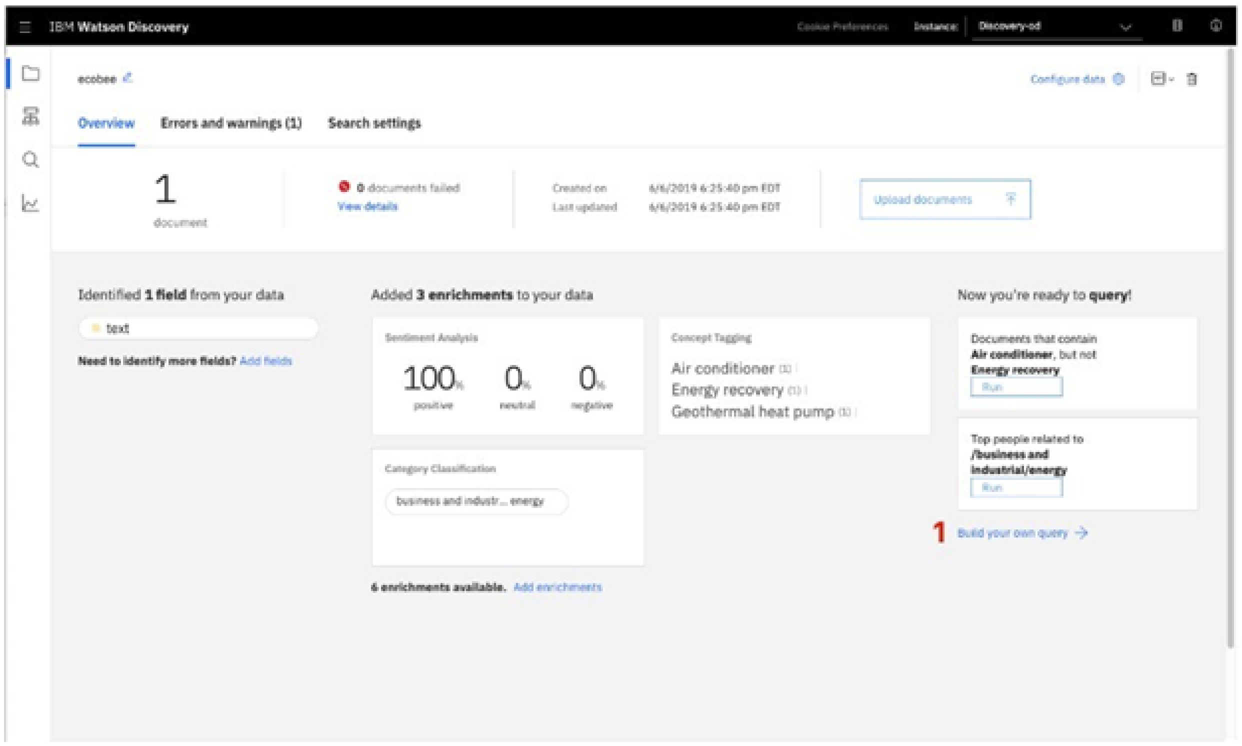
2. Configure WatsonDiscovery

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 rep o.

The Ecobee is a popular residential thermostat that has a wifi interface and multiple configuration options.

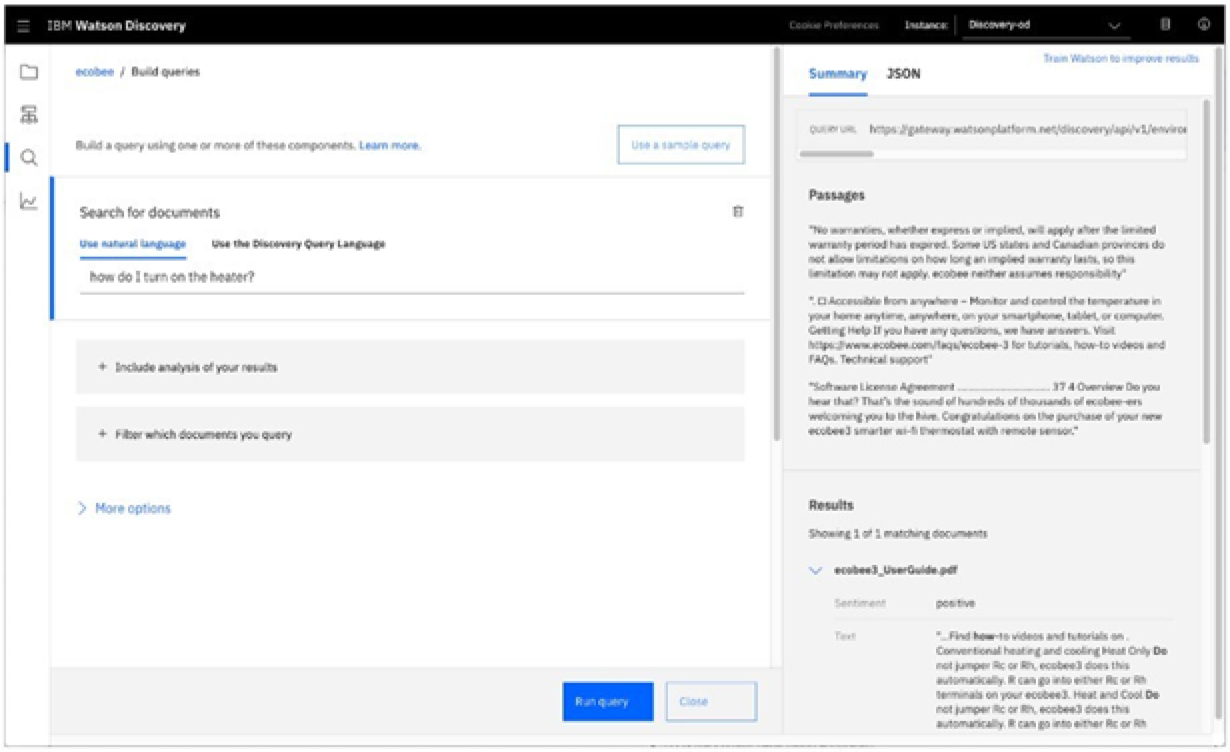
Before applying SDU to our document, lets do some simple queries on the data so that we can compare it to results found after applying SDU.



IBM Watson Discovery
Cookie Preferences
Instance: Discovery-od
ecobee
Configure data
Overview
Errors and warnings (1)
Search settings
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
Upload documents
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
Run
Top people related to /business and industrial/energy
Run
1
Build your own query

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.



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/enviro

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

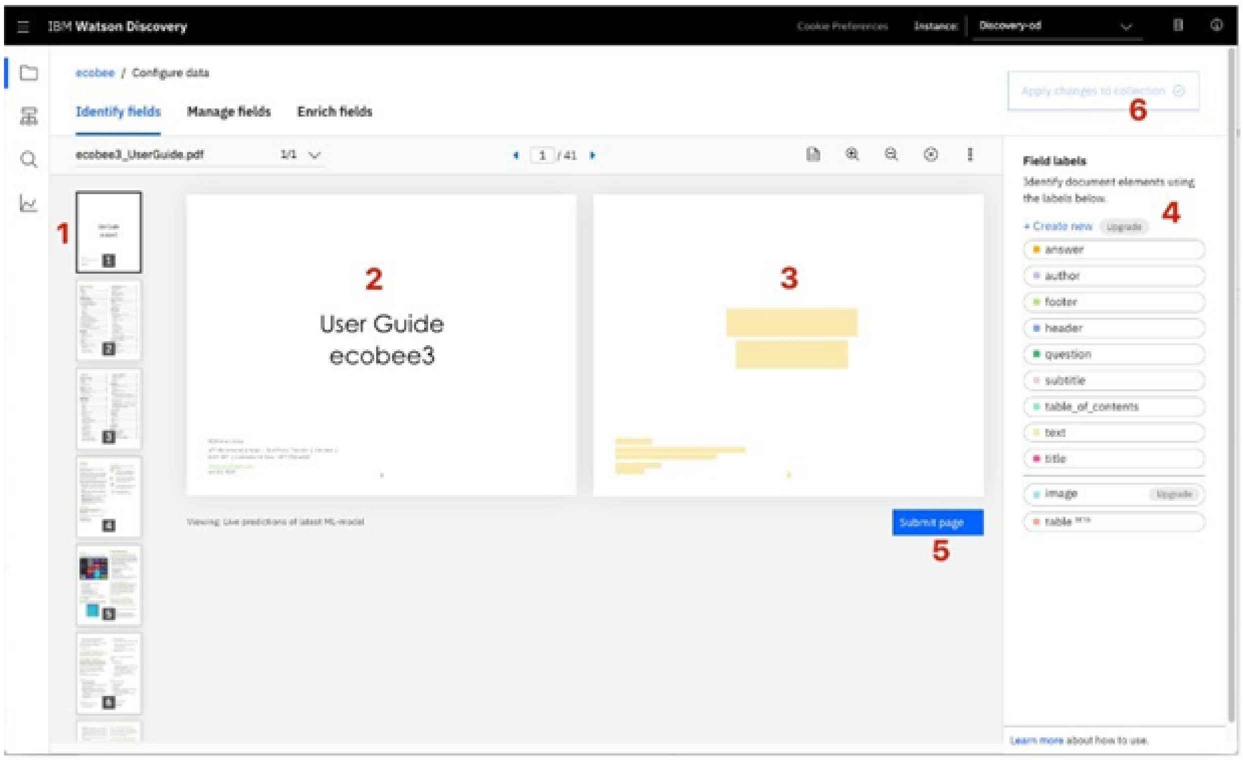
ecobee3_UserGuide.pdf

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

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 SDUprocess.

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 though 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

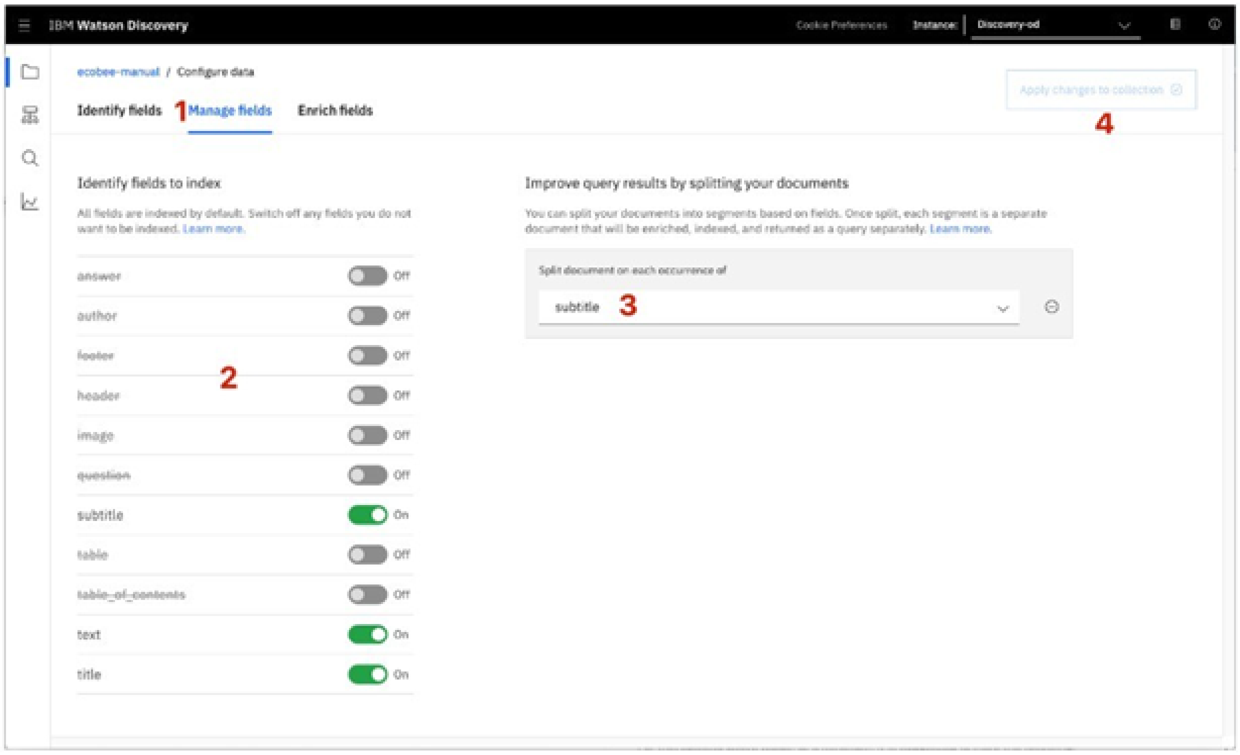
All page numbers as footers

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.



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

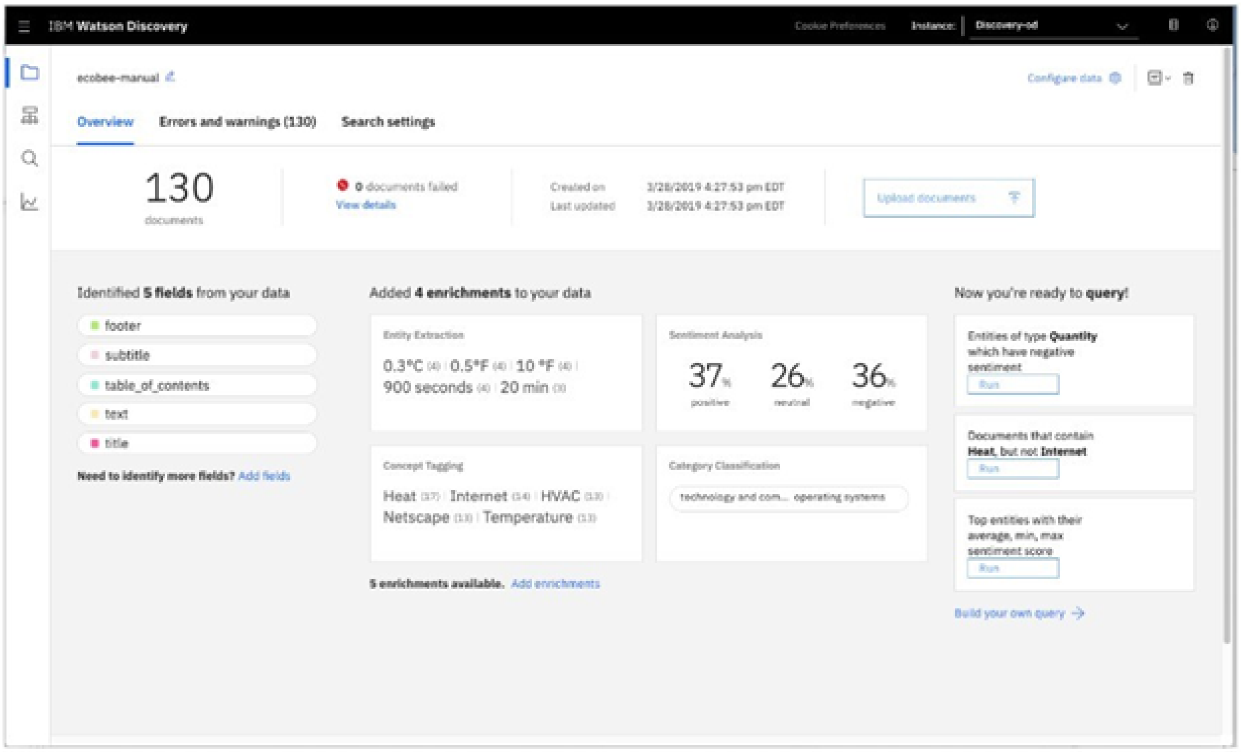
1. Here is where you tell Discovery which fields to ignore. Using the on/off buttons, turn off all labels except subtitles and text.

2. is telling Discovery to split the document apart, based on subtitle.

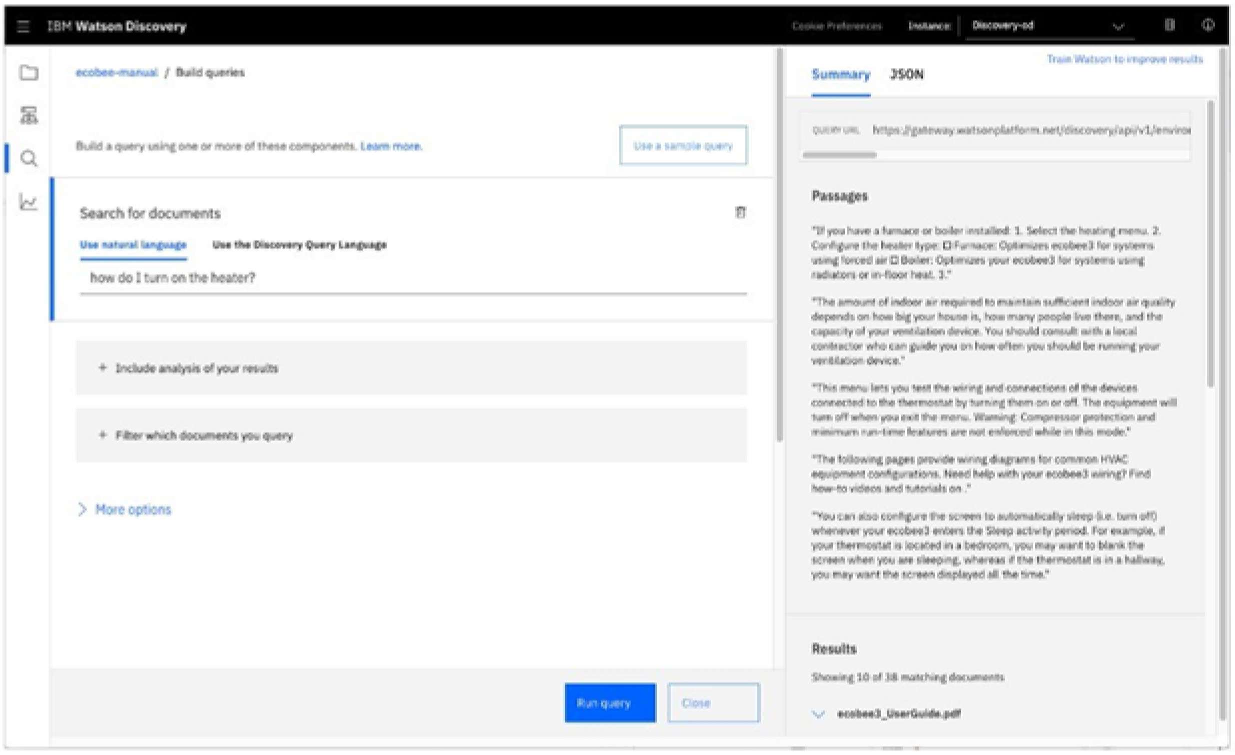
Click [4] to submit your changes.

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:


IBM Watson Discovery
Cookie Preferences
Instance: Discovery-od
ecobee-manual
Configure data
Overview
Errors and warnings (130)
Search settings
130
documents
0 documents failed
View details
Created on 3/28/2019 4:27:53 pm EDT
Last updated 3/28/2019 4:27:53 pm EDT
Upload documents
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 (37) | 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
Run
Documents that contain Heat, but not Internet
Run
Top entities with their average, min, max sentiment score
Run
Build your own query →

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
Use a sample query
Include analysis of your results
Filter which documents you query
More options
Run query
Close
Summary
JSON
Train Watson to improve results

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

Search for documents

Use natural language — Use the Discovery Query Language

how do I turn on the heater?

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

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

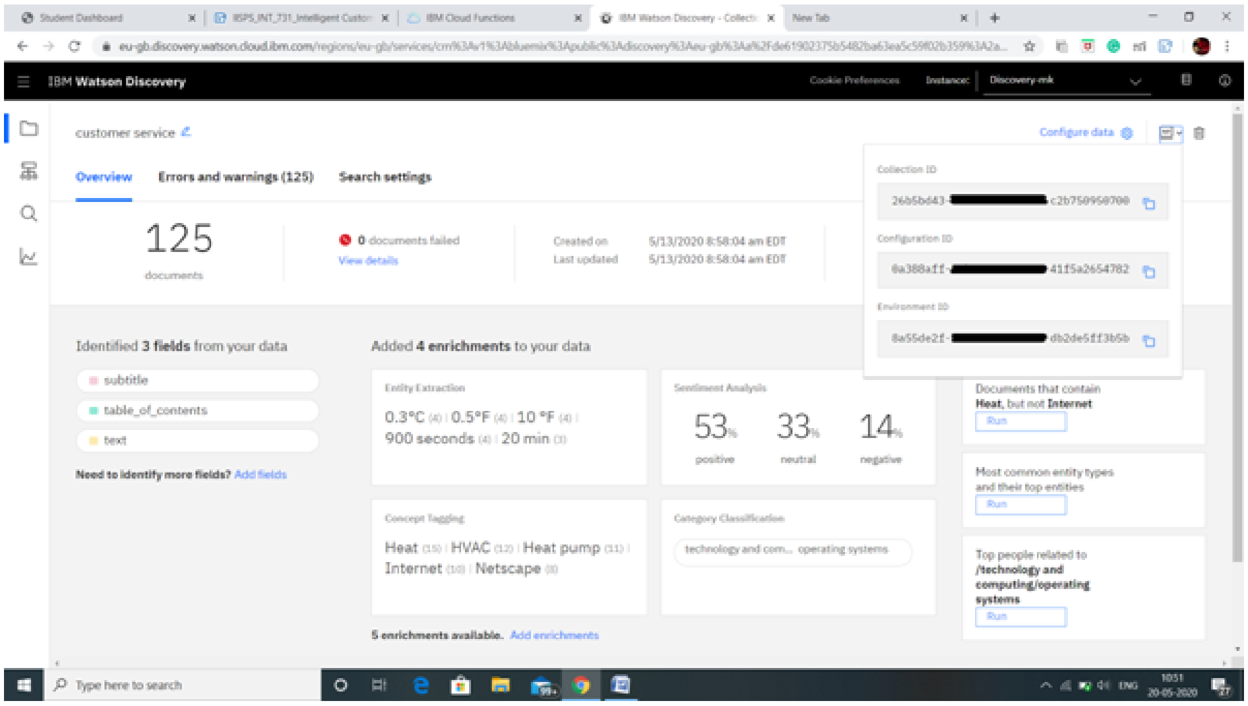
ecobee3_UserGuide.pdf

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:


IBM Watson Discovery

customer service

Overview | Errors and warnings (125) | Search settings

125 documents

0 documents failed
View details

Created on 5/13/2020 8:58:04 am EDT
Last updated 5/13/2020 8:58:04 am EDT

Configure data

Collection ID
26b5bd43-[redacted]c2b750950700

Configuration ID
0a388aff-[redacted]41f5a2654782

Environment ID
8a55de2f-[redacted]db2de5ff3b5b

Identified **3 fields** from your data
- subtitle
- table_of_contents
- text

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
53% positive | 33% neutral | 14% negative

Concept Tagging
Heat (15) | HVAC (12) | Heat pump (11) | Internet (10) | Netscape (8)

Category Classification
technology and com... operating systems

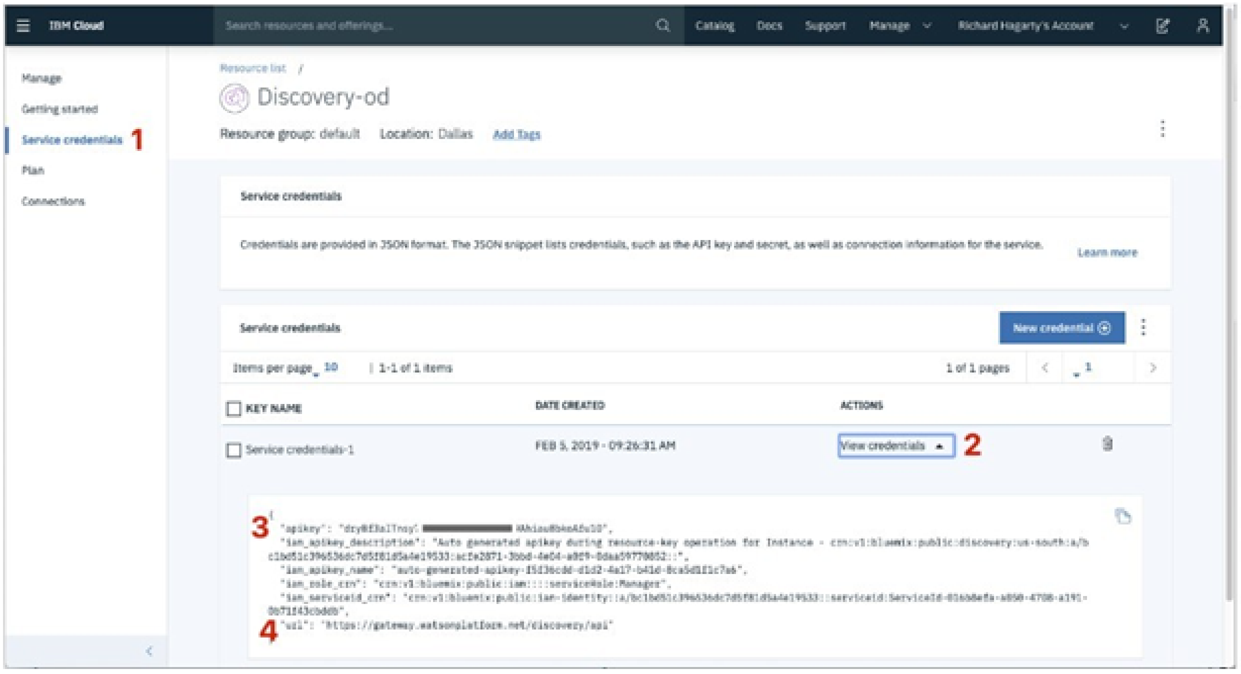
5 enrichments available. Add enrichments

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

Most common entity types and their top entities
Run

Top people related to **/technology and computing/operating systems**
Run

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


IBM Cloud

Resource list /

Discovery-od

Resource group: default   Location: Dallas   Add Tags

Manage
Getting started
Service credentials 1
Plan
Connections

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. Learn more

Service credentials   New credential

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

KEY NAME	DATE CREATED	ACTIONS
Service credentials-1	FEB 5, 2019 - 09:26:31 AM	View credentials 2

```
3{
  "apikey": "dryRE3alTnoy[redacted]4Ahiau8b4oA5u10",
  "iam_apikey_description": "Auto generated apikey during resource-key operation for Instance - crn:v1:bluemix:public:discovery:us-south:a/b
c1bd51c3965 36dc7d5f81d5a4e19533:acfe2871-3bbd-4e04-a8f9-0daa59770852::",
  "iam_apikey_name": "auto-generated-apikey-15d36cdd-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-a850-4708-a191-
0b71f43cbdeb",
4 "url": "https://gateway.watsonplatform.net/discovery/api"
```

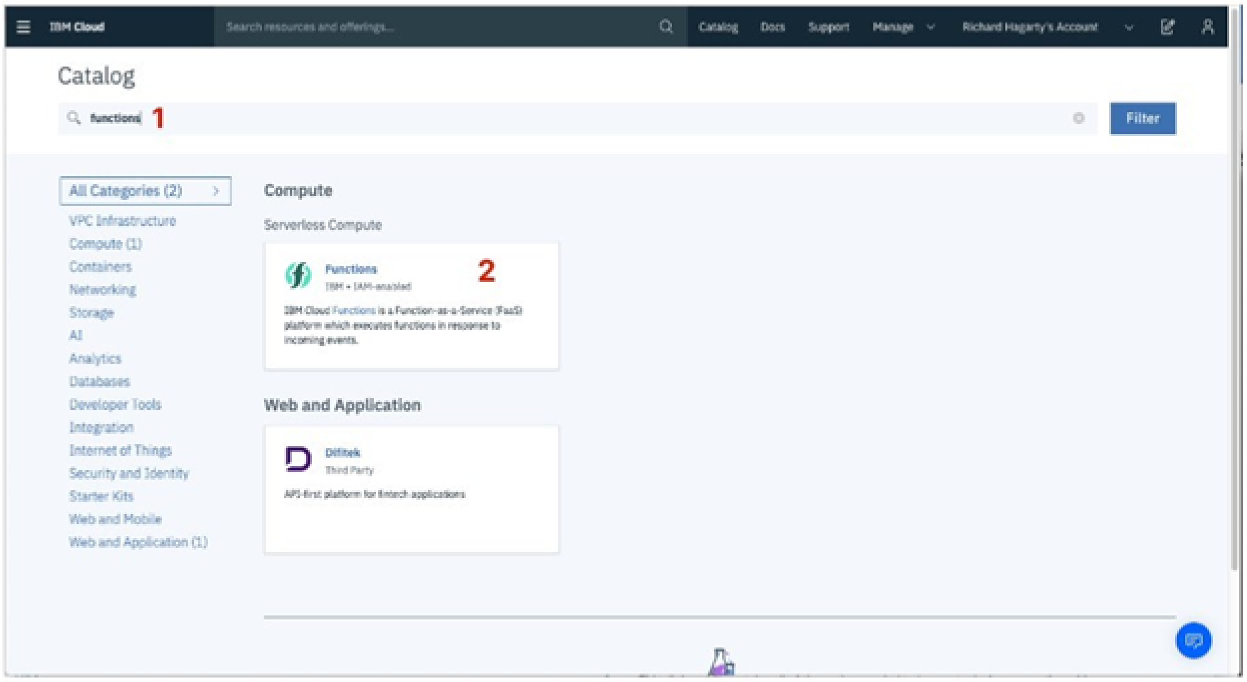
Click the View credentials [2] drop-down menu to view the IAM api key [3] and URL endpoint

[4] for your service.

3. Create IBM Cloud Functionsaction

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]:



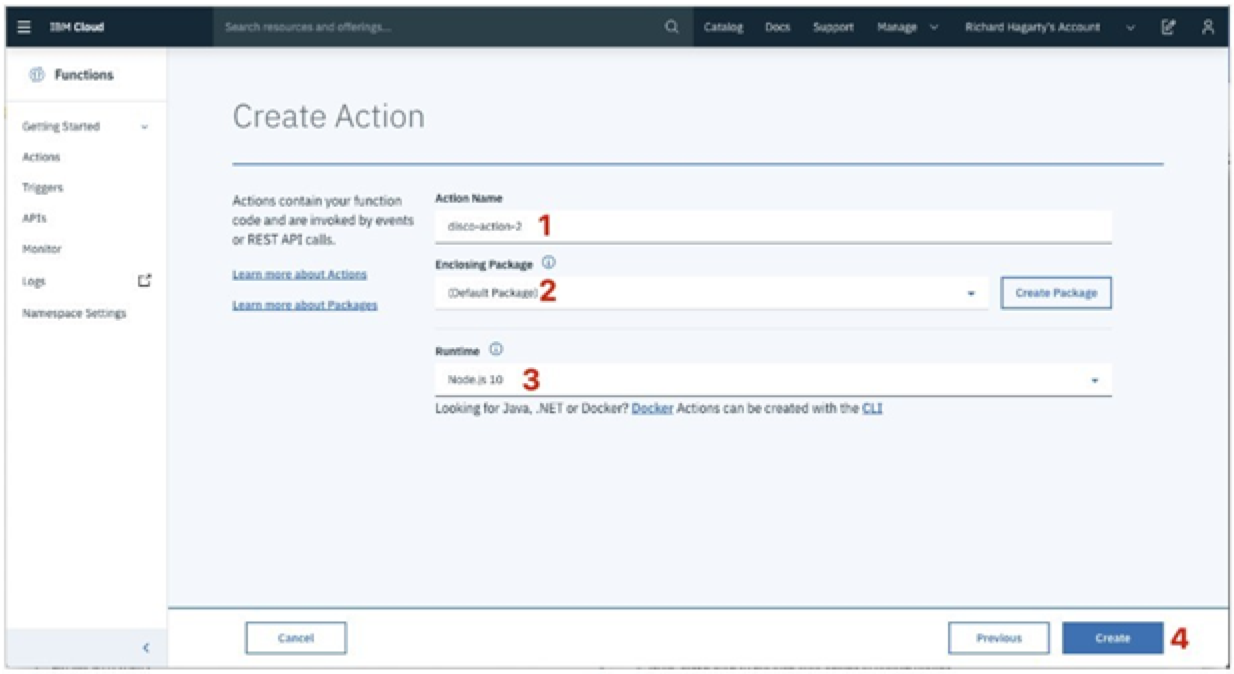
IBM Cloud Functions is a Function-as-a-Service (FaaS) platform which executes functions in response to incoming events.

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.



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

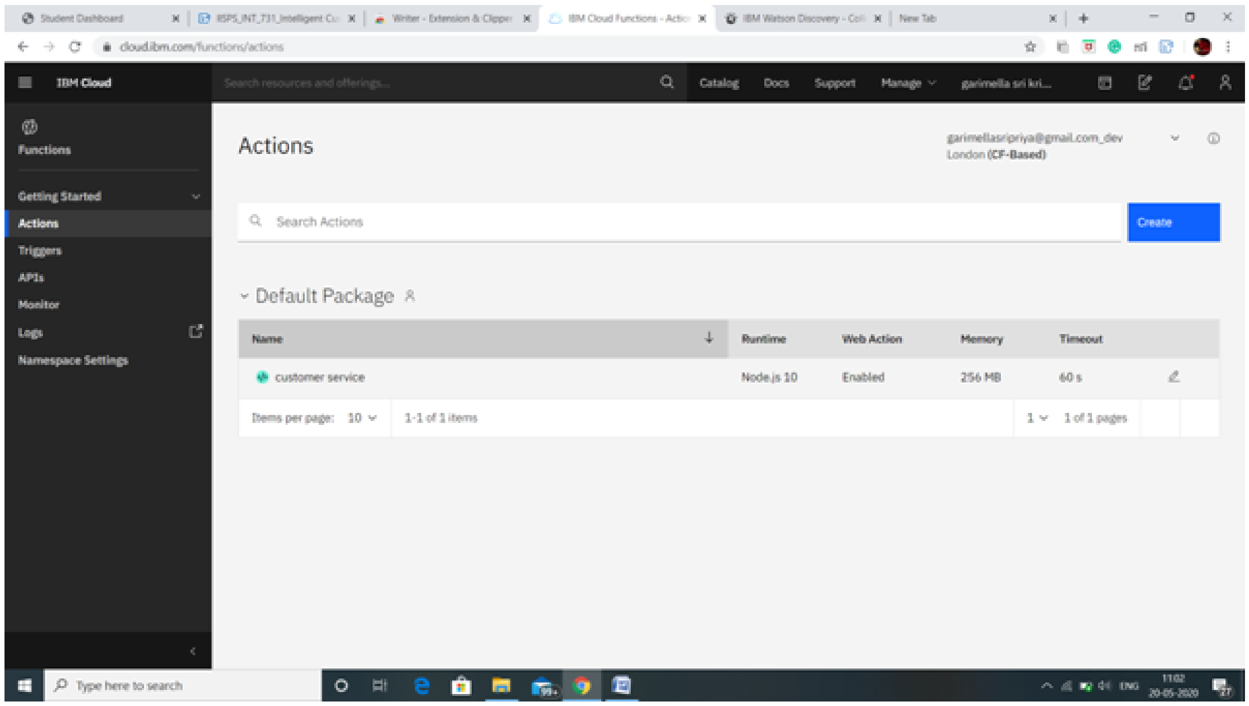
Enclosing Package: (Default Package) 2

Create Package

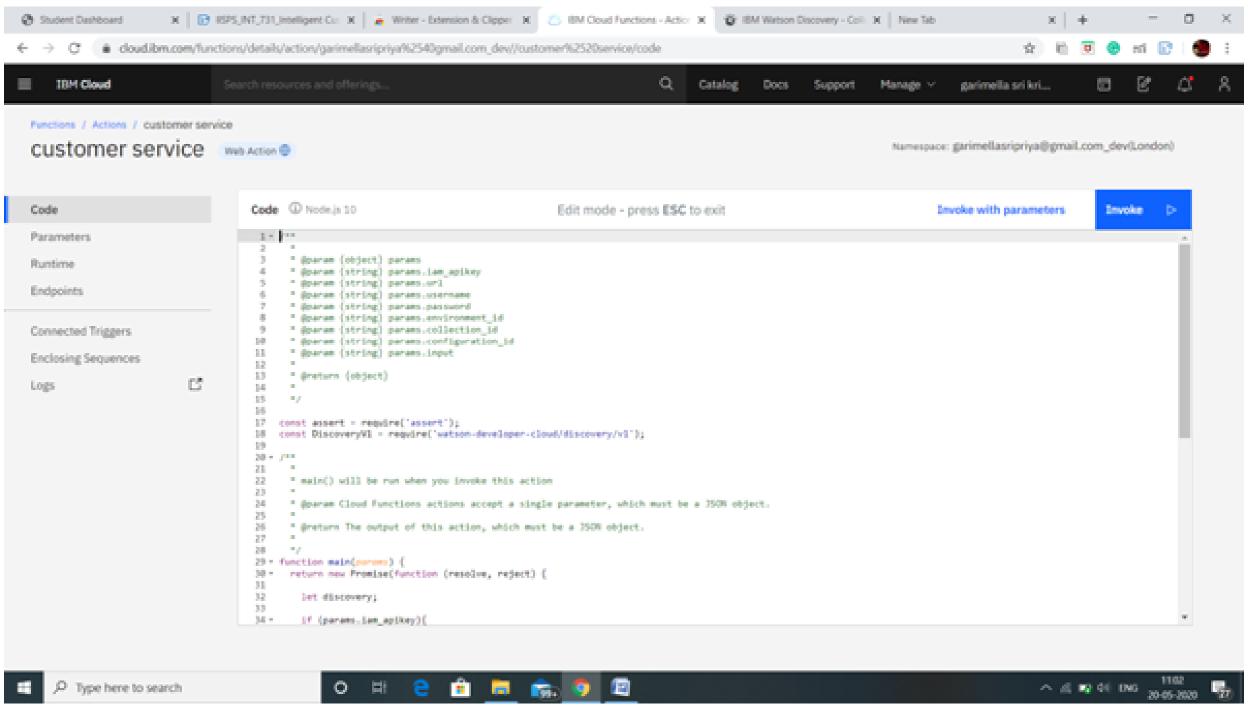
Runtime: Node.js 10 3

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]:



Name	Runtime	Web Action	Memory	Timeout
customer service	Node.js 10	Enabled	256 MB	60 s

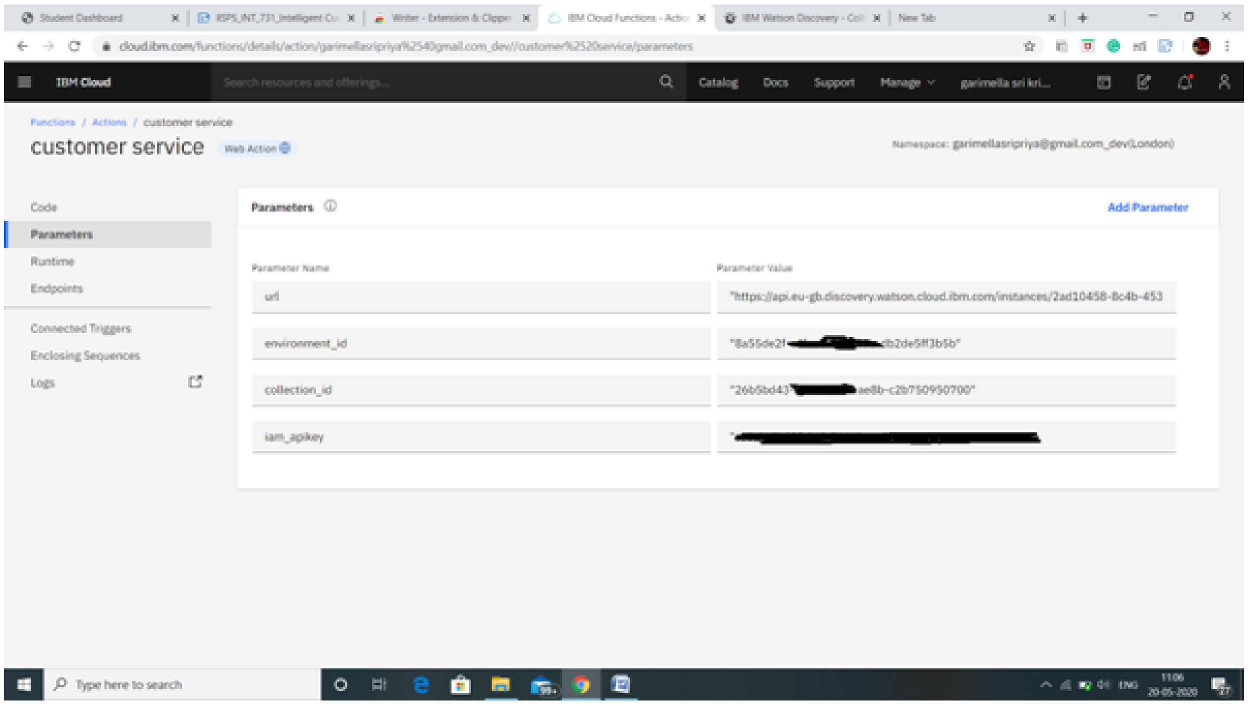


```
/**
  *
  * @param {object} params
  * @param {string} params.iam_apikey
  * @param {string} params.url
  * @param {string} params.username
  * @param {string} params.password
  * @param {string} params.environment_id
  * @param {string} params.collection_id
  * @param {string} params.configuration_id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

In the code editor window [2], cut and paste in the code from the disco-action.is file found in the actions directory of your local rep o. 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]:



Namespace: garimellasripriya@gmail.com_dev(London)

Parameters

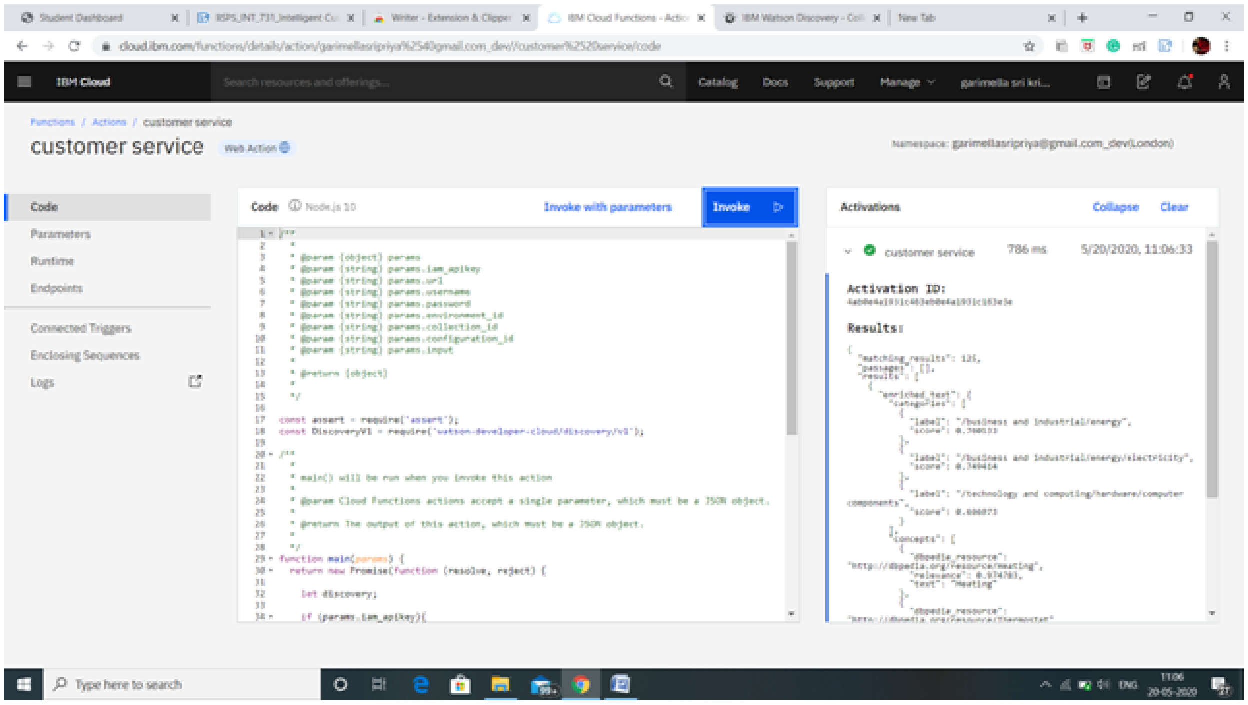
Parameter Name	Parameter Value
url	"https://api.eu-gb.discovery.watson.cloud.ibm.com/instances/2ad10458-8c4b-453
environment_id	"8a55de2f[redacted]db2de5ff3b5b"
collection_id	"26b5bd43[redacted]ae8b-c2b750950700"
iam_apikey	"[redacted]

Add the following keys:

a. url
b. environment_id
c. collection_id
d. am_api key

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 Discoveryservice:


Functions / Actions / customer service

customer service Web Action

Namespace: garimellasripriya@gmail.com_dev(London)

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

Code Node.js 10 Invoke with parameters Invoke

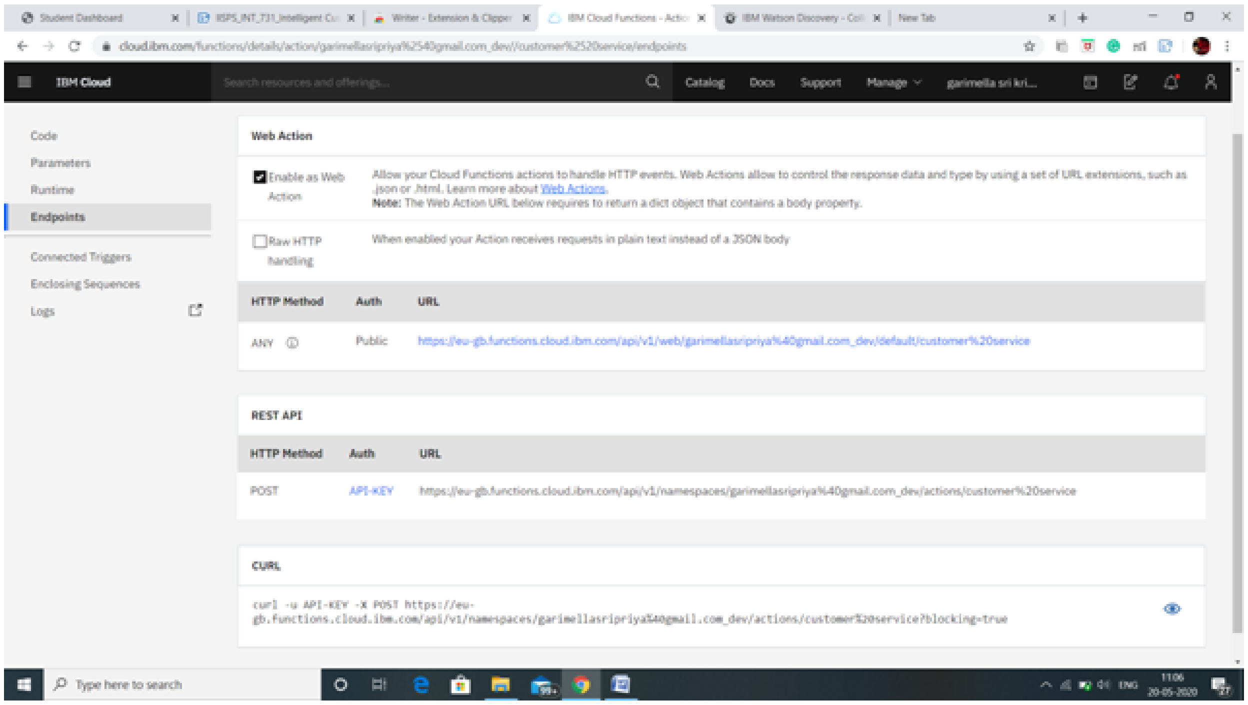
```
/**
  *
  * @param {object} params
  * @param {string} params.iam_apikey
  * @param {string} params.url
  * @param {string} params.username
  * @param {string} params.password
  * @param {string} params.environment_id
  * @param {string} params.collection_id
  * @param {string} params.configuration_id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

Activations Collapse Clear

customer service 786 ms 5/20/2020, 11:06:33

```
Activation ID:
4ab0e4a1931c463eb0e4a1931c163e3e

Results:
{
  "matching_results": 125,
  "passages": [],
  "results": [
    {
      "enriched_text": {
        "categories": [
          {
            "label": "/business and industrial/energy",
            "score": 0.760533
          },
          {
            "label": "/business and industrial/energy/electricity",
            "score": 0.749414
          },
          {
            "label": "/technology and computing/hardware/computer components",
            "score": 0.606873
          }
        ],
        "concepts": [
          {
            "dbpedia_resource": "http://dbpedia.org/resource/Heating",
            "relevance": 0.974783,
            "text": "Heating"
          },
          {
            "dbpedia_resource": "http://dbpedia.org/resource/Thermostat"
```

Next, go to the Endpoints panel [1]:


cloud.ibm.com/functions/details/action/garimellasripriya%2540gmail.com_dev//customer%2520service/endpoints

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

Web Action

[✓] Enable as Web Action — Allow your Cloud Functions actions to handle HTTP events. Web Actions allow to control the response data and type by using a set of URL extensions, such as .json or .html. Learn more about Web Actions.
Note: The Web Action URL below requires to return a dict object that contains a body property.

[ ] Raw HTTP handling — When enabled your Action receives requests in plain text instead of a JSON body

HTTP Method	Auth	URL
ANY	Public	https://eu-gb.functions.cloud.ibm.com/api/v1/web/garimellasripriya%40gmail.com_dev/default/customer%20service

REST API

HTTP Method	Auth	URL
POST	API-KEY	https://eu-gb.functions.cloud.ibm.com/api/v1/namespaces/garimellasripriya%40gmail.com_dev/actions/customer%20service

CURL

```
curl -u API-KEY -X POST https://eu-
gb.functions.cloud.ibm.com/api/v1/namespaces/garimellasripriya%40gmail.com_dev/actions/customer%20service?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 provide curl command [4]. If it fails, re-check your parameter values.

4. Configure Watson Assistant

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

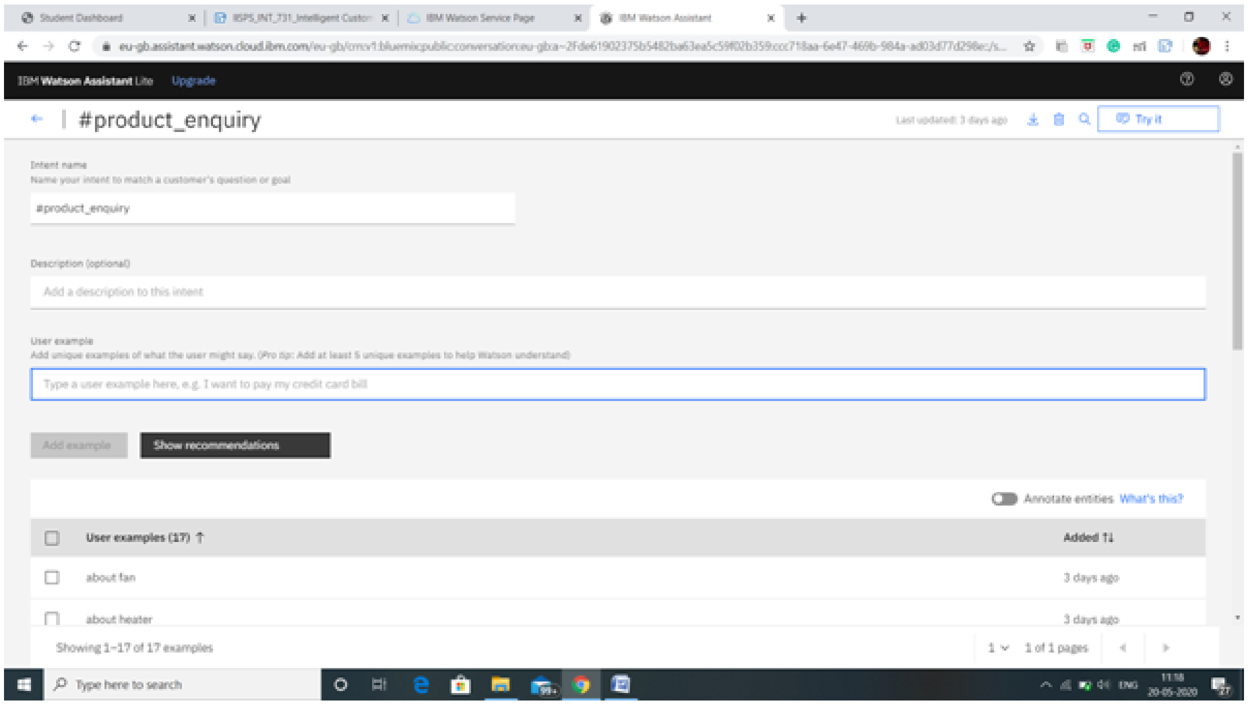
The default customer care dialog does not have a way to deal with any questions involving outside resources, so we will need to add this.

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.

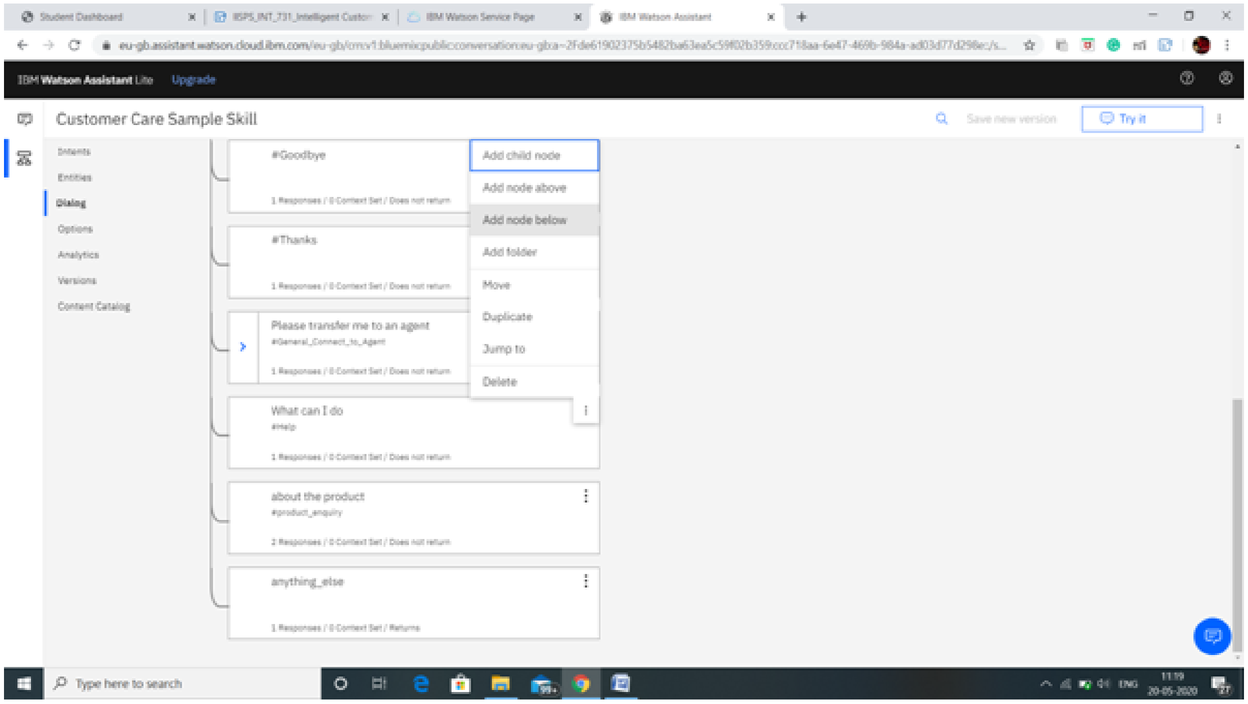
Click the Create intent button.

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

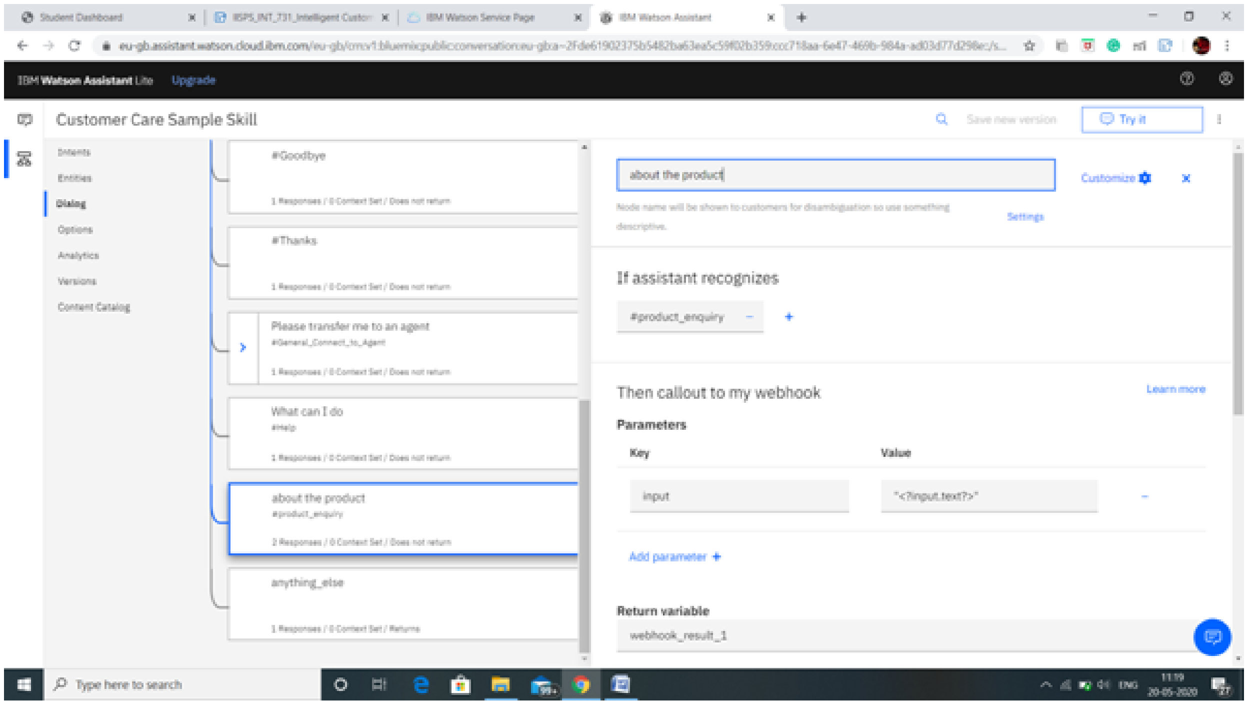

IBM Watson Assistant Lite
Upgrade
#product_enquiry
Try it
Intent name
Name your intent to match a customer's question or goal
#product_enquiry
Description (optional)
Add a description to this intent
User example
Add unique examples of what the user might say. (Pro tip: Add at least 5 unique examples to help Watson understand)
Type a user example here, e.g. I want to pay my credit card bill
Add example
Show recommendations
Annotate entities
What's this?
User examples (17)
Added
about fan
3 days ago
about heater
3 days ago
Showing 1–17 of 17 examples
1 of 1 pages

C rate 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.



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



about the product

Node name will be shown to customers for disambiguation so use something descriptive.

If assistant recognizes

#product_enquiry

Then callout to my webhook

Parameters

Key	Value
input	"<?input.text?>"

Return variable

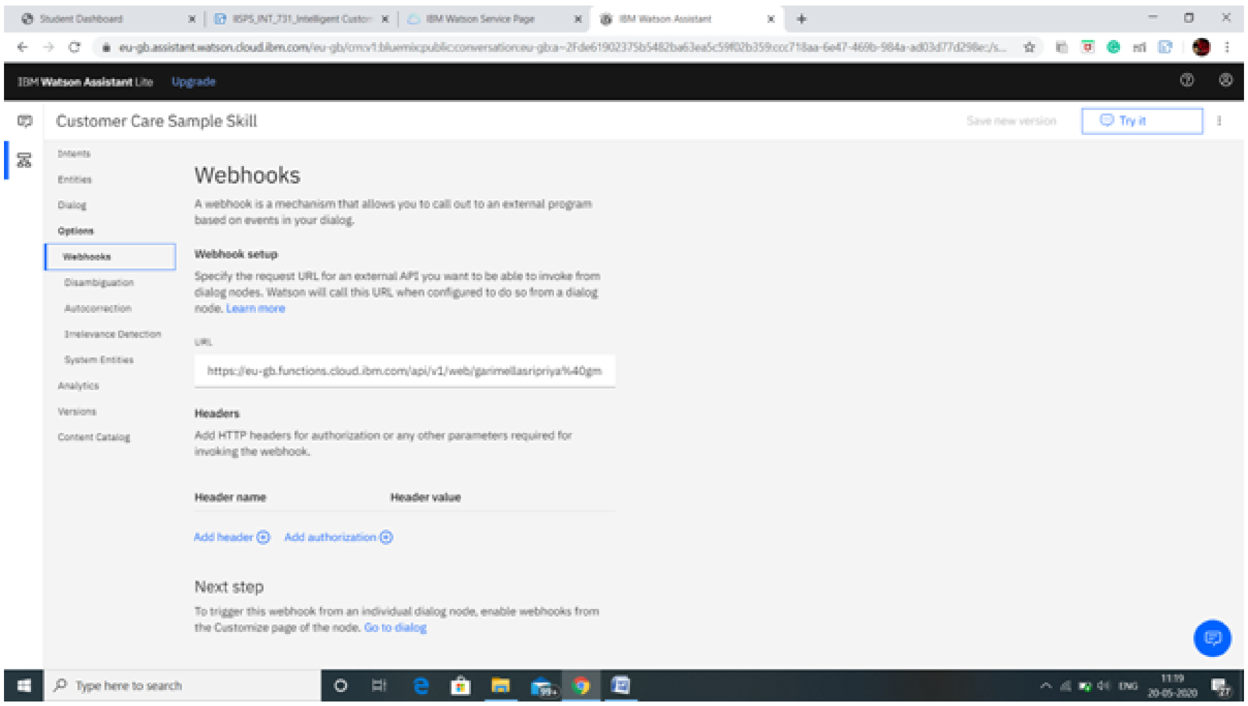
webhook_result_1

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.

Eable 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]:



Webhooks

A webhook is a mechanism that allows you to call out to an external program based on events in your dialog.

Webhook setup

Specify the request URL for an external API you want to be able to invoke from dialog nodes. Watson will call this URL when configured to do so from a dialog node. Learn more

URL: https://eu-gb.functions.cloud.ibm.com/api/v1/web/garimellasripriya%40gm

Headers

Add HTTP headers for authorization or any other parameters required for invoking the webhook.

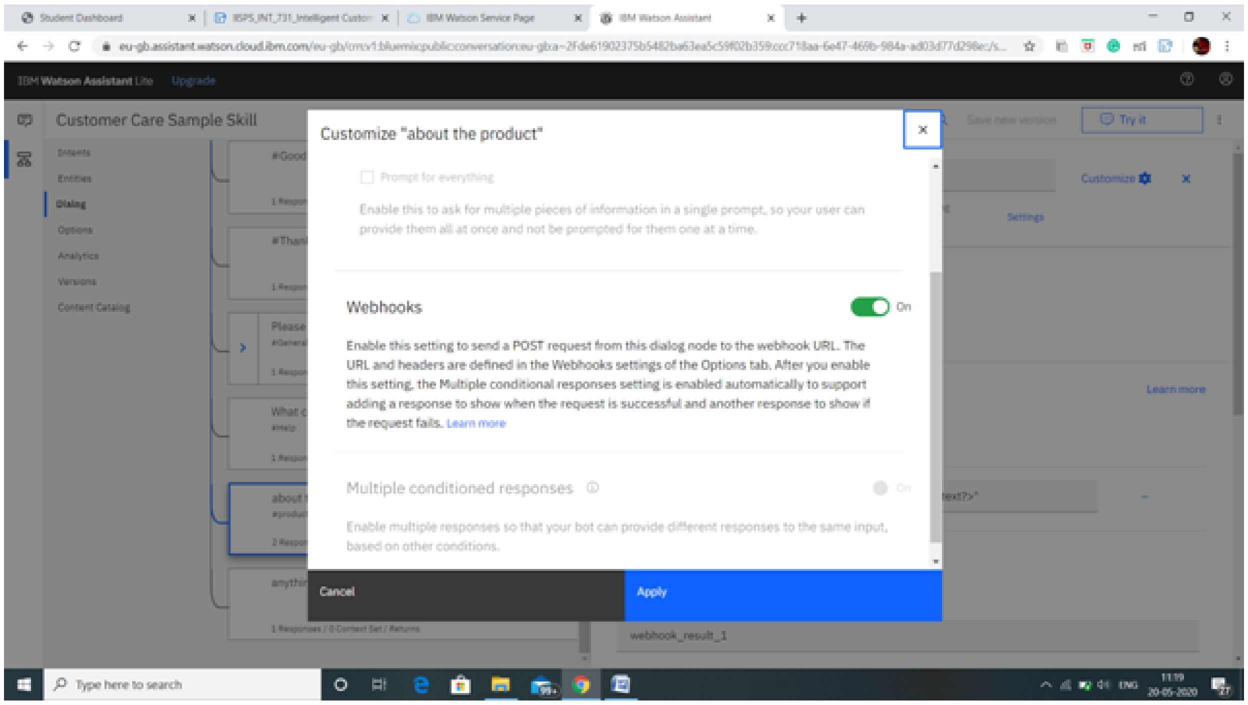
Header name	Header value

Next step

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

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

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:


Customize "about the product"

Prompt for everything

Enable this to ask for multiple pieces of information in a single prompt, so your user can provide them all at once and not be prompted for them one at a time.

Webhooks — On

Enable this setting to send a POST request from this dialog node to the webhook URL. The URL and headers are defined in the Webhooks settings of the Options tab. After you enable this setting, the Multiple conditional responses setting is enabled automatically to support adding a response to show when the request is successful and another response to show if the request fails. Learn more

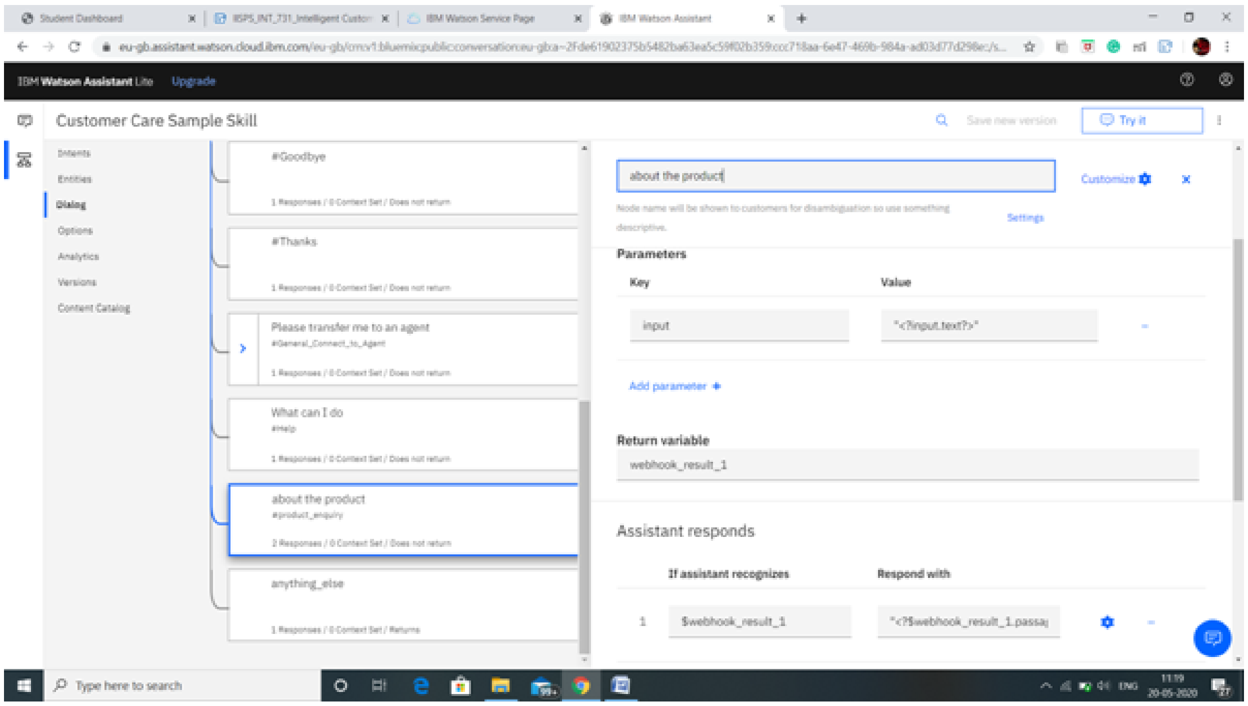
Multiple conditional responses — On

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

Cancel | Apply

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.


Customer Care Sample Skill

Node name: about the product

Parameters

Key	Value
input	"<?input.text?>"

Return variable: webhook_result_1

Assistant responds

	If assistant recognizes	Respond with
1	$webhook_result_1	"<?$webhook_result_1.passa

You 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

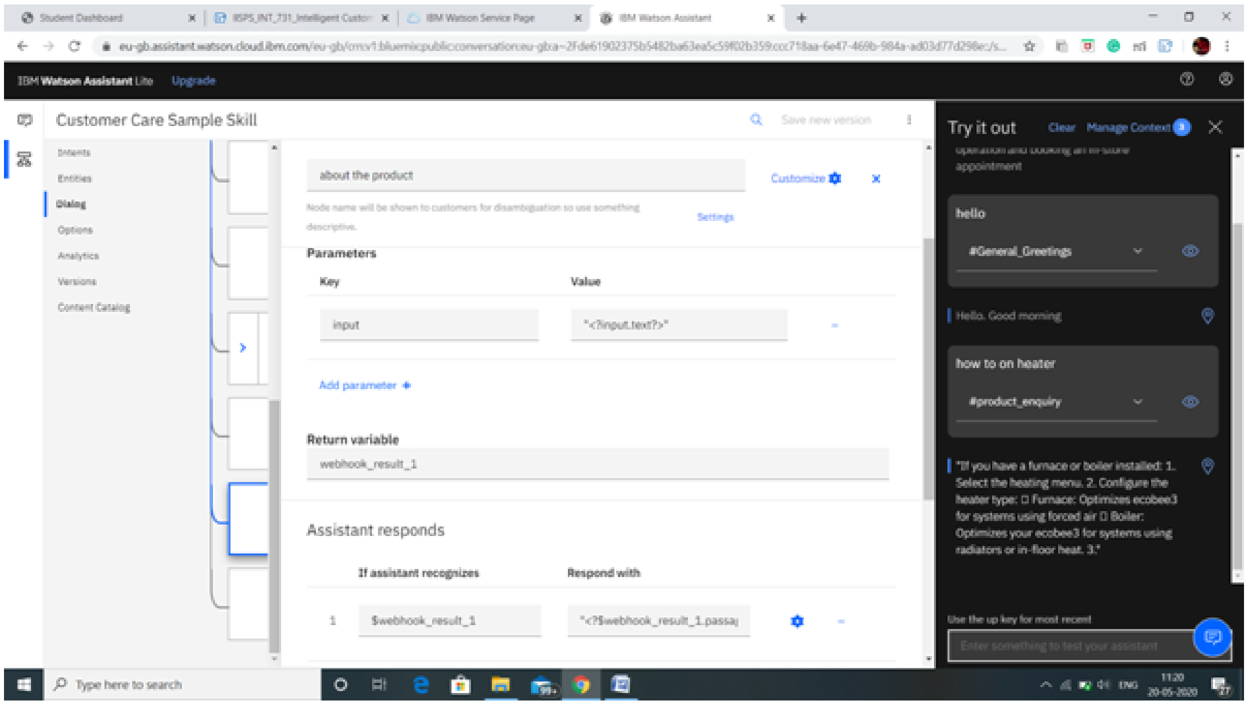
query. Optionally, you can add these responses to aid in

debugging:

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:



about the product

Node name will be shown to customers for disambiguation so use something descriptive.

Parameters

Key	Value
input	"<?input.text?>"

Return variable

webhook_result_1

Assistant responds

	If assistant recognizes	Respond with
1	$webhook_result_1	"<?$webhook_result_1.passa

appointment

hello

#General_Greetings

Hello. Good morning

how to on heater

#product_enquiry

"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."

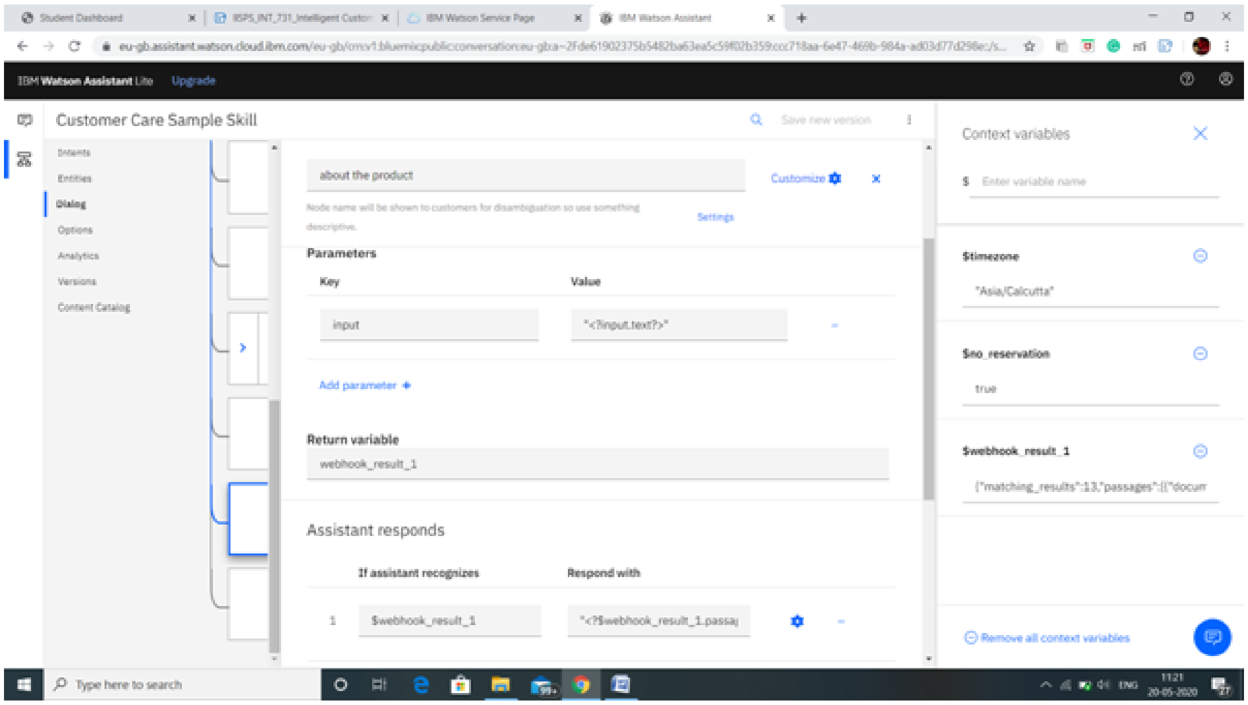
Use the up key for most recent

Enter something to test your assistant

Note that the input "how do I turn on the heater?" has triggered our Ask about product dialog node, which is indicated by the #Product_Information response.

And because we specified that $webhook_result_1.passages be the response, that value is displayed also.

You can also verify that the call was successfully completed by clicking on the Manage Context button at the top right. The response from the Discovery query will be stored in the
$webhook_result_1 variable:



about the product

Node name will be shown to customers for disambiguation so use something descriptive.

Parameters

Key	Value
input	"<?input.text?>"

Return variable
webhook_result_1

Assistant responds

	If assistant recognizes	Respond with
1	$webhook_result_1	"<?$webhook_result_1.passa

Context variables

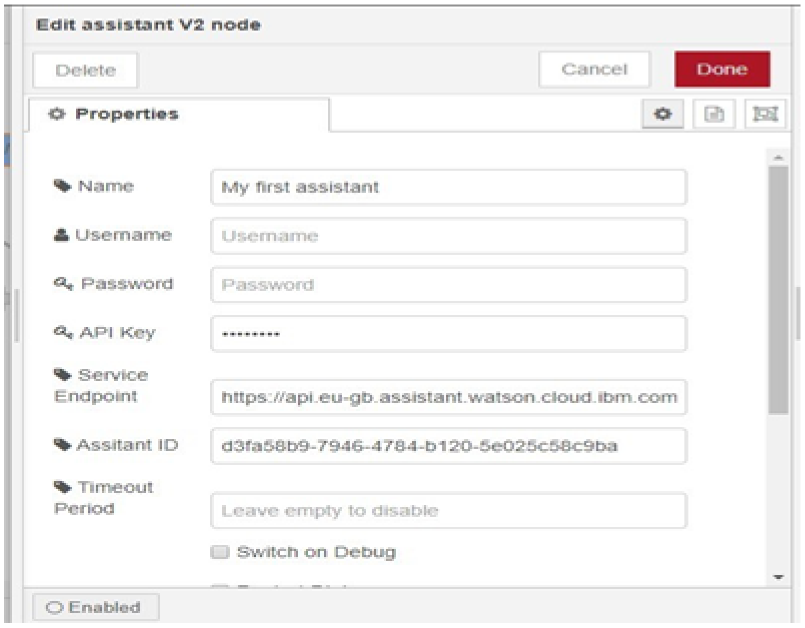
$timezone	"Asia/Calcutta"
$no_reservation	true
$webhook_result_1	{"matching_results":13,"passages":[{"docum

4.Create flow and confidence:

Integration of Watson assistant in Node-RED

1. Double-click on the Watson assistant node

2. Give a name to your node and enter the username, password and workspace id of your Watson assistant service.


Edit assistant V2 node

Delete · Cancel · Done

Properties

Name: My first assistant

Username: Username

Password: Password

API Key: ••••••••••

Service Endpoint: https://api.eu-gb.assistant.watson.cloud.ibm.com

Assitant ID: d3fa58b9-7946-4784-b120-5e025c58c9ba

Timeout Period: Leave empty to disable

Switch on Debug

Enabled

1.After entering all the information click on Done

2.Drag inject node on to the flow from the Input section

3.Drag Debug on to the flow from the output section

4.Double-click on the inject node

5.Select the payload as a string

6.Enter a sample input to be sent to the assistant service and click on done

7.Connect the nodes as shown below and click on Deploy

8.Open Debug window as shown below

9.Click on the button to send input text to the assistant node

10.Observe the output from the assistant service node

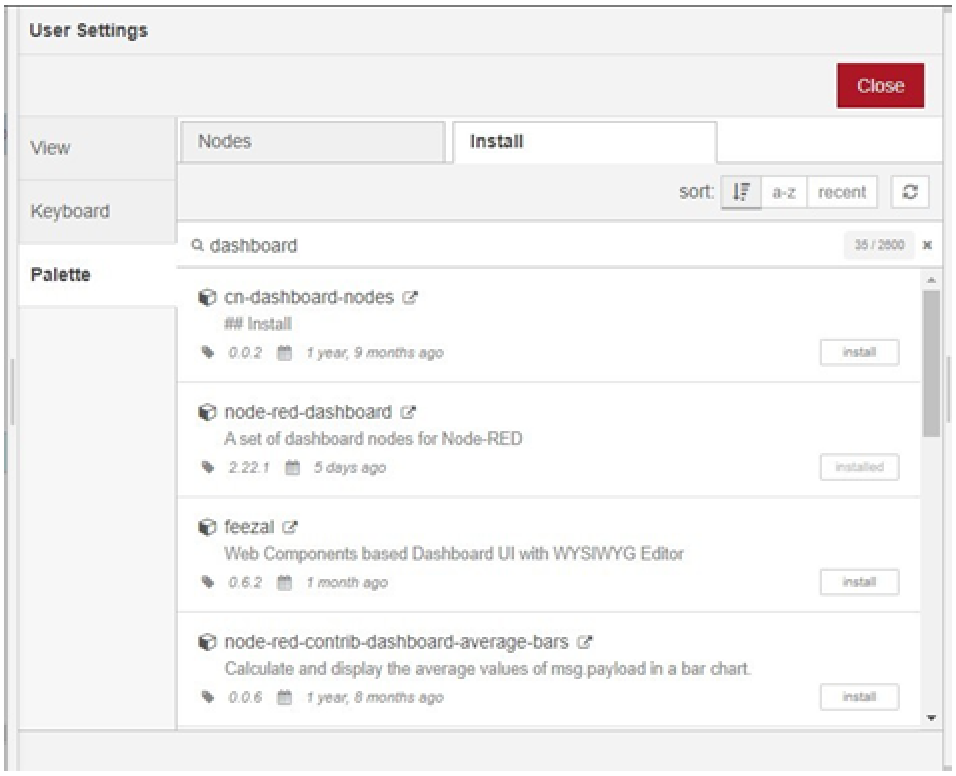
11.The Bot output is located inside “output.text"

12.Drag the function node to parse the JSON data and get the bot response

13.Double click on the function node and enter the JSON parsing code as shown below and click on done

14.Connect the nodes as shown below and click on Deploy

15.Re-inject the flow and observe the parsed output



User Settings

Close

View | Keyboard | Palette

Nodes | Install

sort: a-z recent

dashboard 35 / 2800

cn-dashboard-nodes
Install
0.0.2 1 year, 9 months ago install

node-red-dashboard
A set of dashboard nodes for Node-RED
2.22.1 5 days ago installed

feezal
Web Components based Dashboard UI with WYSIWYG Editor
0.6.2 1 month ago install

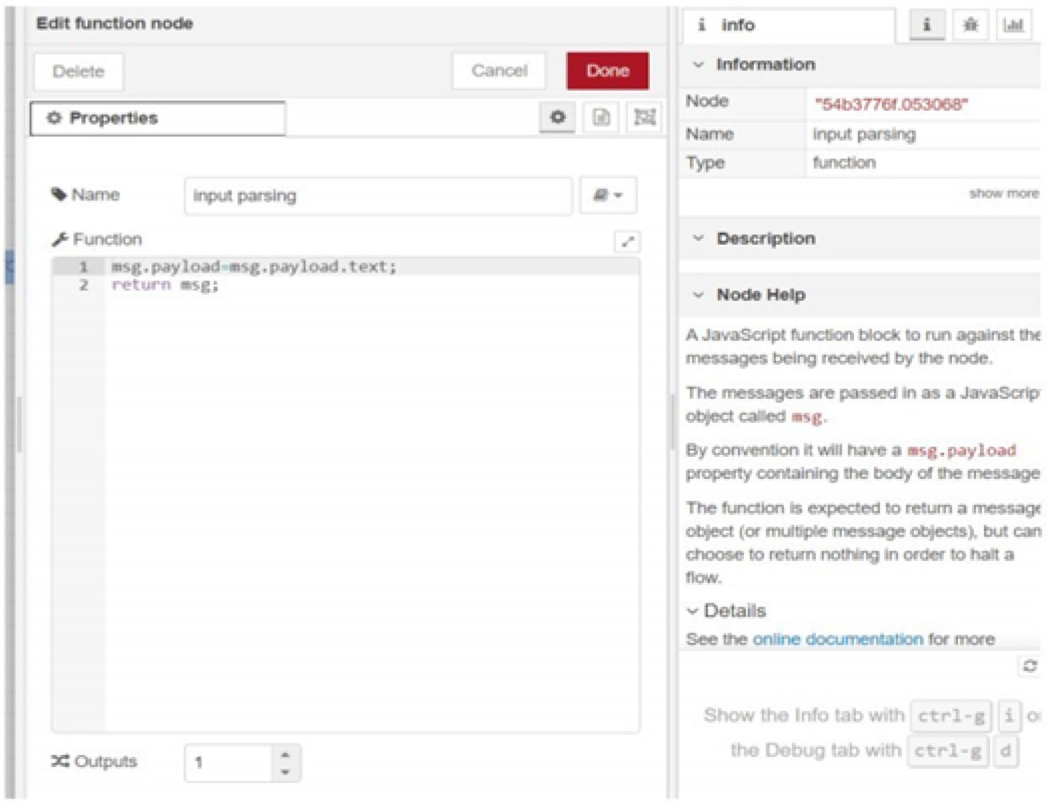
node-red-contrib-dashboard-average-bars
Calculate and display the average values of msg.payload in a bar chart.
0.0.6 1 year, 8 months ago install

1.For creating a web application

UI we need “dashboard“ nodes which should be installed manually.
2.Go to navigation pane and click on manage palette

a. Click on install
b. Search for “node-red-dashboard” and click on install and again click on install on the prompt

c. The following message indicates dashboard nodes are installed, close the manage palette
d. Search for “Form” node and drag on to the flow
e. Doube click on the “form” node to configure
f. Click on the edit button to add the “Group” name and “Tab”name
g. Click on the edit button to add tab name to web application
h. Give sample tab name and click on add do the same thing for the group
i. Give the label as “Enter your input”, Name as “text” and click on Done
j. Drag a function node, double-click on it and enter the input parsing code as shown below



Name: input parsing

Function:

```
msg.payload=msg.payload.text;
return msg;
```

Outputs: 1

Node	"54b3776f.053068"
Name	input parsing
Type	function

A JavaScript function block to run against the messages being received by the node.

The messages are passed in as a JavaScrip object called msg.

By convention it will have a msg.payload property containing the body of the message

The function is expected to return a message object (or multiple message objects), but can choose to return nothing in order to halt a flow.

Details

See the online documentation for more

Show the Info tab with ctrl-g i o

the Debug tab with ctrl-g d

a. Click on done
b. Connect the form output to the input of the function node and output of the function to input of assistant node
c. Search for “text” node from the “dashboard”section

d. Drag two “text” nodes on to the flow

e. Double click on the first text node, change the label as “You” and click on Done

f. Double click on the second text node, change the label as “Bot” and click on Done

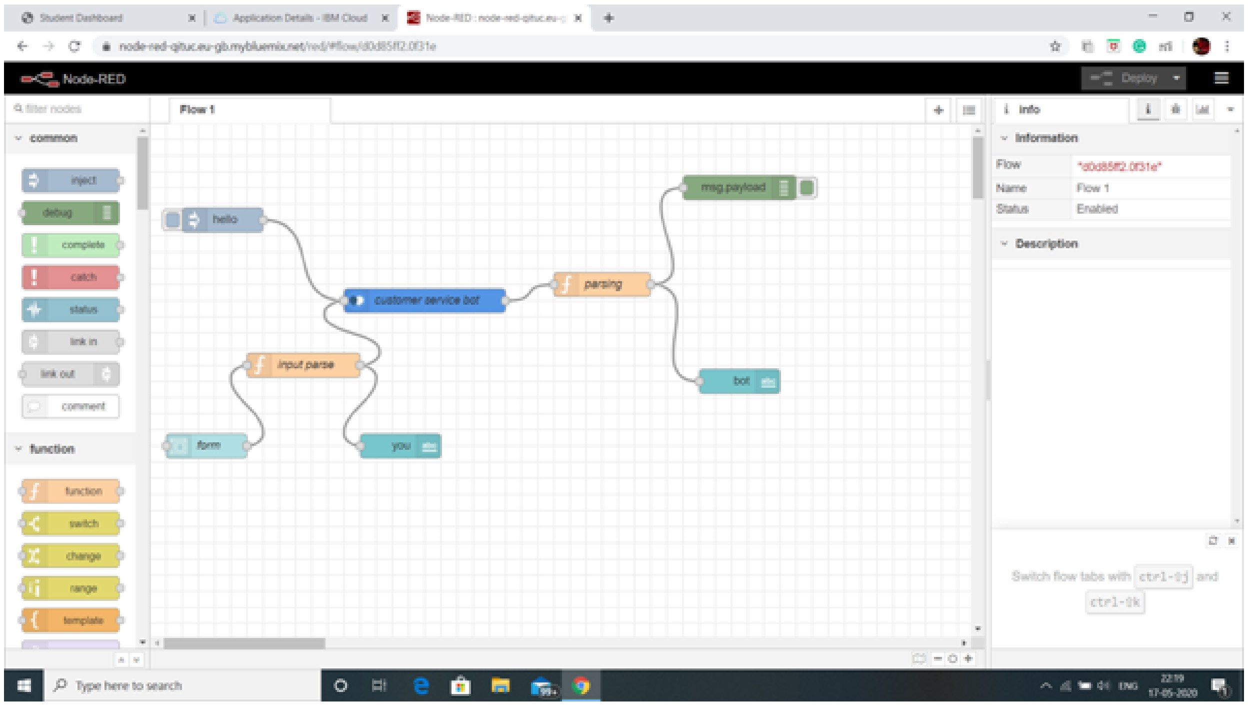
g. Connect the output of “input parsing” function node to “
You” text node and output of “Parsing” function node to the
input of “Bot” text node

h. Click on Deploy

5. FLOW CHART

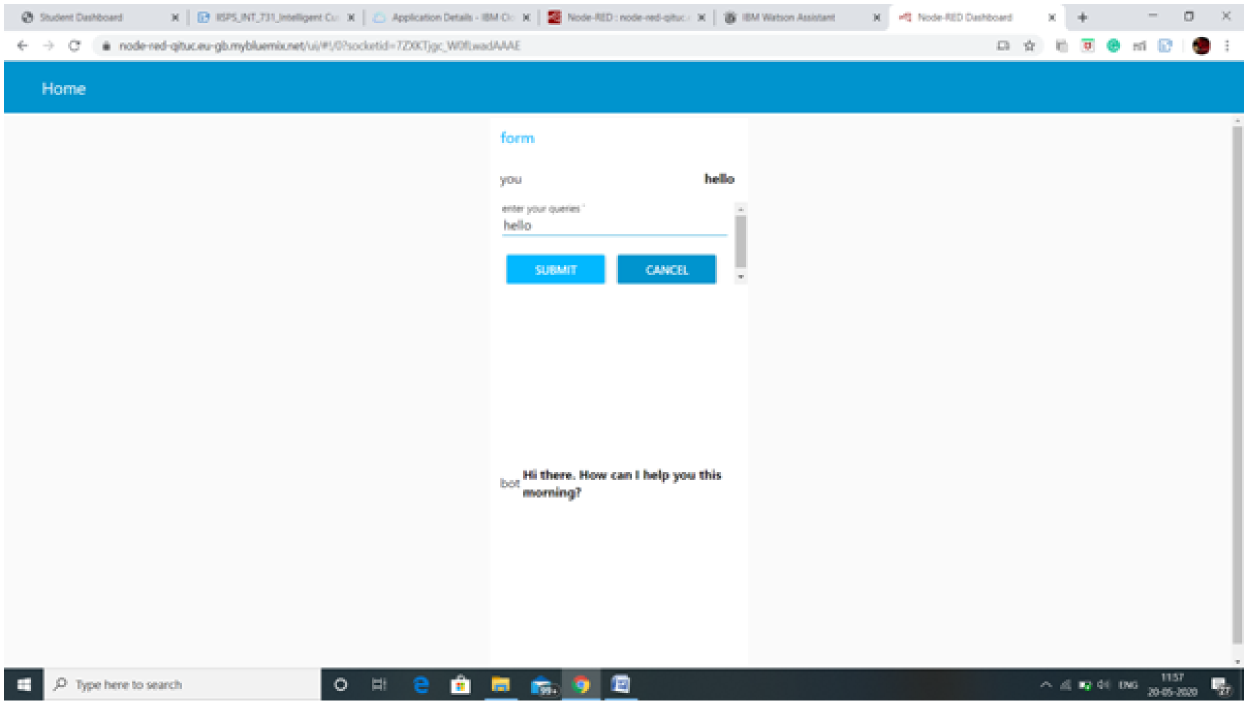
At first go to manage pallet and install dashboard. Now,Create the flow with the help following node:

1. Inject
2. Assistant
3. Debug
4. Function
5. Ui_Form
6. Ui_Text



6.RESULTS

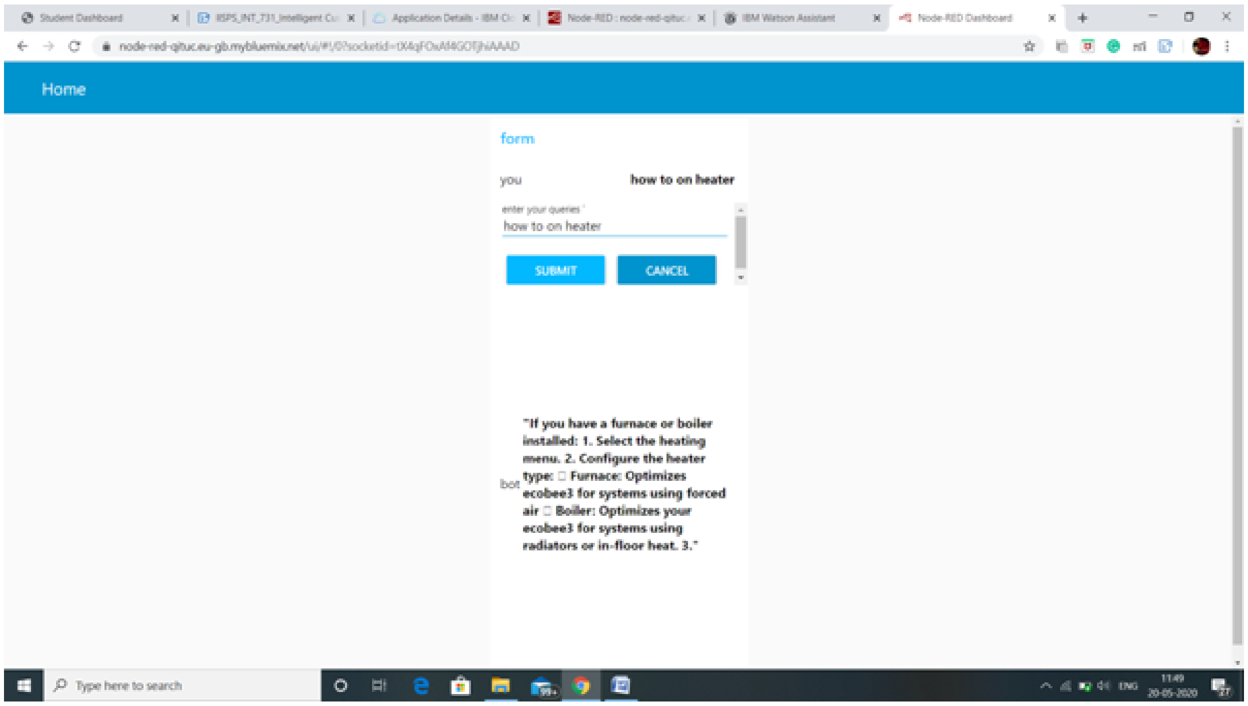
Finally our Node-RED dash board integrates all the components and displayed in the Dashboard UI by typing https://node-red-qituc.eu-gb.mybluemix.net/ui/ in browser



you hello

hello

bot Hi there. How can I help you this morning?



you how to on heater

enter your queries *
how to on heater

bot "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."

7.ADVANTAGES &DISADVANTAGES

Advantages:

1. Campanies can deploy chatbots to rectify simple and general human queries.
2. Reduces man power.
3. Cost efficient.
4. No need to divert calls to customer agent and customer agent can look on other works.
5. can ask queries from home.

Disadvantages:

1. Some times chat bot can mislead customers

2.Giving same answer for different sentiments.

3.Some times cannot connect to customer sentiments and intentions.

8.APPLICATIONS

a. It can deploy in popular social media applications like facebook,slack,telegram.
b. Chatbot can deploy any website to clarify basic doubts of viewers.
c. chat bot is the easy way of communication
d. It can be used for various applications
e. it is used to know about the product
f. it can also be used in knowing basic information

9.CONCLUSION

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

The conclusion for this is that it is the easy way of communication where we can resolve the customer or the consumer queries without the help of the person we can do it virtually.

So customer can access 24 hours for the queries what he have and get resolved his problem simply by sitting in his place and asking the bot so questions.

If the bot does not understand it just give you the number of the customer care where you can contact them for future queries.

10.FUTURESCOPE

We can include Watson studio text to speech and speech to text services to access the chat bot hands free. This is one of the future scope of this project.

11. BIBILOGRAPHY

we used some IBM guidelines etc to resolve our problems.

APPENDIX

Sour Code

1. **CloudFunction(Node.js)**

/**

*

a. @param {object}para ms

b. @param {string}para ms.am_api key

c. @param {string}para ms.url

d. @param {string}params.username

e. @param {string}para ms.password

f. @param {string}params.environment_id

g. @param {string}params.collection_id

h. @param {string}params.configuration_id

i. @param {string}params.input

*

j. @return{object}

*

*/

const assert = require('assert');

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

/**

*

k. main() will be run when you invoke thisaction

*

l. @paramCloudFunctionsactionsacceptasingleparameter,whichmustbeaJSONobject.

*

m. @return The output of this action, which must be a JSONobject.

*

*/

```
 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': '2020-05-09'

 });

 }

 else {

discovery = new DiscoveryV1({ 'username': params.username, 'password':

 params.password, 'url': params.url,

 'version': '2020-05-11'

 });

 }



 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);

      });

    });

  }
```

2.Node Red(flow.json)

```
[
    {
        "id": "d0d85ff2.0f31e",
        "type": "tab",
        "label": "Flow 1",
        "disabled": false,
        "info": ""
    },
    {
        "id": "81a35420.f26278",
        "type": "ui_form",
    "z": "d0d85ff2.0f31e",
    "name": "form",
    "label": "",
    "group": "eee5eb1.9de6f18",
    "order": 0,
    "width": "6",
    "height": "6",
    "options": [
        {
            "label": "enter your queries",
            "value": "text",
            "type": "text",
            "required": true,
            "rows": null
        }
    ],
    "formValue": {
        "text": ""
    },
    "payload": "",
    "submit": "submit",
    "cancel": "cancel",
    "topic": "",
    "x": 70,
    "y": 400,
    "wires": [
        [
            "2fc109df.9935e6"
        ]
    ]
},
{
    "id": "e4411f54.83b62",
    "type": "ui_text",
    "z": "d0d85ff2.0f31e",
    "group": "eee5eb1.9de6f18",
    "order": 1,
    "width": 0,
    "height": 0,
    "name": "",
    "label": "you",
    "format": "{{msg.payload}}",
    "layout": "row-spread",
    "x": 310,
    "y": 400,
    "wires": []
},
{
    "id": "8ea21b5b.9dfc78",
    "type": "inject",
    "z": "d0d85ff2.0f31e",
    "name": "",
    "topic": "",
    "payload": "hello",
    "payloadType": "str",
    "repeat": "",
    "crontab": "",
    "once": false,
    "onceDelay": 0.1,
    "x": 90,
    "y": 120,
    "wires": [
        [
            "3464ad27.b91b72"
        ]
    ]
},
{
    "id": "2fc109df.9935e6",
    "type": "function",
    "z": "d0d85ff2.0f31e",
    "name": "input parse",
    "func": "msg.payload=msg.payload.text;\nreturn msg;",
    "outputs": 1,
        "noerr": 0,
        "x": 190,
        "y": 300,
        "wires": [
            [
                "3464ad27.b91b72",
                "e4411f54.83b62"
            ]
        ]
    },
    {
        "id": "971684ee.01b3d8",
        "type": "function",
        "z": "d0d85ff2.0f31e",
        "name": "parsing",
        "func": "msg.payload=msg.payload.output.text[0];\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 560,
        "y": 200,
        "wires": [
            [
                "ae891913.e5e228",
                "de71deca.ca4b1"
            ]
        ]
    },
    {
        "id": "ae891913.e5e228",
        "type": "ui_text",
        "z": "d0d85ff2.0f31e",
        "group": "eee5eb1.9de6f18",
        "order": 0,
        "width": "6",
        "height": "9",
        "name": "",
        "label": "bot",
        "format": "{{msg.payload}}",
        "layout": "row-left",
        "x": 730,
        "y": 320,
        "wires": []
    },
    {
        "id": "de71deca.ca4b1",
        "type": "debug",
        "z": "d0d85ff2.0f31e",
        "name": "",
        "active": true,
        "tosidebar": true,
        "console": false,
        "tostatus": false,
        "complete": "payload",
        "targetType": "msg",
        "x": 730,
        "y": 80,
        "wires": []
    },
    {
        "id": "3464ad27.b91b72",
        "type": "watson-conversation-v1",
        "z": "d0d85ff2.0f31e",
        "name": "customer service bot",
        "workspaceid": "2b5beca9-21e4-4391-b502-309530f12a63",
        "multiuser": false,
        "context": true,
        "empty-payload": false,
        "service-endpoint": "",
        "timeout": "",
        "optout-learning": false,
        "x": 340,
        "y": 220,
        "wires": [
            [
                "971684ee.01b3d8"
            ]
        ]
    },
    {
        "id": "eee5eb1.9de6f18",
        "type": "ui_group",
        "z": "",
        "name": "form",
        "tab": "59bfe54e.277f4c",
        "order": 2,
        "disp": true,
        "width": "6",
        "collapse": false
    },
    {
        "id": "59bfe54e.277f4c",
        "type": "ui_tab",
        "z": "",
        "name": "Home",
        "icon": "dashboard",
        "disabled": false,
        "hidden": false
    }
]


[
    {
        "id": "d0d85ff2.0f31e",
        "type": "tab",
        "label": "Flow 1",
        "disabled": false,
        "info": ""
    },
    {
        "id": "eee5eb1.9de6f18",
        "type": "ui_group",
        "z": "",
        "name": "form",
        "tab": "59bfe54e.277f4c",
        "order": 2,
        "disp": true,
        "width": "6",
        "collapse": false
    },
    {
        "id": "59bfe54e.277f4c",
        "type": "ui_tab",
        "z": "",
        "name": "Home",
        "icon": "dashboard",
        "disabled": false,
        "hidden": false
    },
    {
        "id": "22f35ab6.d3ff06",
        "type": "ui_group",
        "z": "",
        "name": "Default",
        "tab": "59bfe54e.277f4c",
        "order": 1,
        "disp": true,
        "width": "6",
        "collapse": false
    },
    {
        "id": "3b2ea5a6.df5fca",
        "type": "ui_base",
        "theme": {
            "name": "theme-light",
            "lightTheme": {
                "default": "#0094CE",
                "baseColor": "#0094CE",
        "baseFont": "-apple-system,BlinkMacSystemFont,Segoe
UI,Roboto,Oxygen-Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif",
        "edited": true,
        "reset": false
      },
      "darkTheme": {
        "default": "#097479",
        "baseColor": "#097479",
        "baseFont": "-apple-system,BlinkMacSystemFont,Segoe
UI,Roboto,Oxygen-Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif",
        "edited": false
      },
      "customTheme": {
        "name": "Untitled Theme 1",
        "default": "#4B7930",
        "baseColor": "#4B7930",
        "baseFont": "-apple-system,BlinkMacSystemFont,Segoe
UI,Roboto,Oxygen-Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif"
      },
      "themeState": {
        "base-color": {
          "default": "#0094CE",
          "value": "#0094CE",
          "edited": false
        },
        "page-titlebar-backgroundColor": {
          "value": "#0094CE",
          "edited": false
        },
        "page-backgroundColor": {
          "value": "#fafafa",
          "edited": false
        },
        "page-sidebar-backgroundColor": {
          "value": "#ffffff",
          "edited": false
        },
            "group-textColor": {
                "value": "#1bbfff",
                "edited": false
            },
            "group-borderColor": {
                "value": "#ffffff",
                "edited": false
            },
            "group-backgroundColor": {
                "value": "#ffffff",
                "edited": false
            },
            "widget-textColor": {
                "value": "#111111",
                "edited": false
            },
            "widget-backgroundColor": {
                "value": "#0094ce",
                "edited": false
            },
            "widget-borderColor": {
                "value": "#ffffff",
                "edited": false
            },
            "base-font": {
                "value": "-apple-system,BlinkMacSystemFont,Segoe
UI,Roboto,Oxygen-Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif"
            }
        },
        "angularTheme": {
            "primary": "indigo",
            "accents": "blue",
            "warn": "red",
            "background": "grey"
        }
    },
    "site": {
            "name": "Node-RED Dashboard",
            "hideToolbar": "false",
            "allowSwipe": "false",
            "lockMenu": "false",
            "allowTempTheme": "true",
            "dateFormat": "DD/MM/YYYY",
            "sizes": {
                "sx": 48,
                "sy": 48,
                "gx": 6,
                "gy": 6,
                "cx": 6,
                "cy": 6,
                "px": 0,
                "py": 0
            }
        }
    },
    {
        "id": "81a35420.f26278",
        "type": "ui_form",
        "z": "d0d85ff2.0f31e",
        "name": "form",
        "label": "",
        "group": "eee5eb1.9de6f18",
        "order": 0,
        "width": "6",
        "height": "6",
        "options": [
            {
                "label": "enter your queries",
                "value": "text",
                "type": "text",
                "required": true,
                "rows": null
            }
        ],
        "formValue": {
            "text": ""
        },
        "payload": "",
        "submit": "submit",
        "cancel": "cancel",
        "topic": "",
        "x": 70,
        "y": 400,
        "wires": [
            [
                "2fc109df.9935e6"
            ]
        ]
    },
    {
        "id": "e4411f54.83b62",
        "type": "ui_text",
        "z": "d0d85ff2.0f31e",
        "group": "eee5eb1.9de6f18",
        "order": 1,
        "width": 0,
        "height": 0,
        "name": "",
        "label": "you",
        "format": "{{msg.payload}}",
        "layout": "row-spread",
        "x": 310,
        "y": 400,
        "wires": []
    },
    {
        "id": "8ea21b5b.9dfc78",
        "type": "inject",
        "z": "d0d85ff2.0f31e",
        "name": "",
        "topic": "",
        "payload": "hello",
        "payloadType": "str",
        "repeat": "",
        "crontab": "",
        "once": false,
        "onceDelay": 0.1,
        "x": 90,
        "y": 120,
        "wires": [
            [
                "3464ad27.b91b72"
            ]
        ]
    },
    {
        "id": "2fc109df.9935e6",
        "type": "function",
        "z": "d0d85ff2.0f31e",
        "name": "input parse",
        "func": "msg.payload=msg.payload.text;\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 190,
        "y": 300,
        "wires": [
            [
                "3464ad27.b91b72",
                "e4411f54.83b62"
            ]
        ]
    },
    {
        "id": "971684ee.01b3d8",
        "type": "function",
        "z": "d0d85ff2.0f31e",
        "name": "parsing",
        "func": "msg.payload=msg.payload.output.text[0];\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 560,
        "y": 200,
        "wires": [
            [
                "ae891913.e5e228",
                "de71deca.ca4b1"
            ]
        ]
    },
    {
        "id": "ae891913.e5e228",
        "type": "ui_text",
        "z": "d0d85ff2.0f31e",
        "group": "eee5eb1.9de6f18",
        "order": 0,
        "width": "6",
        "height": "9",
        "name": "",
        "label": "bot",
        "format": "{{msg.payload}}",
        "layout": "row-left",
        "x": 730,
        "y": 320,
        "wires": []
    },
    {
        "id": "de71deca.ca4b1",
        "type": "debug",
        "z": "d0d85ff2.0f31e",
        "name": "",
        "active": true,
        "tosidebar": true,
        "console": false,
        "tostatus": false,
        "complete": "payload",
        "targetType": "msg",
        "x": 730,
        "y": 80,
        "wires": []
    },
    {
        "id": "3464ad27.b91b72",
        "type": "watson-conversation-v1",
        "z": "d0d85ff2.0f31e",
        "name": "customer service bot",
        "workspaceid": "2b5beca9-21e4-4391-b502-309530f12a63",
        "multiuser": false,
        "context": true,
        "empty-payload": false,
        "service-endpoint": "",
        "timeout": "",
        "optout-learning": false,
        "x": 340,
        "y": 220,
        "wires": [
            [
                "971684ee.01b3d8"
            ]
        ]
    }
]
```

References:

1. https://www.ibm.com/cloud/architecture/tutorials/cognitive_discovery

2. https//cloud.ibm.com/docs/assistant?topic=assistant-getting-started
3. h
ttps//developer.ibm.com/recipes/tutorials/how-to-create-a-watson-chatbot-on-nodered
/
4. https//www.iotgyan.com/learning-resource/integration-of-watson-assistant-to-node-red
5. https//github.com/IBM/watson-discovery-sdu-with-assistant

https//www.youtube.com/watch?v=Jpr3wVH3FVA