

# **PROJECT REPORT**

## **Intelligent Customer Help Desk with Smart Document Understanding – SB51111**

Name: NAVEEN RAJI EAPEN

E-mail: naveen.re1721@saintgits.org

Category: Artificial Intelligence Internship at SmartBridge

Application ID: SPS\_APL\_20200003657

Project ID: SPS\_PRO\_99

# INDEX

## **1. INTRODUCTION**

1.1 Overview

1.2 Purpose

## **2. LITERATURE SURVEY**

2.1 Existing problem

2.2 Proposed solution

## **3. THEORITICAL ANALYSIS**

3.1 Block diagram

3.2 Software designing

## **4. EXPERIMENTAL PROCEDURE**

## **5. FLOWCHART**

## **6. RESULT**

## **7. ADVANTAGES AND DISADVANTAGES**

## **8. APPLICATIONS**

## **9. CONCLUSION**

## **10. FUTURE SCOPE**

## **11. BIBLIOGRAPHY**

## **12. REFERENCE**

# 1. INTRODUCTION

## 1.1 Overview:

### 1) Project Summary

A customer care chatbot can answer simple questions, such as store locations and hours, directions, and perhaps even making appointments. In this project, the queries will be handled in a better way. If the customer's question is about the operation of a device, the application shall pass the question onto Watson Discovery Service and we can handle the queries in a better way. We will build a chatbot that uses various Watson AI Services like Watson Discovery, Watson Assistant, Watson Cloud functions and Node-Red and deliver an effective user friendly Web User Interface.

### 2) Project Requirements

- i) Customers care dialog skill in Watson assistant.
- ii) Smart document understanding to build an enhanced Watson Discovery Collection.
- iii) IBM cloud functions.
- iv) Web application with integration to all these services and deploy the same on IBM cloud platform.

### 3) Functional Requirements

The system shall be able to identify all kinds of queries from the user beyond the scope by Smart Document Understanding feature of Watson Discovery to train it on what text in the owner's manual is important and what is not.

### 4) Technical Requirements

- i) IBM Cloud services and functions.
- ii) Python.

iii) IBM Watson services.

iv) Node-Red.

## 5) Software Requirements

i) IBM Cloud functions web action that allows Watson assistant to post queries to Watson discovery.

ii) Build a web application with integration to all these services and deploy the same on IBM Cloud platform.

## 6) Project Deliverables

A web application based intelligent chatbot using AI.

## **1.2 Purpose:**

The typical customer care chat bot can answer simple questions, such as store locations and hours, directions, and 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 is not valid or offer to speak to a real person. In this project, there will be an another option. If the customer question is about the operation of a device, the application shall pass the question onto Watson Discovery Service, which has been pre-loaded with the device's owner's manual. So now, instead of "Would you like to speak to a customer representative?", we can return relevant sections of the owner's manual to help solve our customers' problems. To take it a step further, the project shall use the Smart Document Understanding feature of Watson Discovery to train it on what text in the owner's manual is important and what is not. This will improve the answers returned from the queries.

### **1.2.1 Scope of Work:**

1. Create a customer care dialog skill in Watson Assistant
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 Watson Discovery.
4. Build a web application with integration to all these services and deploy the same on IBM Cloud Platform.

## **2. LITERATURE SURVEY**

### **2.1 Existing Problem:**

Generally, chatbots are used for receiving queries from users and getting general responses. If the question is out of the predetermined question set of the chatbot, response from the chatbot will be to try again, or that it doesn't understand, or ask to repeat the question again or directs the customer to the customer service agent. But a good customer chatbot should minimize involvement of the customer service agent to chat with the customer to clarify his/her doubts. So to achieve this we should incorporate a virtual agent within the chatbot so that it will take care of the real involvement of the customer service agent and customer can clarify his doubts with the chatbot.

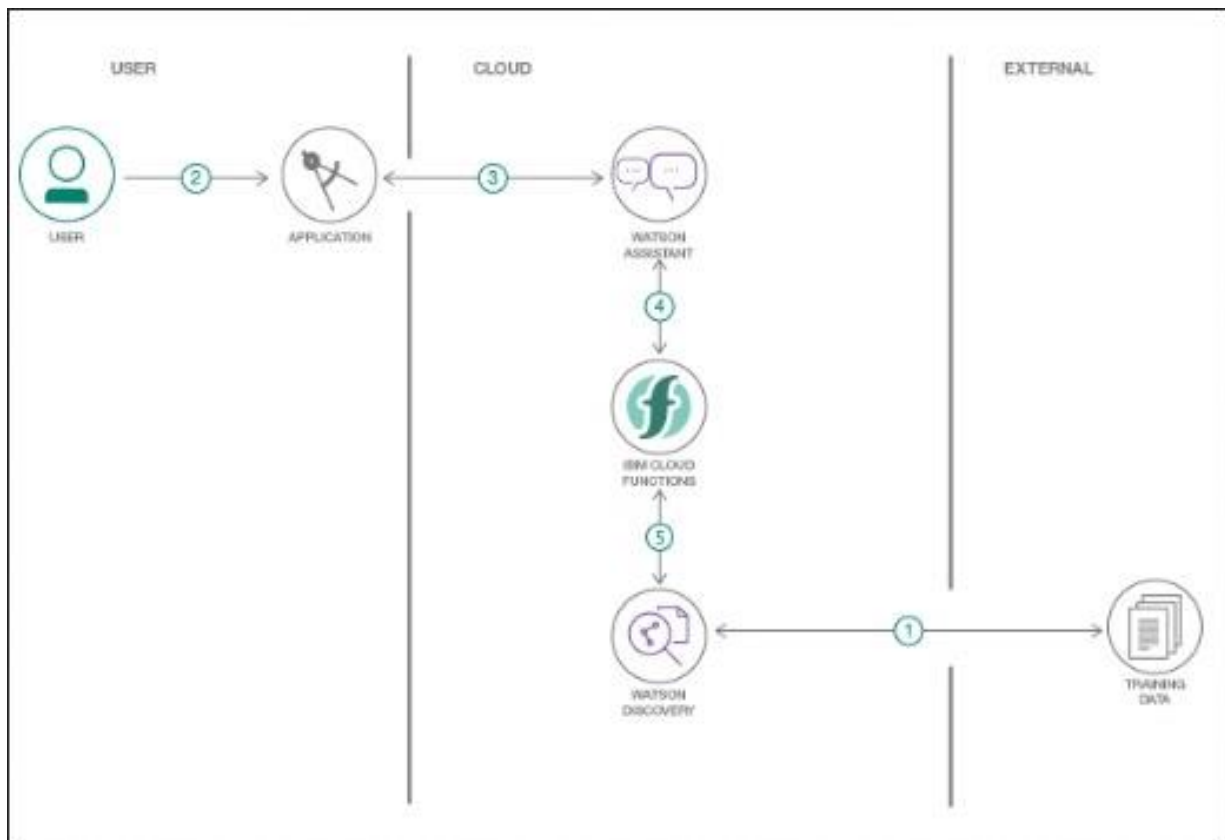
### **2.2 Proposed Solution:**

For the above problem to be solved, we have to incorporate a virtual agent in chatbot so that it can understand the queries that are posted by the customer. The virtual agent should be trained on some modules of records based on the company background so that it can answer the queries related to the product or related to the company. In this project, I have used Watson Discovery Service to achieve the above solution, along with Watson Assistant and built a User-Interface using Node-RED.

# 3. THEORITICAL ANALYSIS

## 3.1 Block/Flow Diagram:

The following flow is the basic working flow of the project.



## 3.2 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 PROCEDURE

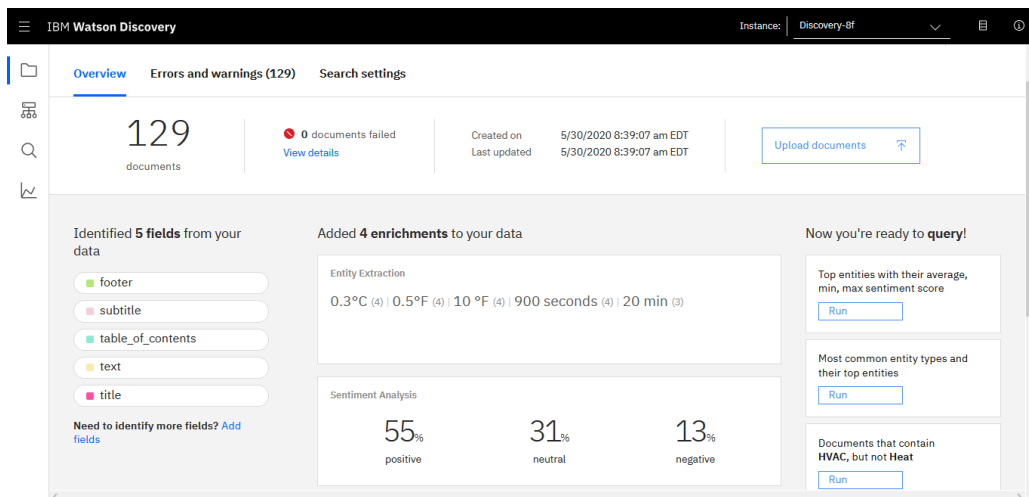
## 1. Create IBM Cloud services.

Create the following services:

- Watson Discovery
- Watson Assistant
- Node Red

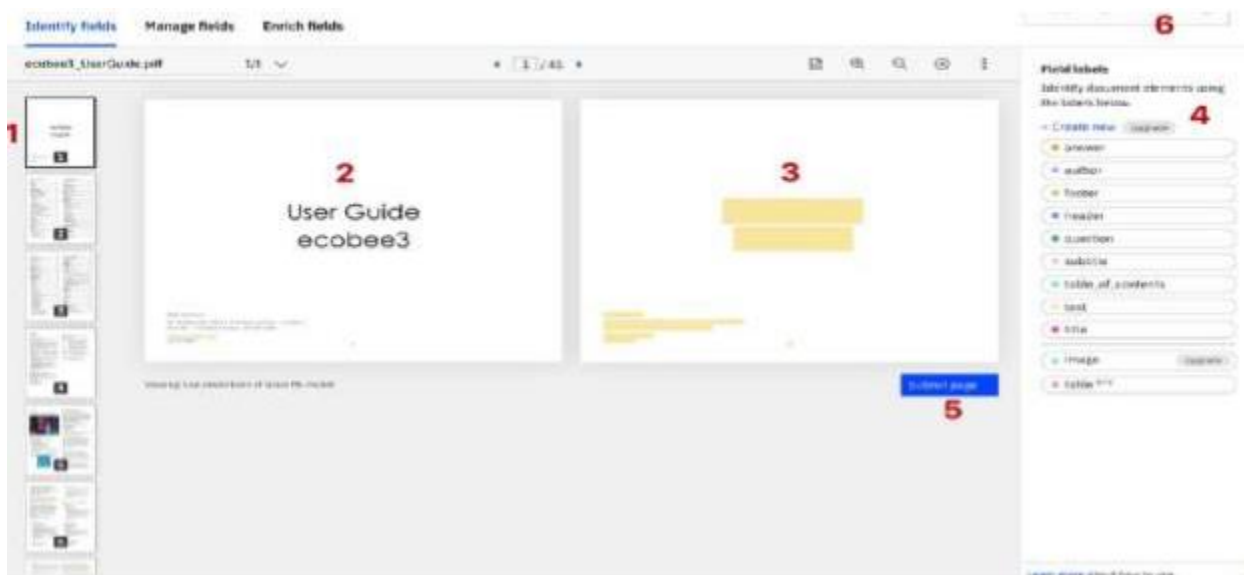
## 2. Configure Watson Discovery

- Import the document
- Launch the Watson Discovery tool and create a new data collection by selecting the Upload your own data option.
- Give the data collection a unique name.
- When prompted, select and upload the Ecobee3 user guide in pdf format.
- The Ecobee3 is a popular residential thermostat that has a Wi-Fi 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.
- Enter queries related to the operation of the thermostat and view the results. 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 the pages in the document so that Discovery can learn what text is important, and what text can be ignored. Follow the following instructions to achieve this task:

- [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.



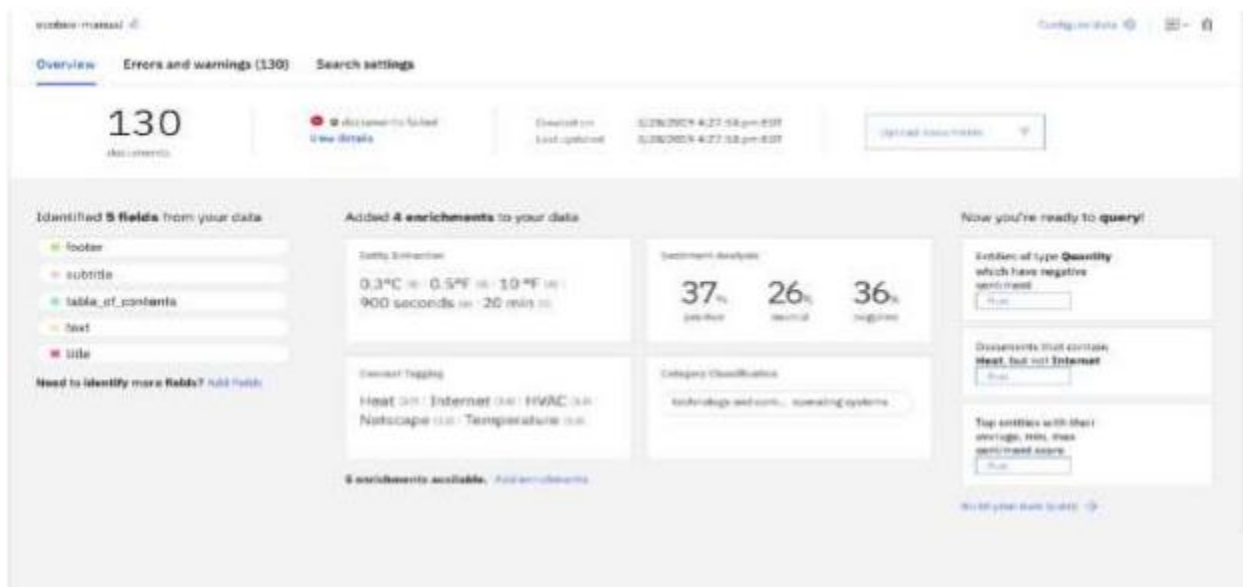


- As you go through the annotations one page at a time, Discovery is learning and should start automatically updating the upcoming pages. Once you get to a page that is already correctly annotated, you can stop, or simply click Submit [5] to acknowledge it is correct. The more pages you annotate, the 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 as table\_of\_contents
  - All headers and sub-headers (typed in light green text) as a subtitle
  - All page numbers as footers
  - All warranty and licensing information (located in the last few pages) as a footer
  - All other text should be marked as text.
- Once you click the Apply changes to collection button [6], you will be asked to reload the document. Choose the same owner's manual .pdf document as before.
- Next, click on the Manage fields [1] tab.



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

- Click [4] to submit your changes.
- Once again, you will be asked to reload the document.
- Now, because of splitting the document apart, your collection will look very different. The results to the queries related to the product are very accurate after annotation of the document.



### 3. Create IBM Cloud Functions action

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

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



- From the Functions main panel, click on the Actions tab. Then click on Create.
- From the Create panel, select the Create Action option.
- On the Create Action panel, provide a unique Action Name [1], keep the default package [2], and select the Node.js 10 [3] runtime. Click the Create button [4] to create the action.
- Once your action is created, click on the Code tab [1]:
  - In the code editor window [2], cut and paste in the code from the cloud\_function.js file found in the actions directory of your local repository. The code is pretty straightforward - it simply connects to the Discovery service, makes a query against the collection, then returns the response.

IBM Cloud Search resources and offerings... Catalog Docs Support Manage Richie George's...

Functions / Actions / discovery-function

## discovery-function

Web Action Namespace: richie.gt1721@saintgits.org\_dev(Dallas)

Code Parameters Runtime Endpoints

Connected Triggers Enclosing Sequences Logs

Code Node.js 10 Edit mode - press ESC to exit Invoke with parameters Invoke

```
1 const assert = require('assert');
2 const DiscoveryV1 = require('watson-developer-cloud/discovery/v1');
3
4 function main(params) {
5   return new Promise(function (resolve, reject) {
6     let discovery;
7
8     if (params.iam_apikey) {
9       discovery = new DiscoveryV1({
10         'iam_apikey': params.iam_apikey,
11         'url': params.url,
12         'version': '2019-03-25'
13       });
14     } else {
15       discovery = new DiscoveryV1({
16         'username': params.username,
17         'password': params.password,
18         'url': params.url,
19         'version': '2019-03-25'
20       });
21     }
22
23     discovery.query({
24       'environment_id': params.environment_id,
25       'collection_id': params.collection_id,
26       'natural_language_query': params.input,
27       'passages': true,
28       'count': 3,
29     });
30   });
31 }
```

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

● Select the Parameters tab: Add the following keys:

- url
- environment\_id
- collection\_id
- iam\_apikey

IBM Cloud Search resources and offerings... Catalog Docs Support Manage Richie George's...

Functions / Actions / discovery-function

## discovery-function

Web Action Namespace: richie.gt1721@saintgits.org\_dev(Dallas)

Code Parameters Runtime Endpoints

Connected Triggers Enclosing Sequences Logs

Parameters Add Parameter

Parameter Name	Parameter Value	
url	"https://api.eu-gb.discovery.watson.cloud.ibm.com/instances/adbf09c"	🗑
environment_id	"d4a7ae00-4afb-4f03-a883-17c9ea8e0684"	🗑
collection_id	"093a4f8d-7ab9-4edc-85d0-78a0ccaae6f9"	🗑
iam_apikey	"0-qBztoSdGAJ1pXrMHbwc1mEPcEq4etVIHlKPuE0rA47"	🗑

- 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:
  - Next, go to the Endpoints panel [1]:
  - 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.

## **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.

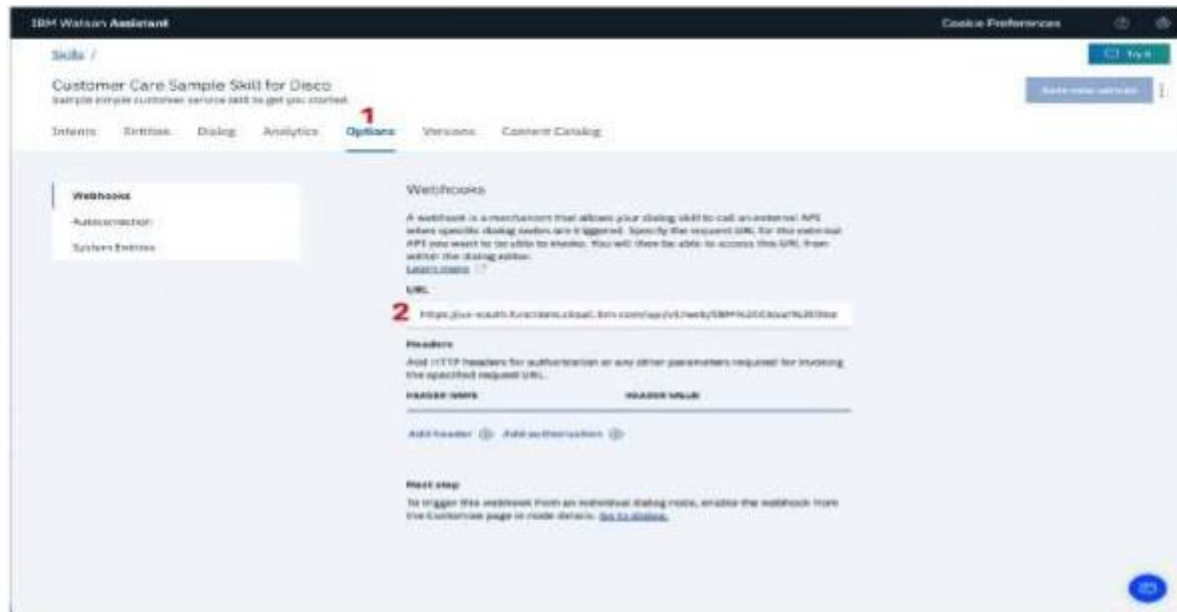
The screenshot shows the 'Create intent' form in Watson Assistant. The intent name is '#Product\_Information'. The description is 'User wants help using the thermostat'. There are three user examples listed: 'How do I access the settings', 'How do I set the time', and 'How do I turn on the heater'. The interface includes buttons for 'Add example' and 'Show recommendations'.

- Click the Create intent button.
- Name the intent #Product\_Information, and at a minimum, enter the following example questions to be associated with it.
- 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].

The screenshot shows the 'Dialog' editor in Watson Assistant. On the left, a list of dialog nodes is shown, including 'Directions and location', 'Have an appointment', 'Transfer to agent', 'Small Talk', 'Ask about product', and 'Anything else'. The 'Ask about product' node is selected. On the right, the configuration for this node is shown. It includes a trigger condition 'If assistant recognizes: #Product\_Information' and a response 'Then respond with: Ask about product'. The response is set to 'sequential'.

- and select the Add node below [3] option.
- Name the node "Ask about product" [1] and assign it our new intent [2]. This means that if Watson Assistant recognizes a user input such as "how do I set the time?", it will direct the conversation to this node.
- Enable Webhook from Assistant

- Set up access to our WebHook for the IBM Cloud Functions action you created in Step #4 (Create IBM Clouds Function).







- Select the Options tab [1]:
  - 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:
  - Click Apply.
  - The dialog node should have a Return variable [1] set automatically to \$webhook\_result\_1.
  - This is the variable name you can use to access the result from the Discovery service query.
- You will also need to pass in the users question via the parameter input [2]. The key needs to be set to the value: "<?input.text?>"
  - If you fail to do this, Discovery will return results based on a blank query.
- Optionally, you can add these responses to aid in debugging:


Return variable

`$webhook_result_1`

---

Then respond with

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

[Add response](#) 

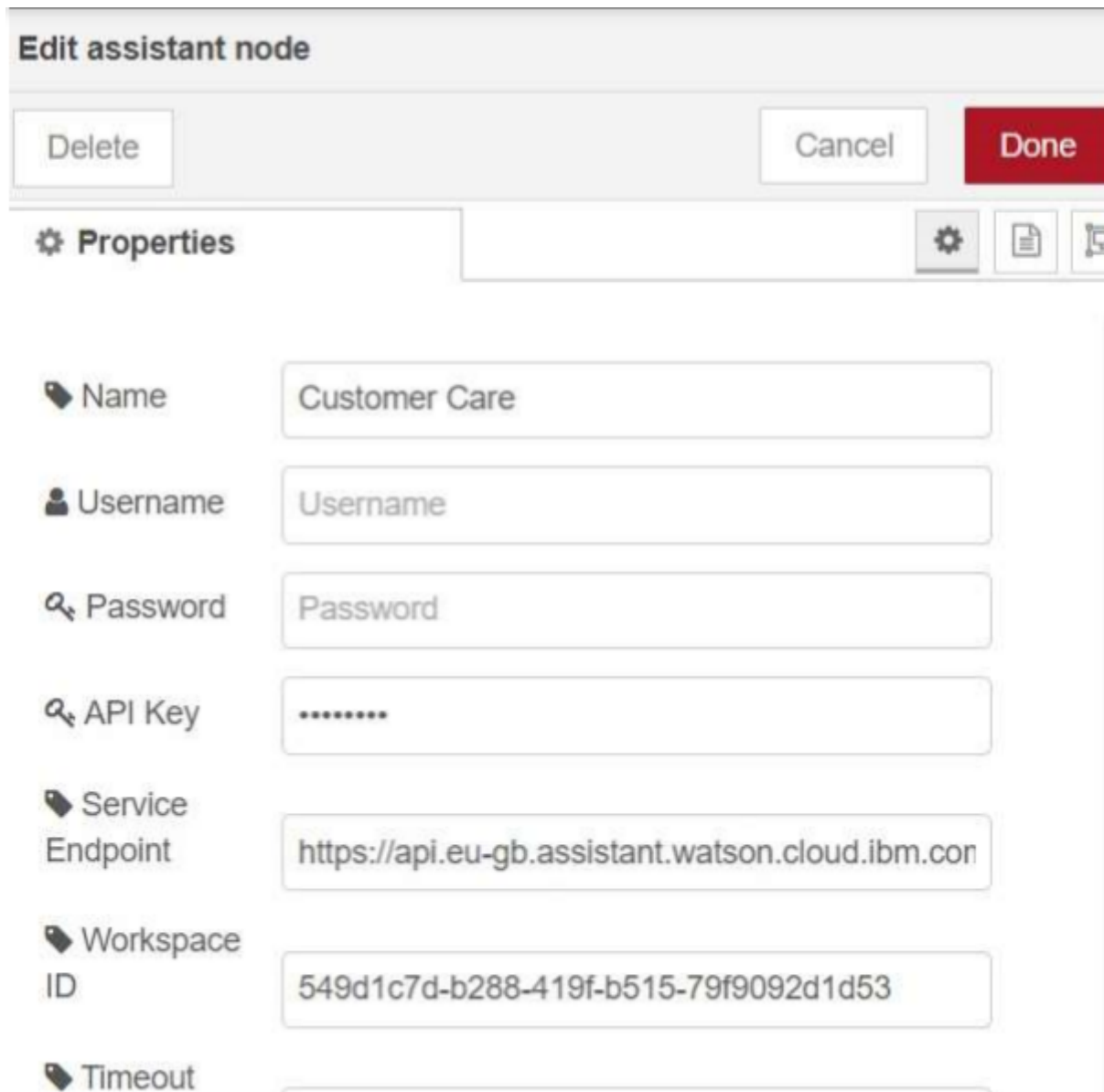
### Test in Assistant Tooling

- From the Dialog panel, click the Try It button, located at the top right side of the panel.
- Enter some user input:
- 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.
  - So 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.

## 5. Create flow and configure node

Integration of Watson assistant in Node-RED:





**Edit assistant node**

Delete Cancel Done

**Properties**

Name Customer Care

Username Username

Password Password

API Key .....

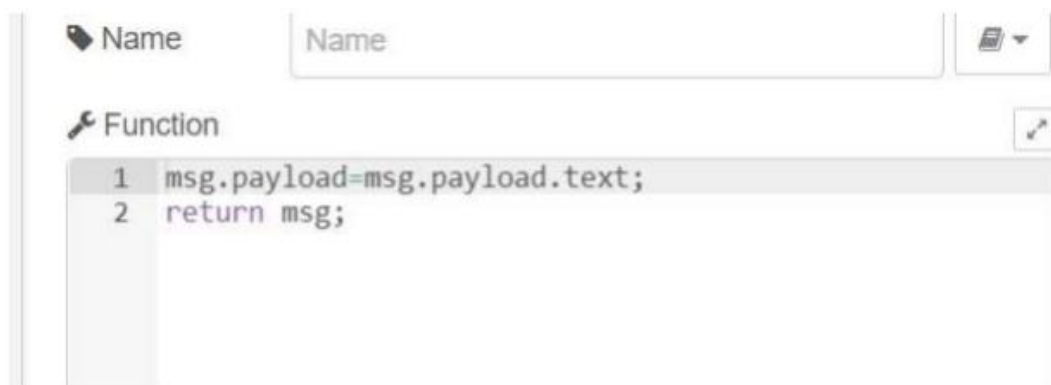
Service Endpoint https://api.eu-gb.assistant.watson.cloud.ibm.com

Workspace ID 549d1c7d-b288-419f-b515-79f9092d1d53

Timeout

- 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 template node on to the flow from the Input section.
- Drag Debug on to the flow from the output section.
- 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.
- Search for “Form” node and drag on to the flow
- Double click on the “form” node to configure
- Click on the edit button to add the “Group” name and “Tab” name
- Click on the edit button to add tab name to web application
- Give sample tab name and click on add do the same thing for group
- Give the label as “Enter your query”, Name as “text” and click on Done
- Drag a function node, double-click on it and enter the input parsing code as shown below

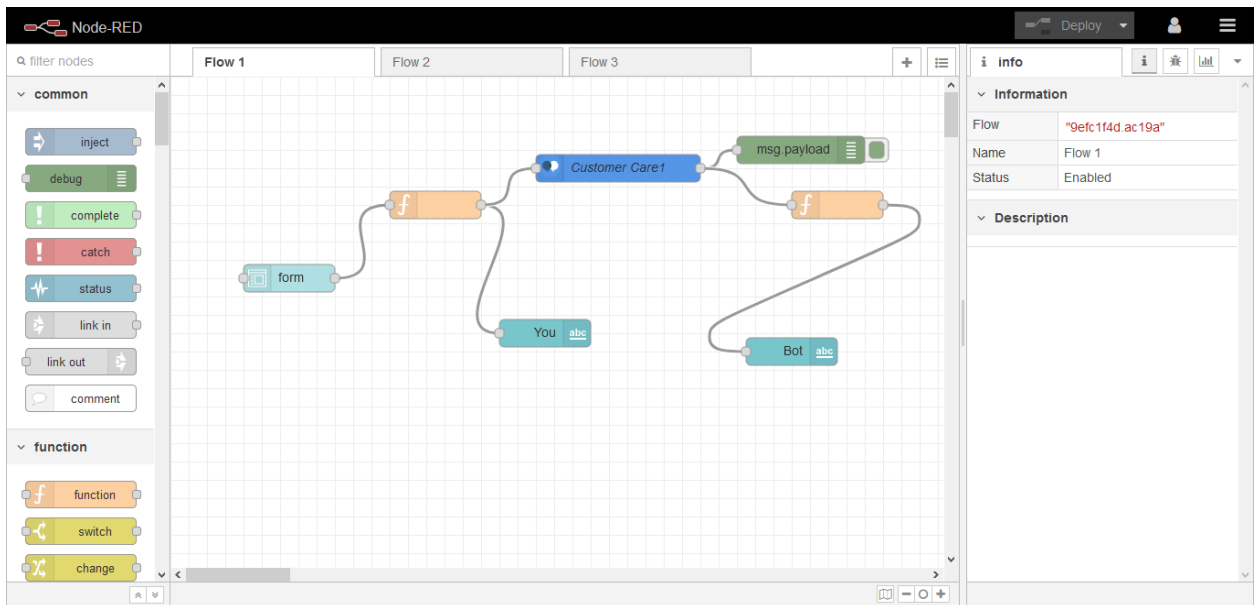


# 5.FLOWCHART

At first, go to manage palette and install dashboard.

Now, Create the flow with the help of following node:

- Template
- Assistant
- Debug
- Function
- Ui\_Form

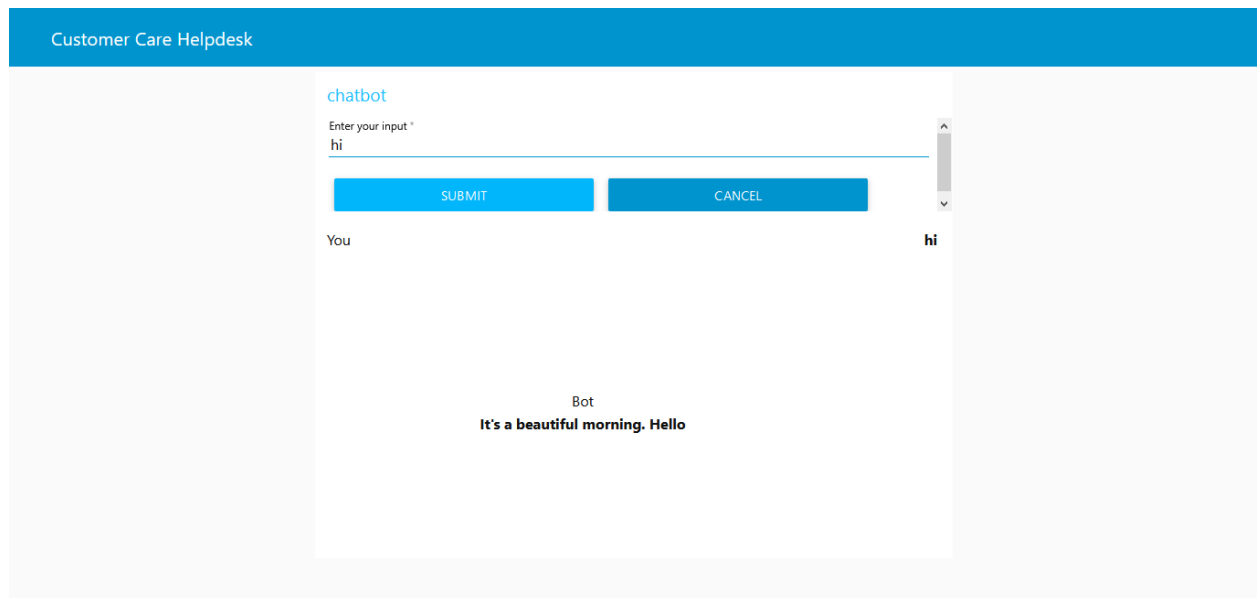


## 6. RESULTS

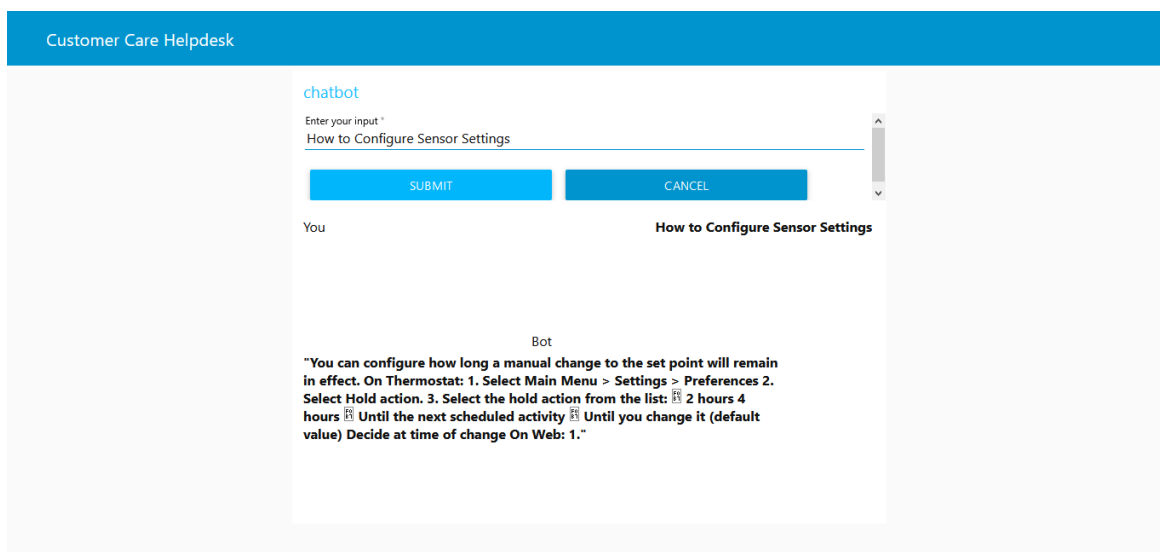
Finally, our Node-RED dash board integrates all the components and displayed in the Dashboard UI by typing URL.

<https://node-red-pdjhr.eu-gb.mybluemix.net/ui/#!/0?socketid=CEW6cQvMUzVWgmT3AAAH>

Here are some pictures showing the responses of our queries by the chatbot:



Query 1:



## Query 2:

Customer Care Helpdesk

chatbot

Enter your input \*

How to customize screen brightness

SUBMITCANCEL

You

How to customize screen brightness

Bot

"You can customize the brightness of your ecobee3's screen. The brightness for both the active and standby screens can be configured independently. You can also configure the screen to automatically sleep (i.e. turn off) whenever your ecobee3 enters the Sleep activity period."

## Query 3:

Customer Care Helpdesk

chatbot

Enter your input \*

How to set date and time

SUBMITCANCEL

You

How to set date and time

Bot

"The Date & Time screen lets you configure your time zone settings. If you didn't configure Wi-Fi in the previous step, you may need to reconfigure the current time and date. These settings are required in order for the scheduling features of your ecobee3 to work properly."

**Video presentation:-** <https://www.youtube.com/watch?v=Ey15Wizh0IE>

**Testimonial Video:-** <https://www.youtube.com/watch?v=UdvINSnTNGw>

# 7. 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 agent and customer agent can look on other works.

## **Disadvantages:**

- Sometimes chat bot can mislead customers.
- Giving same answer for different sentiments.
- Sometimes cannot connect to customer sentiments and intentions.

## **8. APPLICATIONS**

- It can be deployed on many popular social media applications like Facebook, Slack, Telegram.
- The Chatbot can deploy any website to clarify basic doubts of viewers

## 9. CONCLUSION

Thus, we have successfully created the Intelligent Customer Help Desk Smart Chatbot using Watson Assistant, Watson Discovery Services, Node-RED and Cloud-functions.

**Video presentation:-** <https://www.youtube.com/watch?v=Ey15Wizh0IE>

**Testimonial Video:-** <https://www.youtube.com/watch?v=UdvINSnTNGw>



## **10. FUTURE SCOPE**

We can include Watson Studio Text to Speech and Speech to Text services to access the chatbot hands free. This is one of the future scope of this project.

# 11.BIBLIOGRAPHY

/\*\*

\*

\* @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);  
  
});  
  
});  
  
}
```

## 12. Reference:

- [https://www.ibm.com/cloud/architecture/tutorials/cognitive\\_discovery](https://www.ibm.com/cloud/architecture/tutorials/cognitive_discovery)
- <https://cloud.ibm.com/docs/assistant?topic=assistant-getting-started>
- <https://developer.ibm.com/recipes/tutorials/how-to-create-a-watson-chatbot-on-nodered/>
- <http://www.iotgyan.com/learning-resource/integration-of-watsonassistant-to-node-red>
- <https://github.com/IBM/watson-discovery-sdu-with-assistant>
- IISPS\_INT\_520\_Intelligent Customer Help Desk with Smart Document Understanding

**Video presentation:-** <https://www.youtube.com/watch?v=Ey15Wizh0IE>

**Testimonial Video:-** <https://www.youtube.com/watch?v=UdvINSnTNGw>