Project Report

Topic: Intelligent Customer Help Desk with Smart Document Understanding

Name: Manish Kaswan

E-mail Id: kaswan410@gmail.com

Category: Machine Learning

GitHub id: Manish-Kaswan

Internship at smartinternz.com@2020

Contents

1. INTRODUCTION

1.1 Overview:

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

- Project Requirements: Python, IBM Cloud, IBM Watson
- Functional Requirements: IBM Cloud
- Technical Requirements: Python, Watson AI, ML
- Software Requirements: Watson Assistant, Watson Discovery
- Project Deliverables: Smartinternz Internship
- Project Duration: 19 days

1.2 Purpose:

The typical customer care chatbot can answer simple questions, such as store locations and hours, directions, and maybe even making appointments. When a question falls outside of the scope of the pre-determined question set, the option is typically to tell the customer the question isn't valid or offer to speak to a real person.
In this project, there will be another option. If the customer question is about the operation of a device, the application shall pass the question onto Watson Discovery Service, which has been pre-loaded with the device's 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**.**

Scope of Work:

- Create a customer care dialog skill in Watson Assistant
- Use Smart Document Understanding to build an enhanced Watson Discovery collection
- Create an IBM Cloud Functions web action that allows Watson Assistant to post queries to

Watson Discovery

- Build a web application with integration to all these services & deploy the same on IBM Cloud Platform.

2. LITERATURE SURVEY

2.1 Existing Problem:

Basically, Chatbot are loaded with a certain set of questions that is more like if and else flow, the question or the user input which lies out of the scope of the chatbot is not answered and rather a message like “Try Again “is displayed so that it directs the customer to the customer agent. Generally, the chat bot needs to provide an efficient result with least traffic reaching and problems. So, to achieve certain problems we need SMART chat bot so that it can answer the queries of the customer.

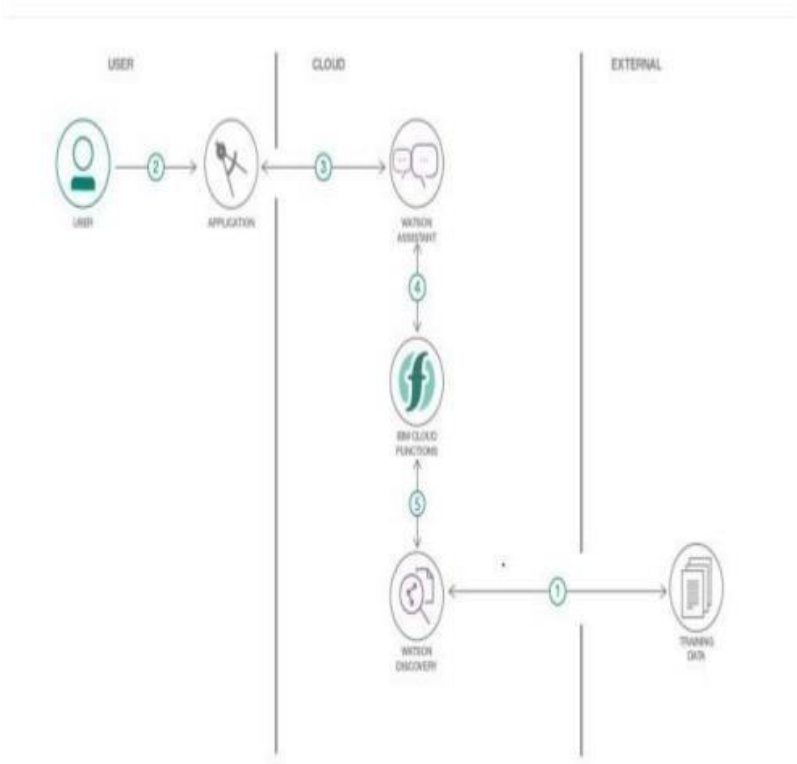
2.2 Proposed Solution:

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

In this project I used Watson Discovery to achieve the above solution.

3. THEORITICAL ANALYSIS

3.1 Block/Flow Diagram



1. The document is annotated using Watson Discovery SDU.

2. The user interacts with the backend server via the app UI. The frontend app UI is a chatbot that engages the user in a conversation.

3. Dialog between the user and backend server is coordinated using a Watson Assistant dialog skill.

4. If the user asks the product operation question, a search query is passed to a predefined IBM Cloud Function action.

5. Cloud function action will query the Watson Discovery service and return the results.

3.2 Hardware/Software Designing

- Create IBM Cloud services.
- Configure Watson Discovery
- Create IBM Cloud Function action
- Configure Watson Assistant
- Create flow and configure node
- Deploy and run node red app

4. EXPERIMENTAL INVESTIGATION

1.Create IBM services

a. Watson Discovery

b. Watson Assistant

c. Node-Red

2.Configuration of Watson Discovery –

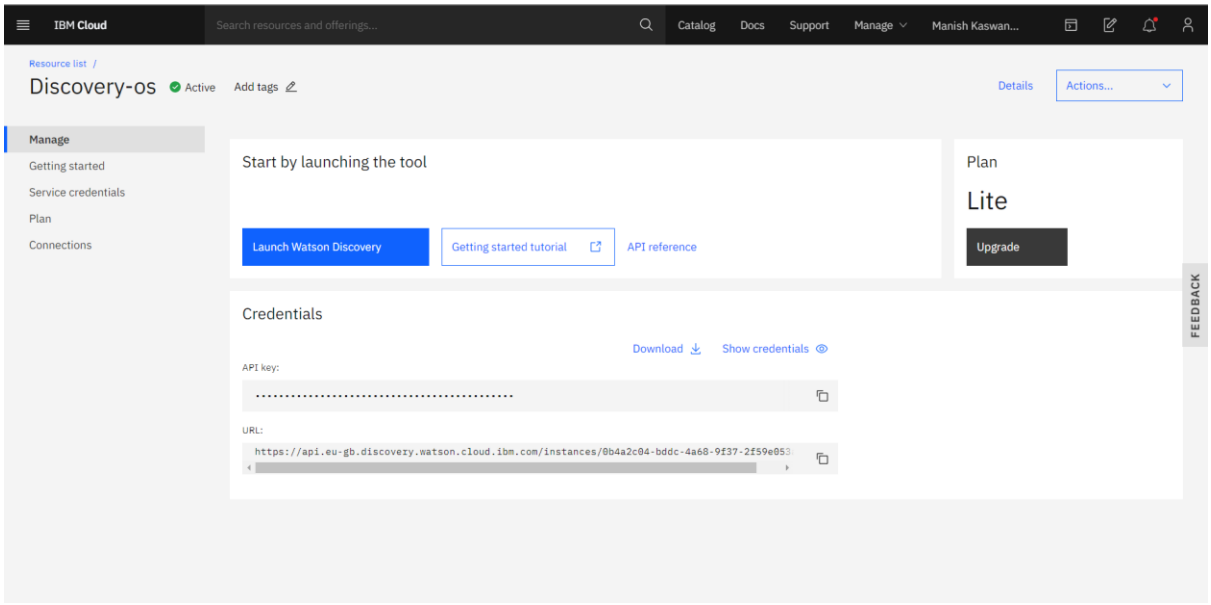
After creating and launching the discovery from the resources we have to attach the document i.e., ecobee3_UserGuide document from the local documents.

It is generally the basic user manual in which it describes the features and uses of the different components and the keys. So, without the smart document the result won't be that accurate enough. So, with this SMART DOCUMENT the user can get the accurate results. This can be done easily by clicking on the configure setting and then labelling each word or element present in the document as their respective label such as title, subtitle, text, image and Footer. Some of the labels are not present in the lite plan. In the lite plan we are provided with limited content of IBM Watson, the labels help us in segmentation of the document which helps the discovery to understand the document better and provide better results. The results provided by the discovery can be improved, all the results are shown in assistant in which the discovery finds the sentiment to be positive i.e. matching between the question or query entered by the user and the data of the document. Better the sentiment analysis accurate the results are.

Follow the below mentioned steps:

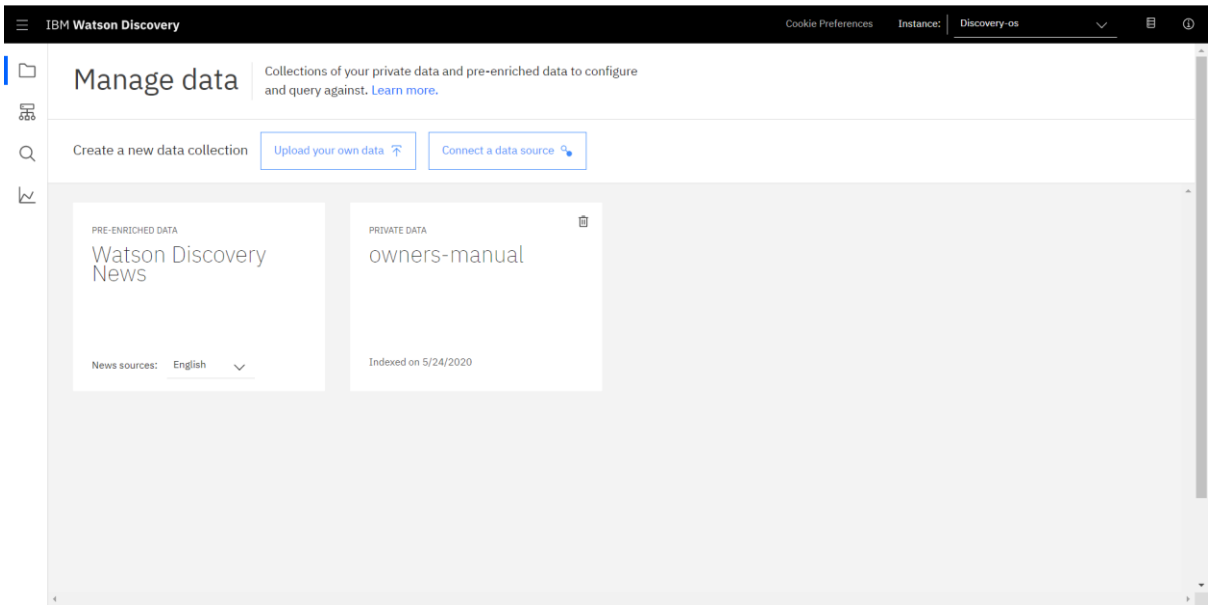
- After creating the discovery from the catalogue, we will be redirected to this base page of discovery where the name of the discovery along with its API Key and URL are mentioned.

These credentials will be used in further steps.

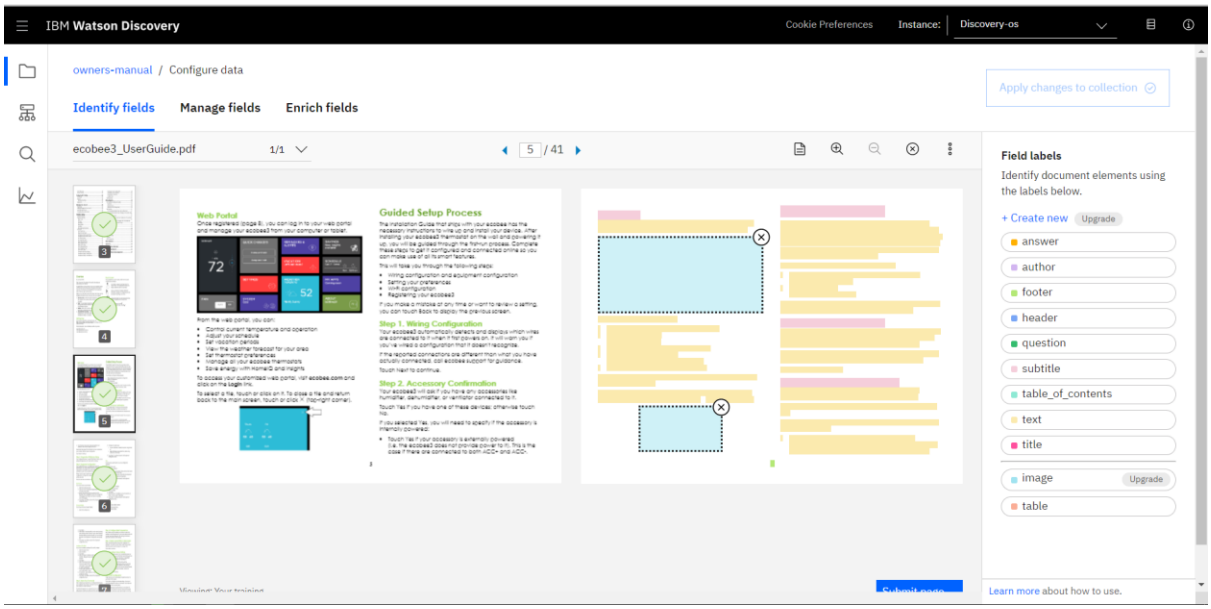


Click on the Launch Watson Discovery [1] to launch the discovery.

- Now in the next step we have to upload the data by clicking, upload your data. Here we have already uploaded the data as manual.

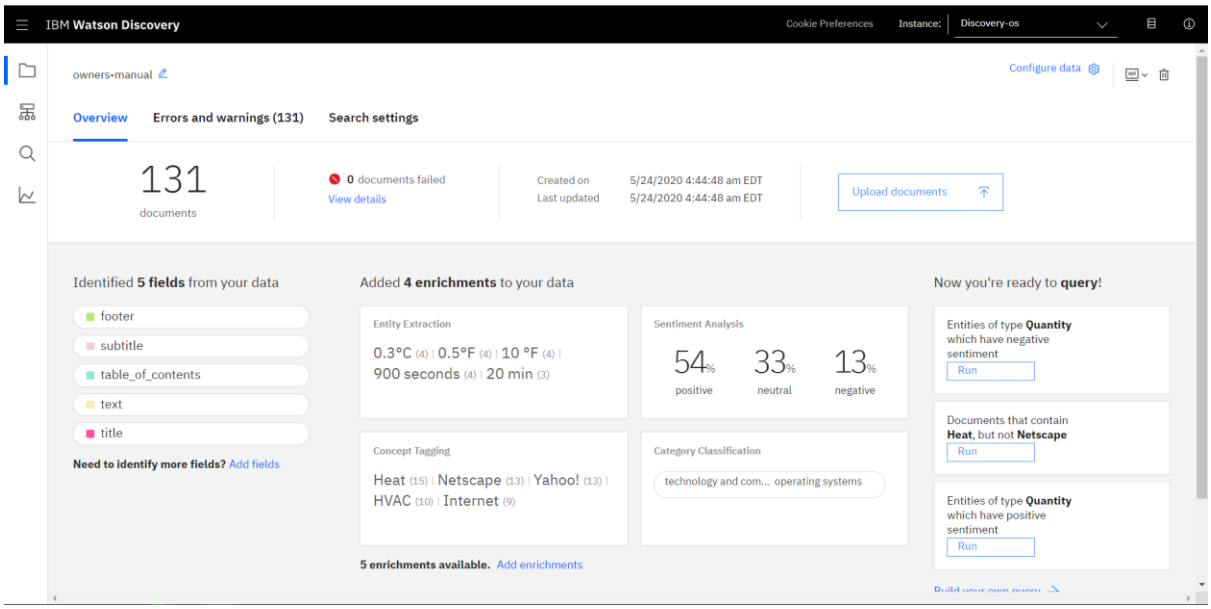


- After uploading the document, we will get a page where we can neglect the warnings and then we have to train the data by customizing it.

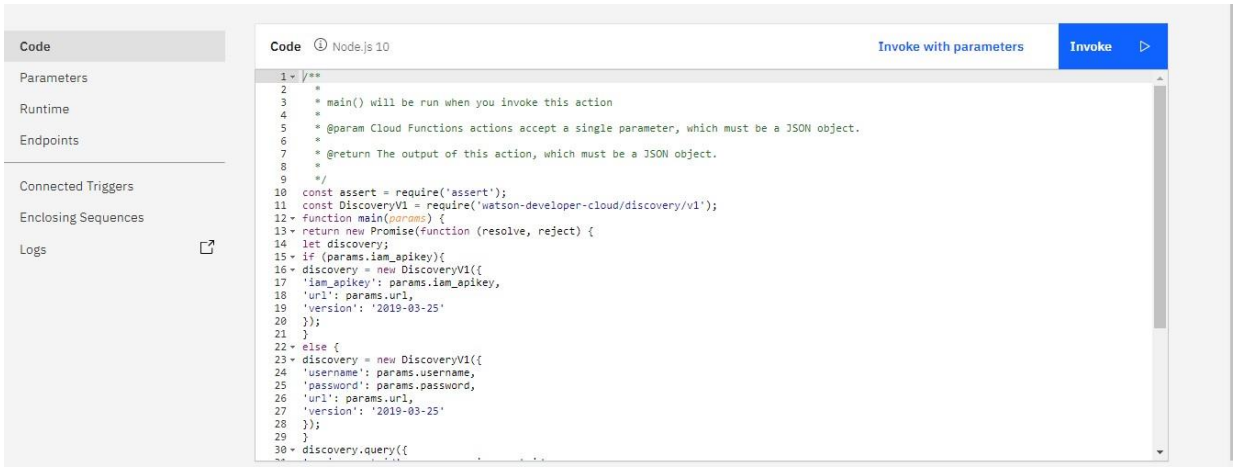


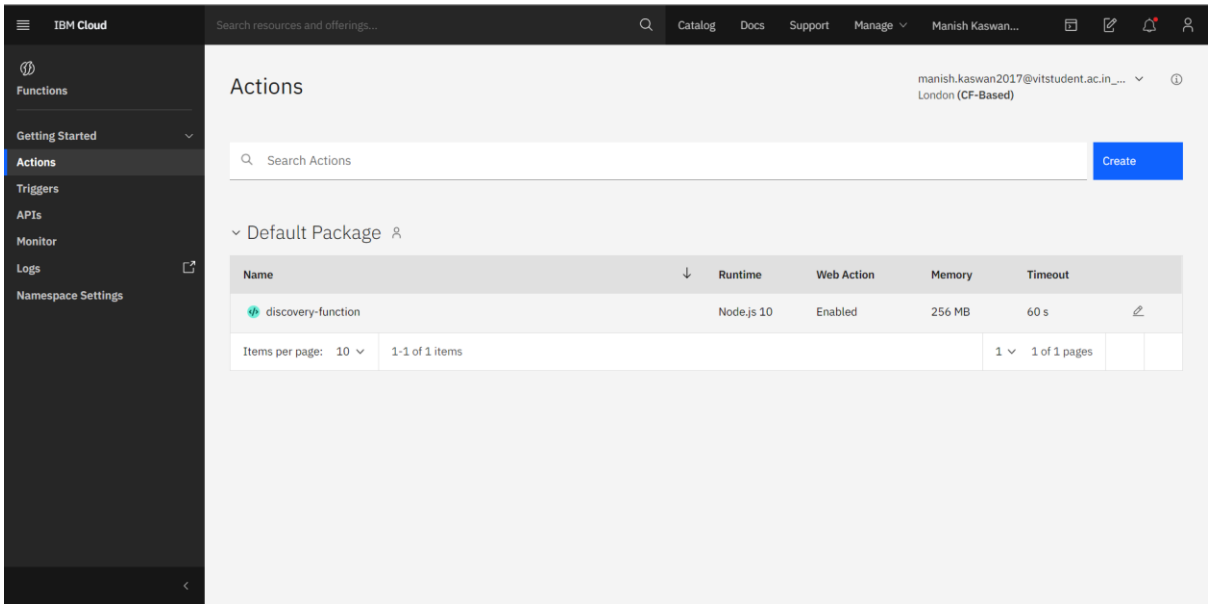
Then after customizing the data, the data need to be split according to different types here we consider it as subtitles format.

- After customizing the data, it will take some time to process and after that we will have multiple documents as shown below, the document we uploaded earlier is segmented in 131 documents as shown below.
-

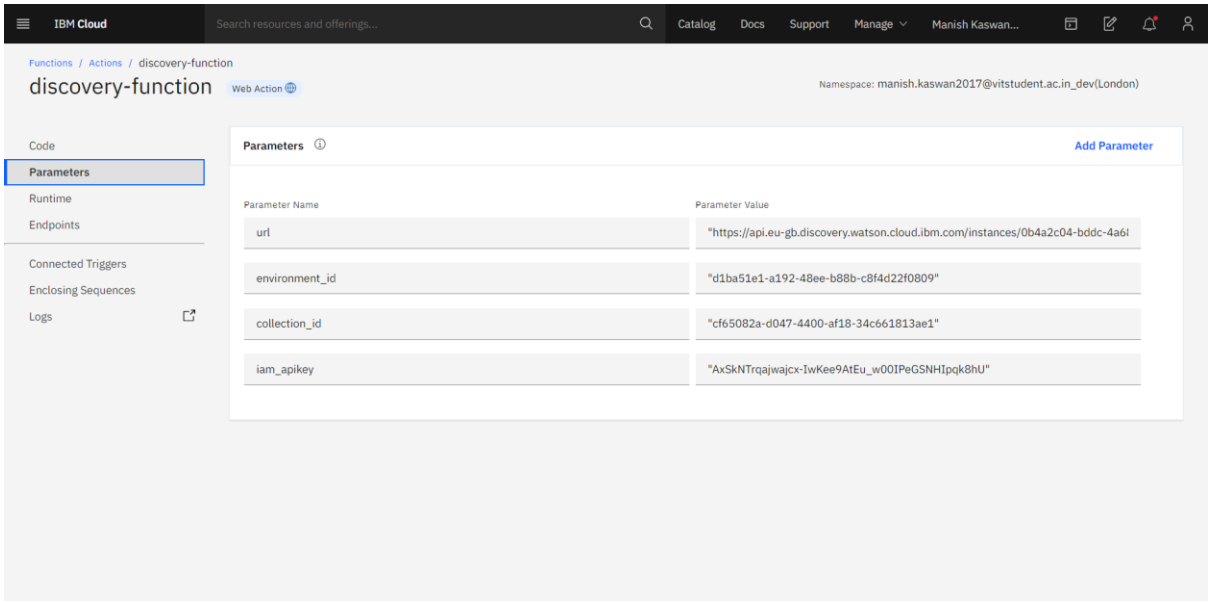


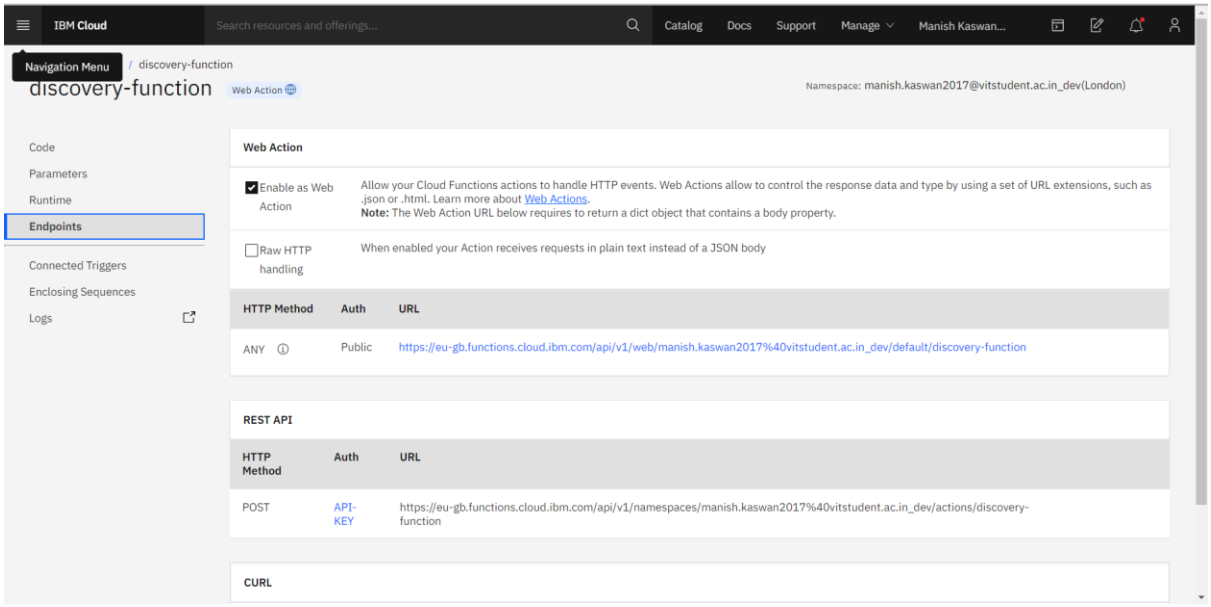
- We already have the API key and the URL.
- Next, we have to create the IBM Cloud Function Action: It is used to link the discovery with assistant, so that our queries can be answered by the discovery. After selecting the action from the IBM catalogue, we have to click on the action tab as shown on the left menu. Here we made the Information function. Then we can post the code which will help us to link the discovery.





We can make the parameters as per the code and paste the parameter value from the discovery credentials. After that we have to click on the endpoint and enable the web action which will generate a public URL and it will be further used.





•Next, we have to make the Watson assistant and use the sample customer care skill for convenience. We can add intent related to product information and the related entities and dialog flow.

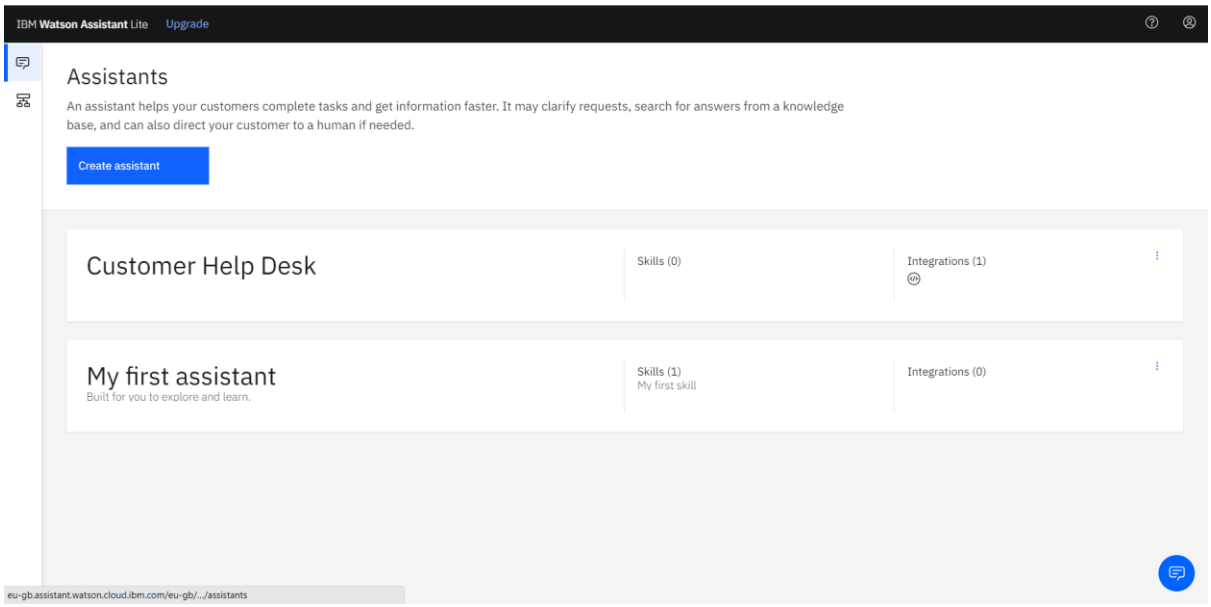
We have to create the intents and the name we used here is Product Information.

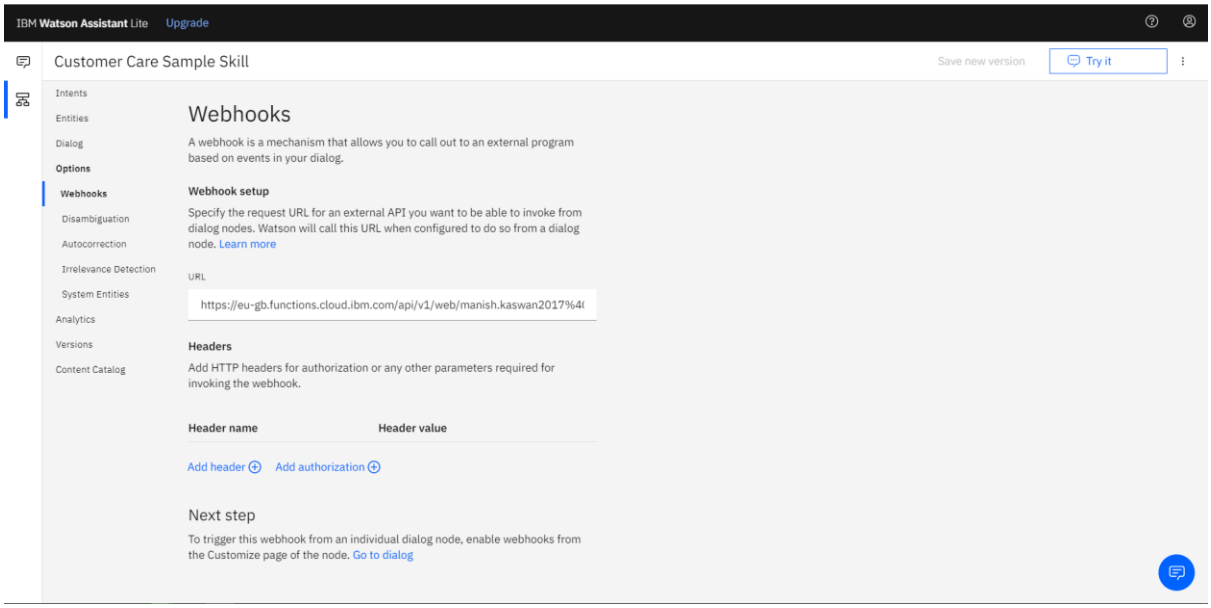
The contents in the document need to be given as example in the intents so that when the customer needs any help the answer can be resulted using the keywords i.e., the examples.

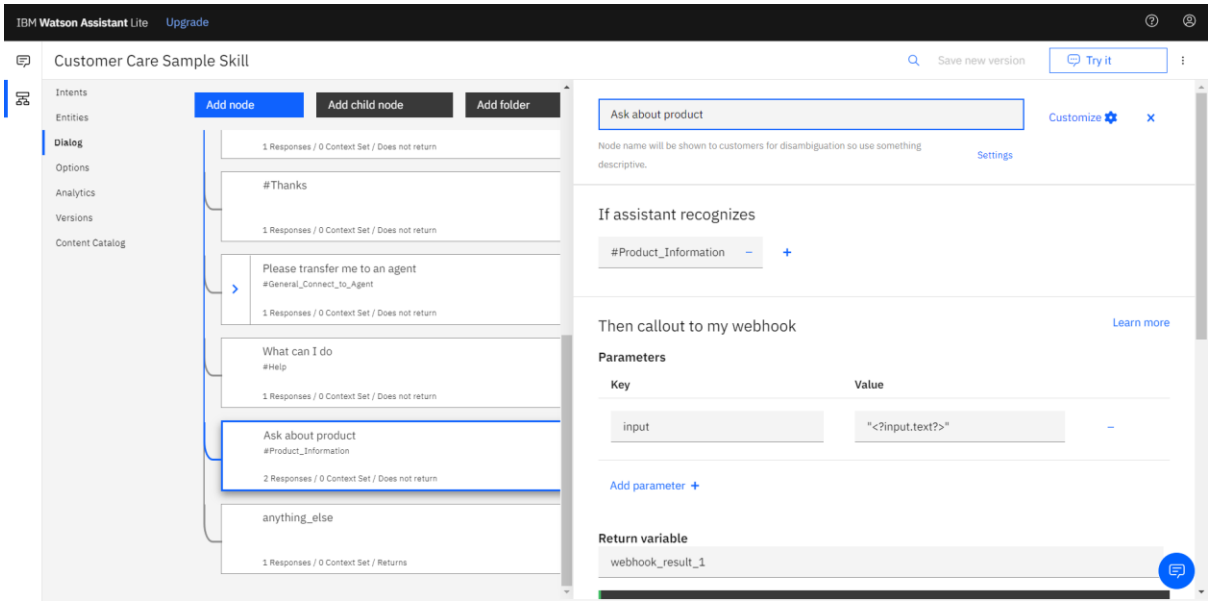
•After adding all the examples, we have to add a node in the dialog i.e., Ask about product where the webhook URL is given so that it takes the document from the intent and searches the result.

We have to enable the webhooks which enables our dialog to send a POST request to the webhook URL.

If the assistant recognizes the intent then the webhook is to be called and returns the value as webhook_result_1 by taking the input parameters.



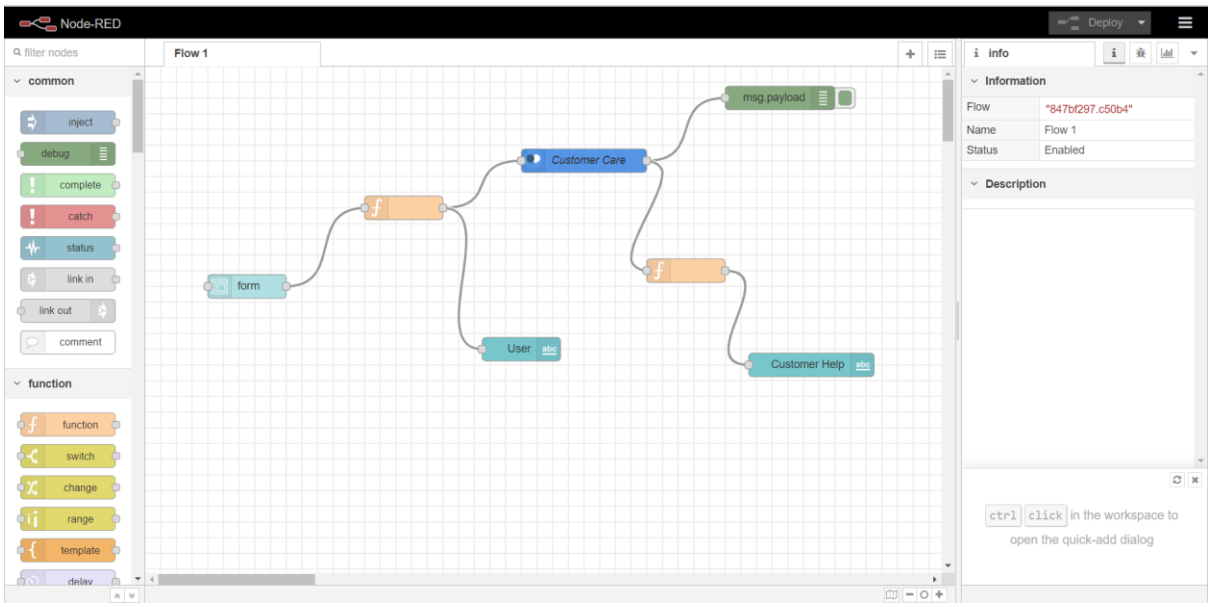



IBM Watson Assistant Lite
Upgrade
Customer Care Sample Skill
Save new version
Try it
Intents
Entities
Dialog
Options
Analytics
Versions
Content Catalog
Add node
Add child node
Add folder
1 Responses / 0 Context Set / Does not return
#Thanks
1 Responses / 0 Context Set / Does not return
Please transfer me to an agent
#General_Connect_to_Agent
1 Responses / 0 Context Set / Does not return
What can I do
#Help
1 Responses / 0 Context Set / Does not return
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
Node name will be shown to customers for disambiguation so use something descriptive.
Settings
If assistant recognizes
#Product_Information
Then callout to my webhook
Learn more
Parameters
Key
Value
input
"<?input.text?>"
Add parameter
Return variable
webhook_result_1

Creation of Node-RED in IBM cloud:

- Step-1: Login to IBM and go to the catalogue
- 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
- Step – 6: Secure your node red editor by giving a username and password and click on Next
- Step – 7: Click Next to continue
- Step – 8: Click Finish
- Step – 9: Click on Go to Node-Red flow editor to launch the flow editor
- Node red editor has various nodes with the respective functionality

• After this we have to make the node-red flow, and link everything. We will get a UI from the node. The roles of different nodes can be understood by the references mentioned in in the end. The final flow will look like as shown below.



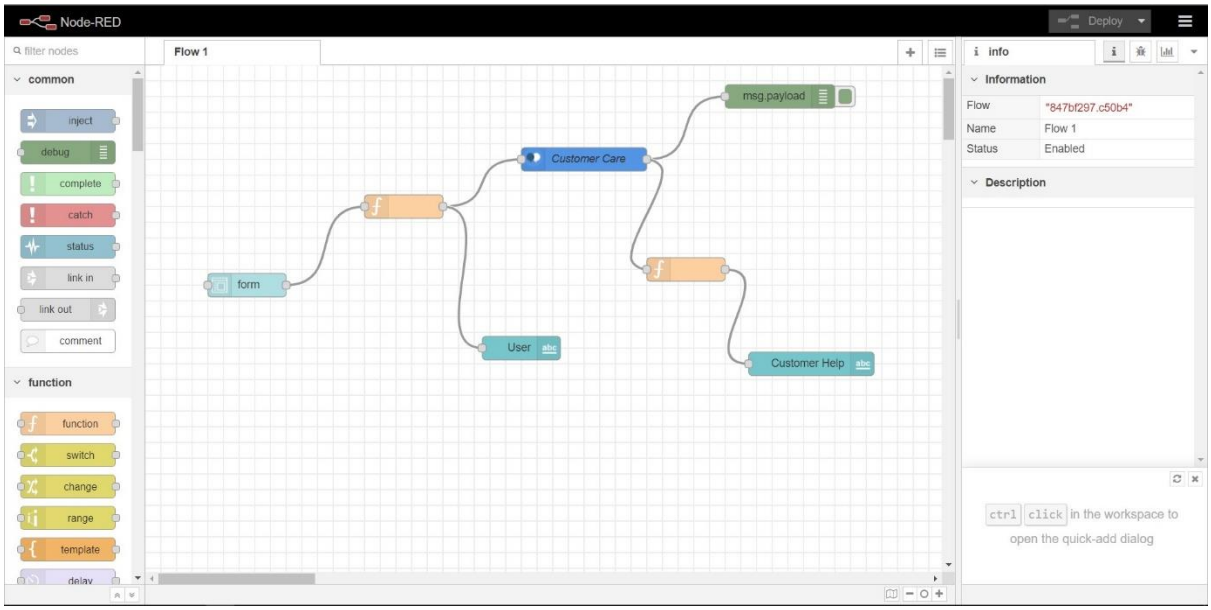
The UI we have is the basic one but can be improved by writing the HTML code in the template node. We can vary the background colour also from the node-red. This is how the initial flow looked like but we imported another flow just to improve the UI This basic flow can be extended to what we have used next, we have imported the flow to make thigs easy and to have an interesting UI.

5. FLOW CHART

1.Create flow and configure node:

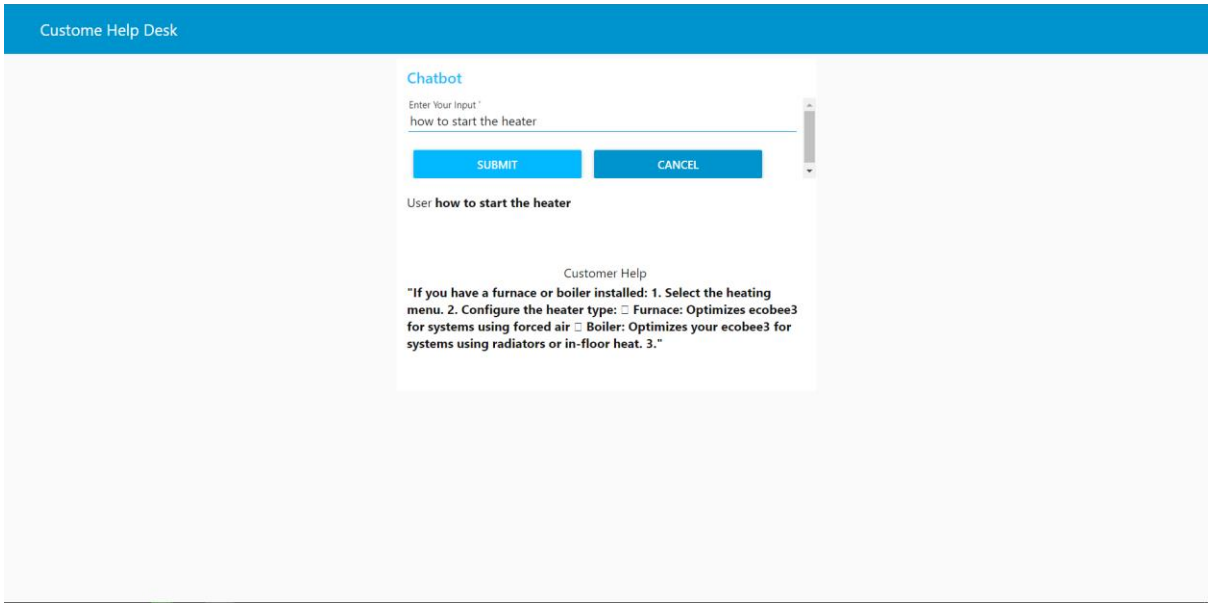
At first go to manage pallet and install dashboard. Now, Create the flow with the help of following node:

- Inject
- Assistant
- Debug
- Function ● Ui_Form
- Ui_Text



6.RESULT

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



7. ADVANTAGES AND DISADVANTAGES

ADVANTAGES:

- More efficient than the previous manuals.
- Reduces work load on the employees.
- Results in accurate and takes less time.
- Solves issues faster.
- Deliver faster, smarter, more personalized service.

DISADVANTAGES:

- Sometimes it may result in wrong answer where there may be issue in the sentimental analysis.
- It may take some time to display the result when there are multiple occurrences of the keyword in the document.
- Giving same answer for different sentiments.
- If the data is not trained properly then the result will not be accurate.

8. APPLICATION

- Chatbot can be applied in various fields to help the customer in finding the result in larger documents.
- It can be used to find the data in social medias and in any communication channel.

9.CONCLUSION

By using this process, we can create a basic chatbot that helps the customer to clear their basic needs and issues. By creating the IBM WATSON, WATSON ASSISTANT and WASTON DISCOVERY for the data to be imported from the local computers and NODERED application to show the flow of the data which results in the output with the usage of the webhooks and we have successfully crated the smart help desk using these applications.

10.FUTURE SCOPE

We can update the data by uploading the pre-built node red flow and then modifying it and then deploying it. The data in the functions can be modified once if we change the documents that need to be processed. We can also improve the results of discovery by enriching it with more fields. We can also include Watson text to audio and Speech to text services to access the chatbot handsfree. These are few of the future scopes which are possible.

11. APPENDIX

11.1 CODE:

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);
  });
 });
}
```

11.2 References:

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