Project Report

Title:-

Intelligent Customer Help Desk with Smart Document Understanding-SB3150

Under guidance of SmartInternz

Link for Chatbot :-

https://node-red-ayubn.eu-gb.mybluemix.net/ui/#!/0?socketid=UdJVRDxAdTiCEe2-AAAB

Made by:-

Name - Vivek Garg

SBID – SB20200003150

Project ID - SPS_PRO_99

Email ID – vivekgarg9999@gmail.com

TABLE OF CONTENTS

1.Introduction

1.1 Overview

In this we will be adding extra functionality i.e., Smart Document Understanding in Watson assistant by using Watson discovery and integrate all the services using node-red and creating web dashboard and deploy the same on IBM cloud.

1.2 Purpose

We are trying to make the bot more robust my making it to answer the query out of its scope.Asthese chatbots are limited to some set of questions.

So in order to avoid such cases we try to add feature like Smart Document Understanding (SDU), the customer won't have to undergo the tedious procedure of connecting with customer care representative. Instead the answer of his query will be displayed through the chatbot. The solution provided will be accurate and handy as it is straight from owner's manual which has been preloaded in Watson Discovery.

2.Literature Survey

2.1 Existing Problem

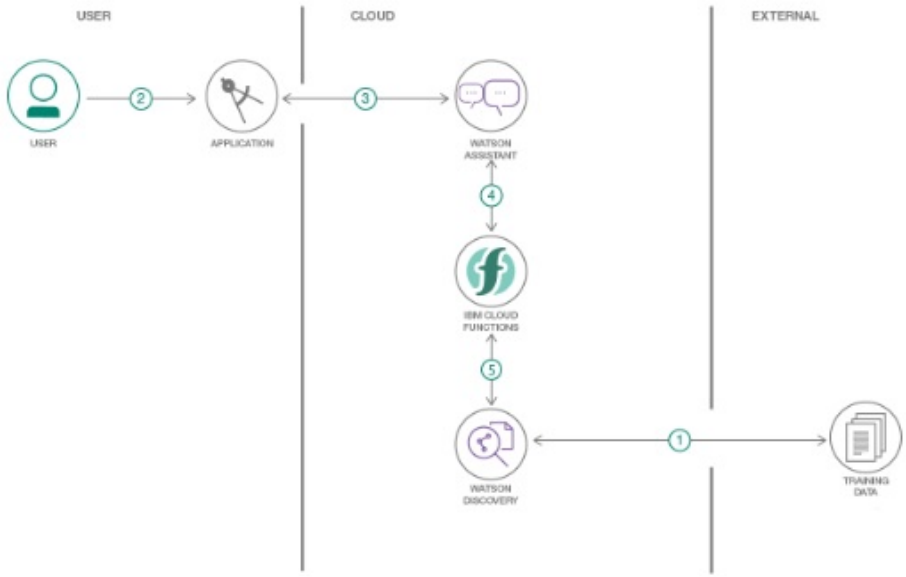
A normal/standard chatbot only reply to certain set of questions. Thus when a customer ask about technical things bot does not reply or it connect to the customer service representative. Due to which customer has to wait until the representative resolve their query. Thus this make the bot useless in customer eyes.

2.2 Proposed Solution

Solution here is Smart Document Understanding using Watson Discovery. Here whenever the question is out of scope assistant we send this query to Watson discovery which is already pre-loaded with technical manual and then discovery will search for that query in the manual and will return the answer. And also if customer is not happy with the response then he/she can talk to representative. In this way the customer won't have to undergo the tedious process of connecting with the representative and the customer will get lightning fast response. This will thereby enhance customer service experience.

3.Theoritical Analysis

3.1 Block Diagram



According to this diagram, the user interact with the Chatbot using node-red web dashboard UI where he enter his query. After this query Is pass on to Watson assistant to response. And if it is out of its scope then query is passed to cloud function action from where it is passed to Watson discovery, which is already annotated with manual to reply the technical query.

3.2 Hardware/Software Designing

- Create IBM Cloud services
- Configure Watson Discovery
- Create IBM Cloud Functions action
- Configure Watson Assistant
- Create flow and configure node
- Deploy and run Node Red app.

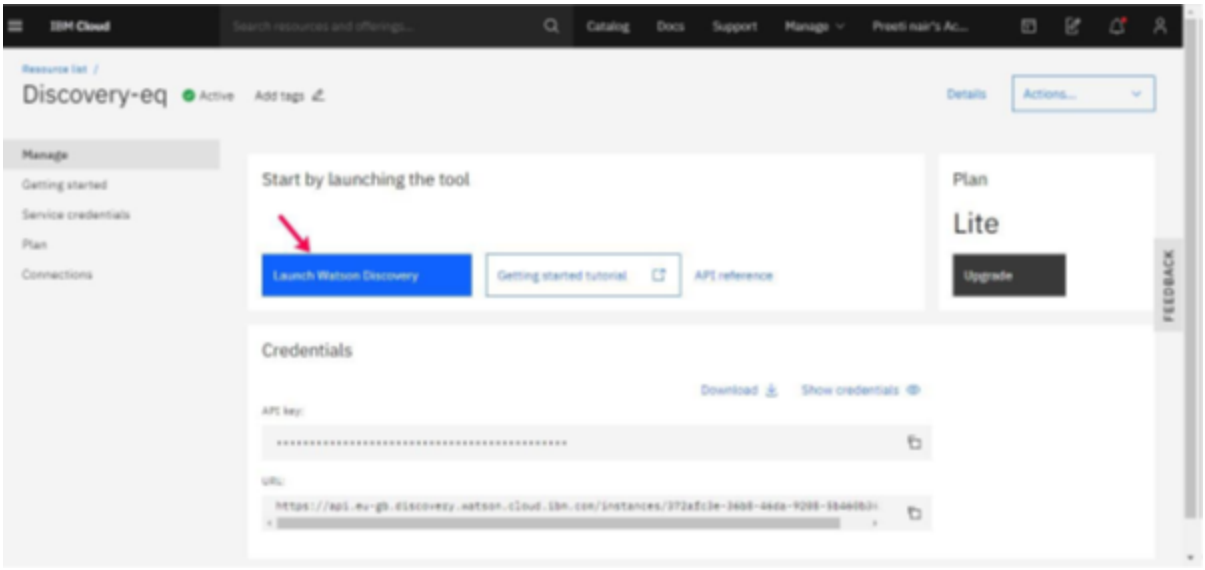
4. Experimental Investigation

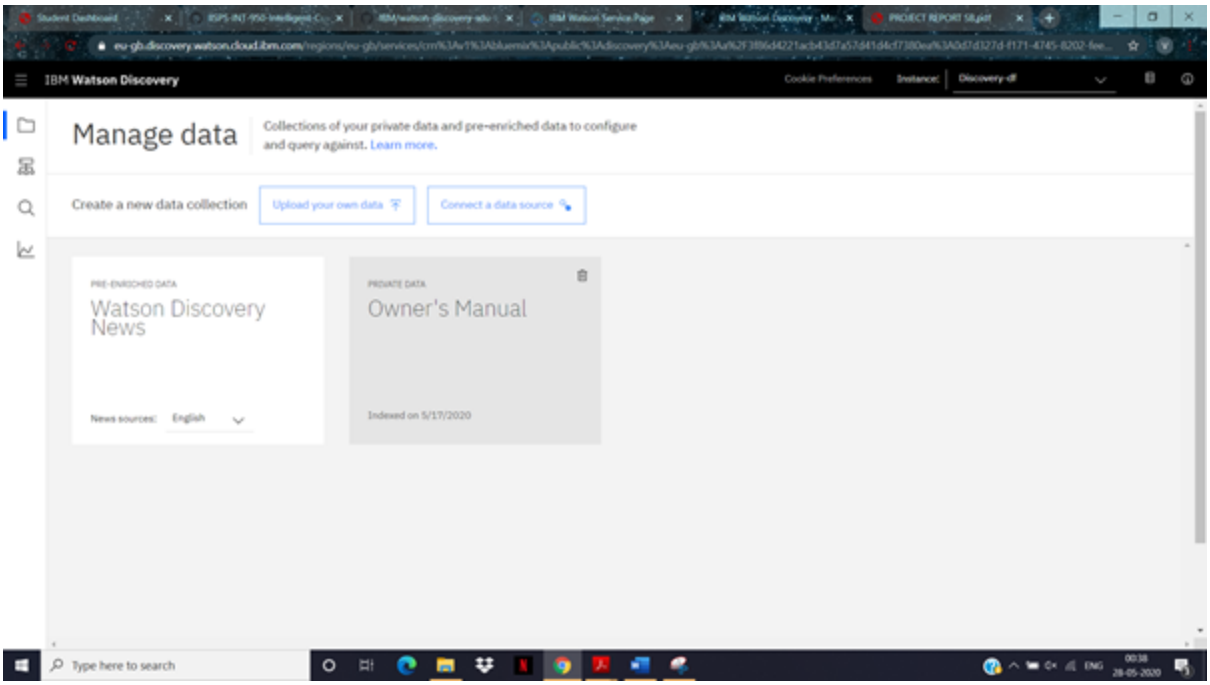
We create essential cloud services:-

- Watson Assistant
- Watson Discovery
- Cloud Function
- Node-Red

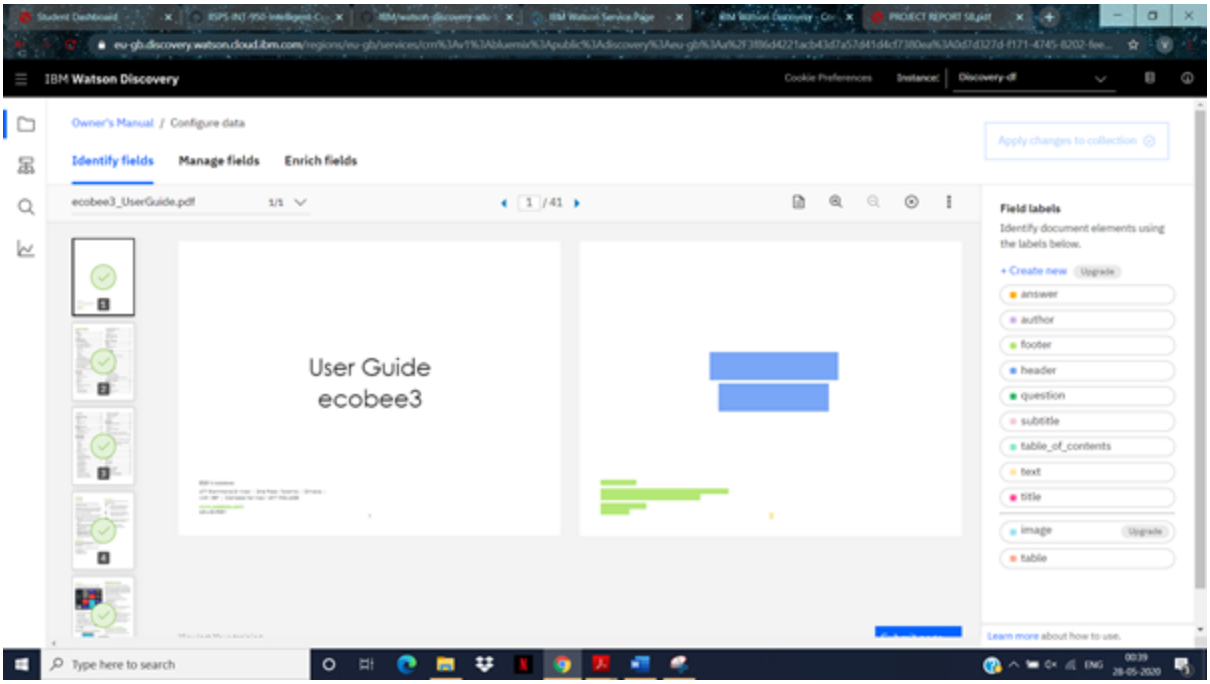
Step 1 :- Configure Watson Discovery

After creating Watson discovery instance on the cloud we open it. After this we create a new collection where we upload the manual and train the discovery by splitting the documents in heading, footer, Subtitles, text, etc. this splitting helps the discovery to understand the document better and this Feature of discovery is called Smart Document Understanding.



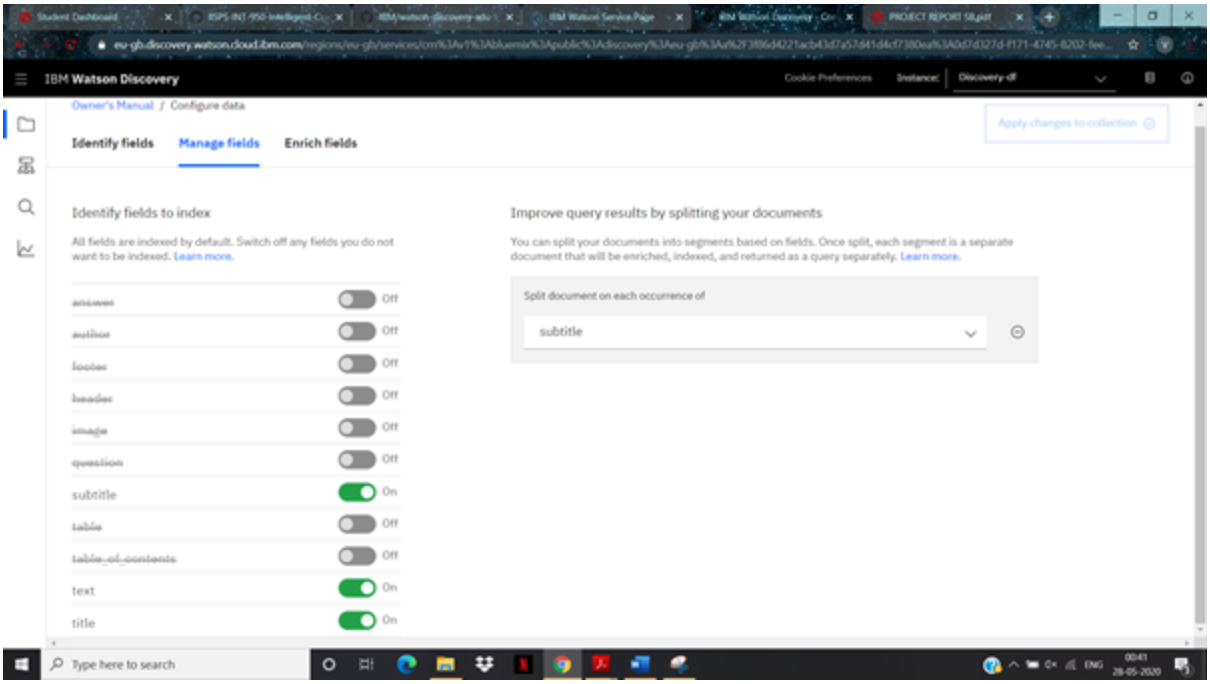


Splitting of document happens here like shown in figure**.**

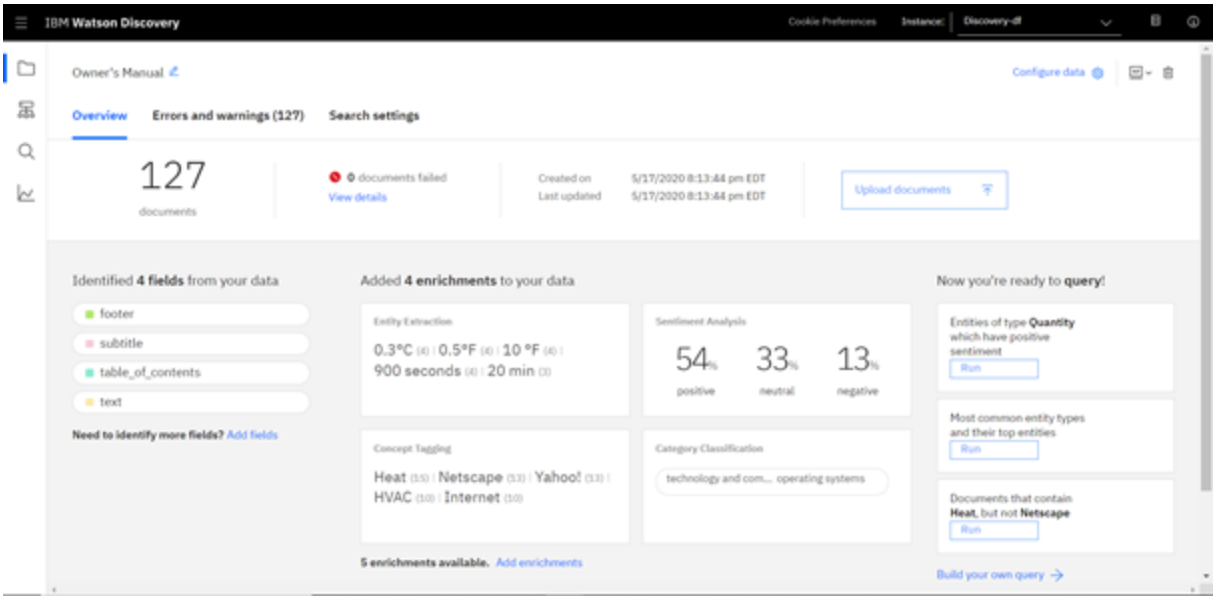


For further segmentation and making the sub documents, we have to

manage the fields. Here we are provided with the option of identifying field to index i.e. what all texts are important for us, as we can see in the below snip, we have turned on only subtitle and text because they are the only 2 labels in which we are interested.

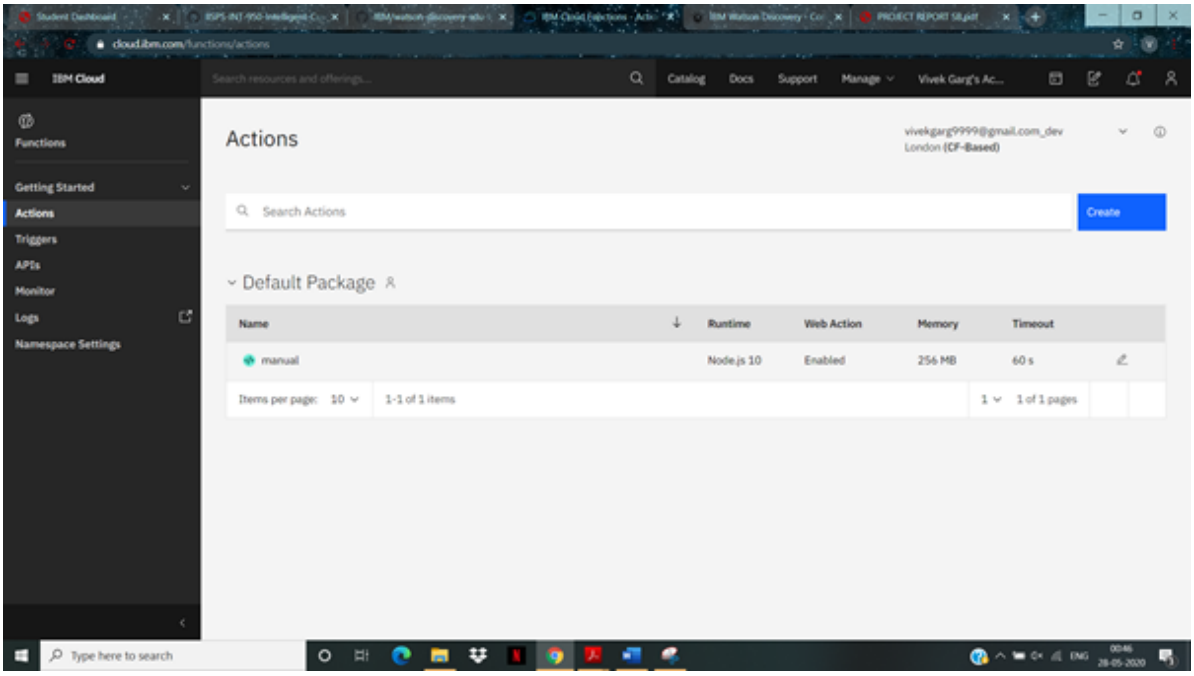


This is the final annotated manual after training discovery

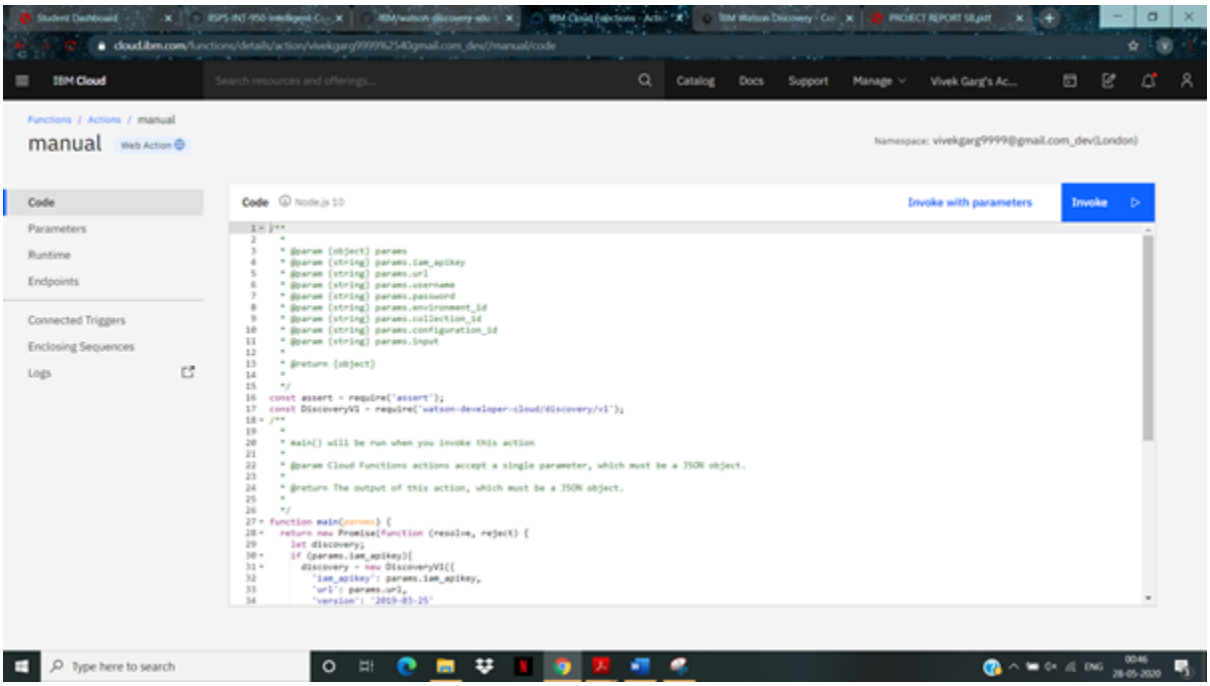


Step 2: Cloud Function

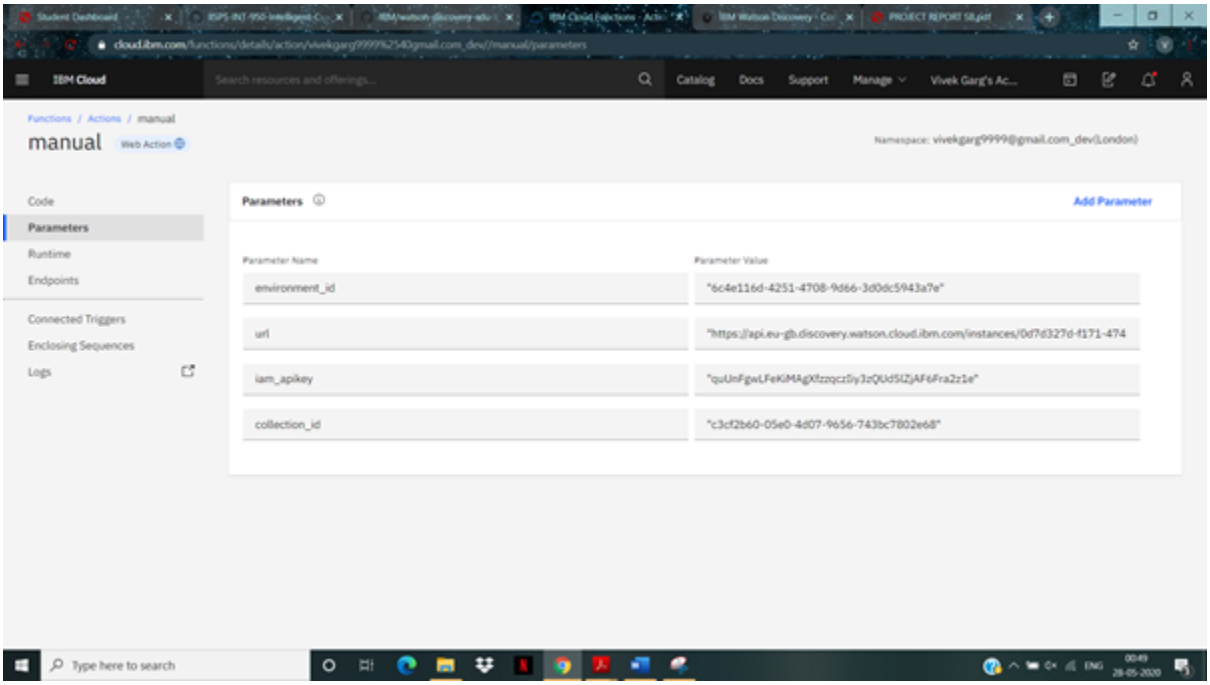
Now to connect the discovery with we need cloud function. Now we open cloud function from IBM catalog and click on action button and we create a function to link function with discovery.

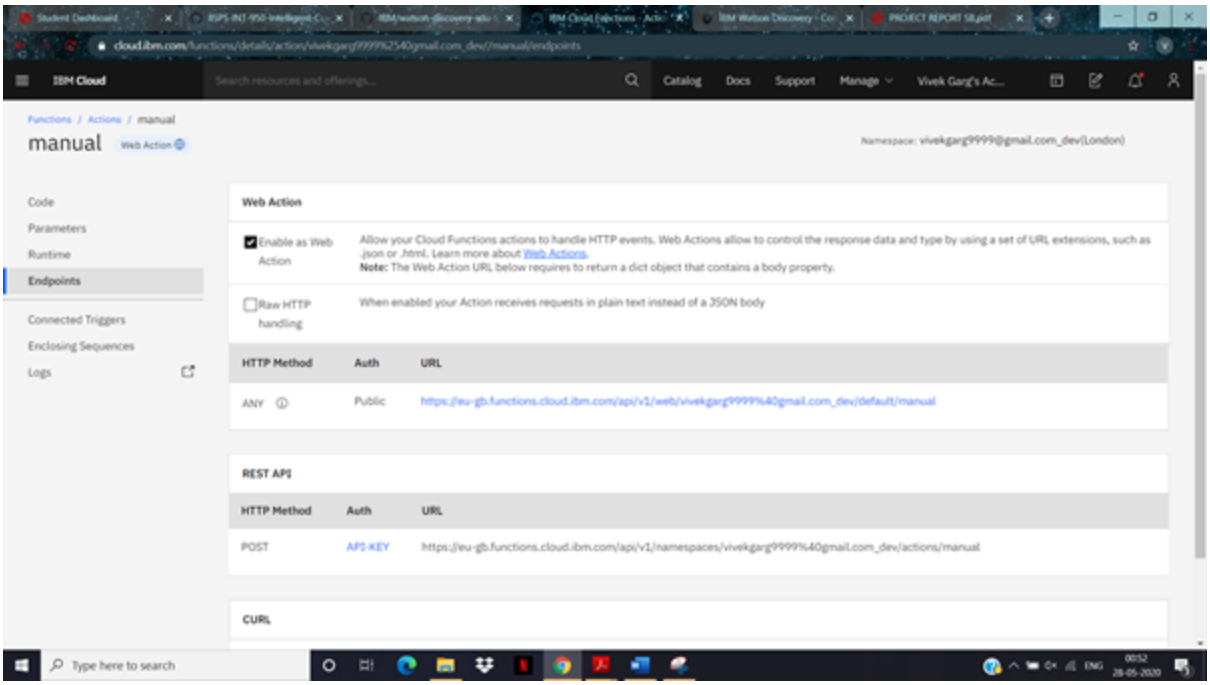


After this we write the code in the editor.



And for parameters we click on parameters tab and enter the parameters like api-key, collection id which has to be obtained from Watson discovery. After all this we need to enable web action to generate a link which will be passed to Watson assistant.



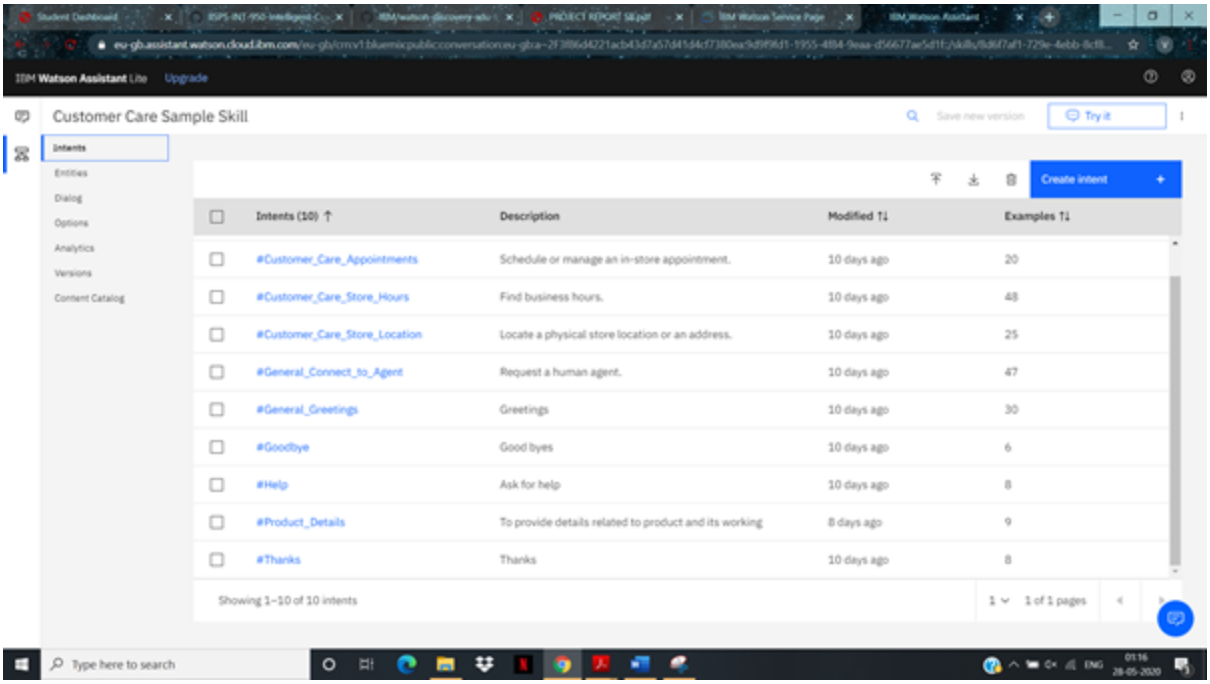


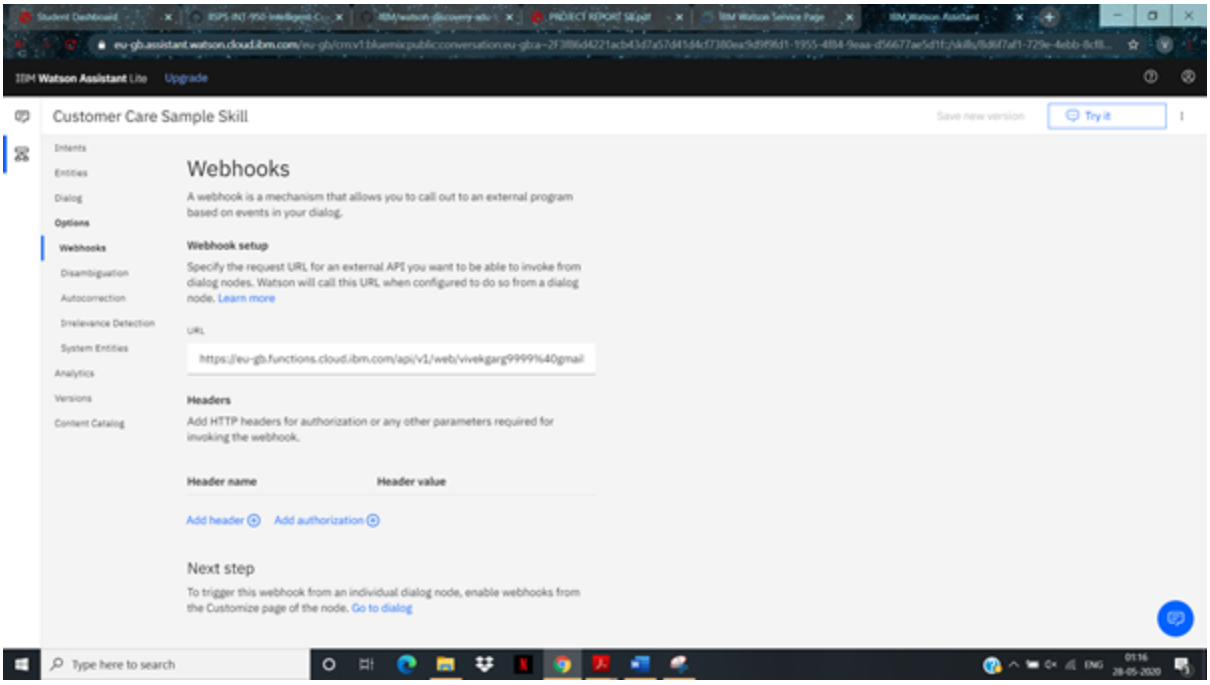
Step 3 :- Watson Assistant

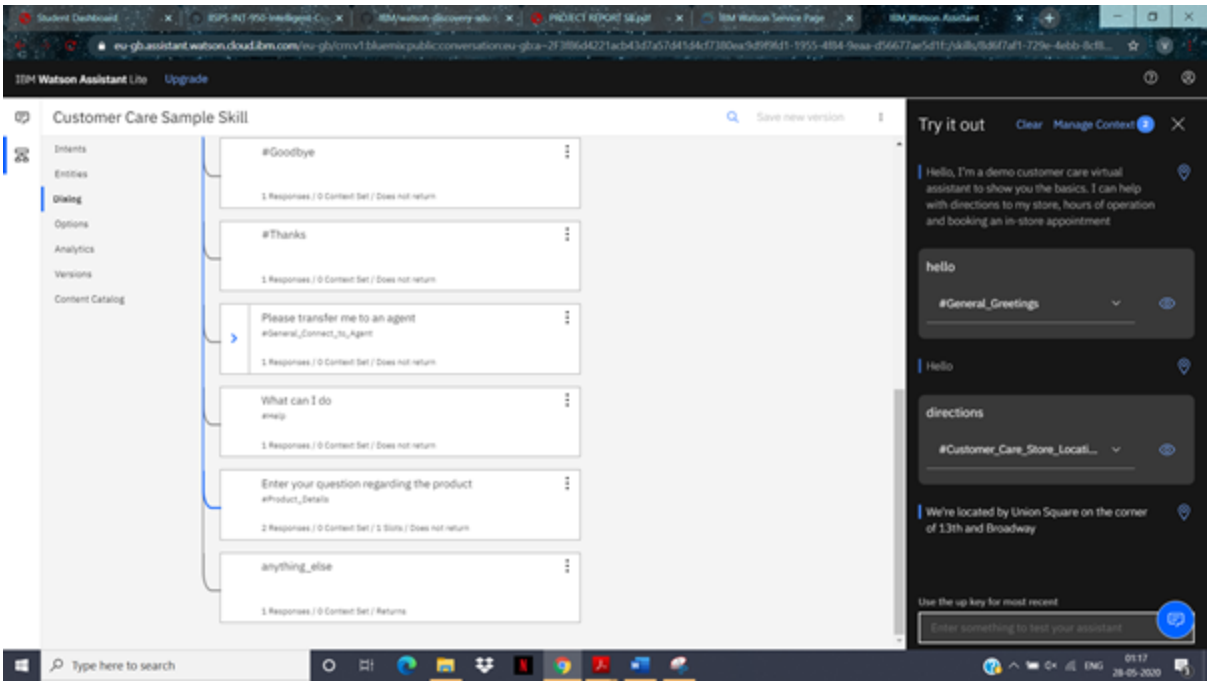
Now we launch Watson assistant and create a skill where we import Sample Customer Care Skill for our convenience. After this we need to add intent for PRODUCT INFORMATION and related entities and dialog flow**.**

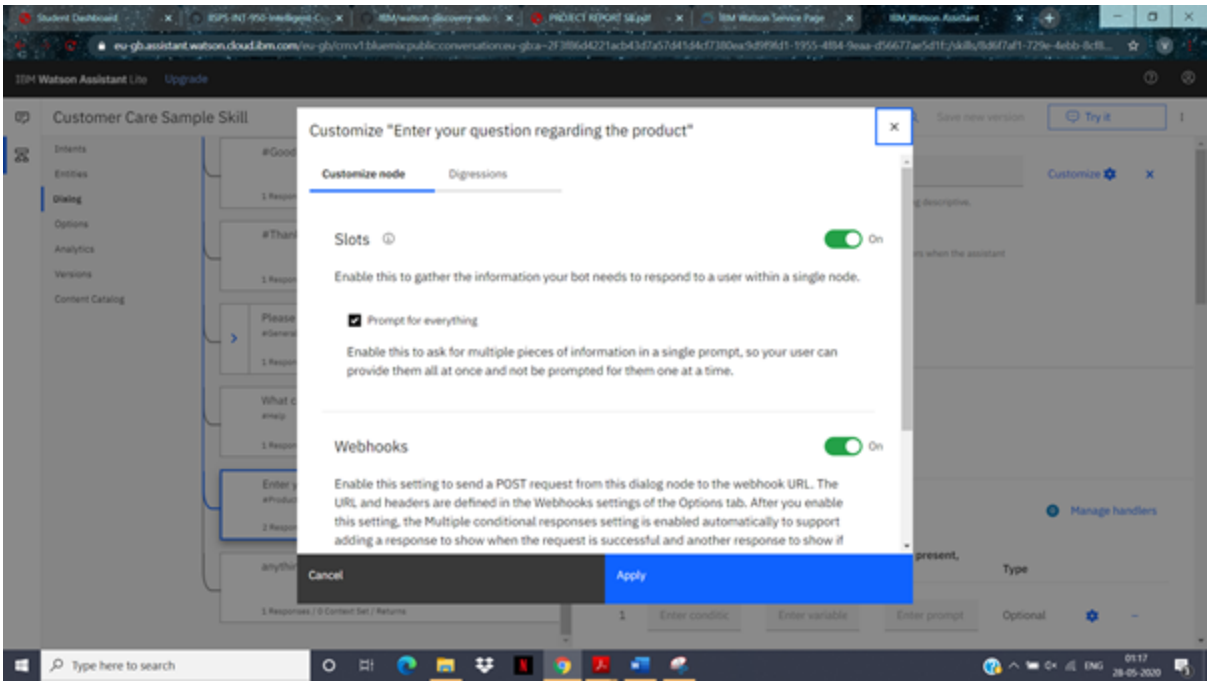
- Intents- These are the categories which we mention or we expect the user input to be, for example: Greetings can be an intent and in it we can have examples as Good Morning, Good Evening and all.
- Entities- These are used to mention the usual typos of the user and the synonyms like some people write the good morning as gm, good morning, good morning, so we can cover all these also instead of returning a message to rephrase.
- Dialog- Here we mention the outputs to be given, these can be static as well as dynamic.

Also after this we enable webhooks which enables our dialog to send a POST request to the webhook URL.





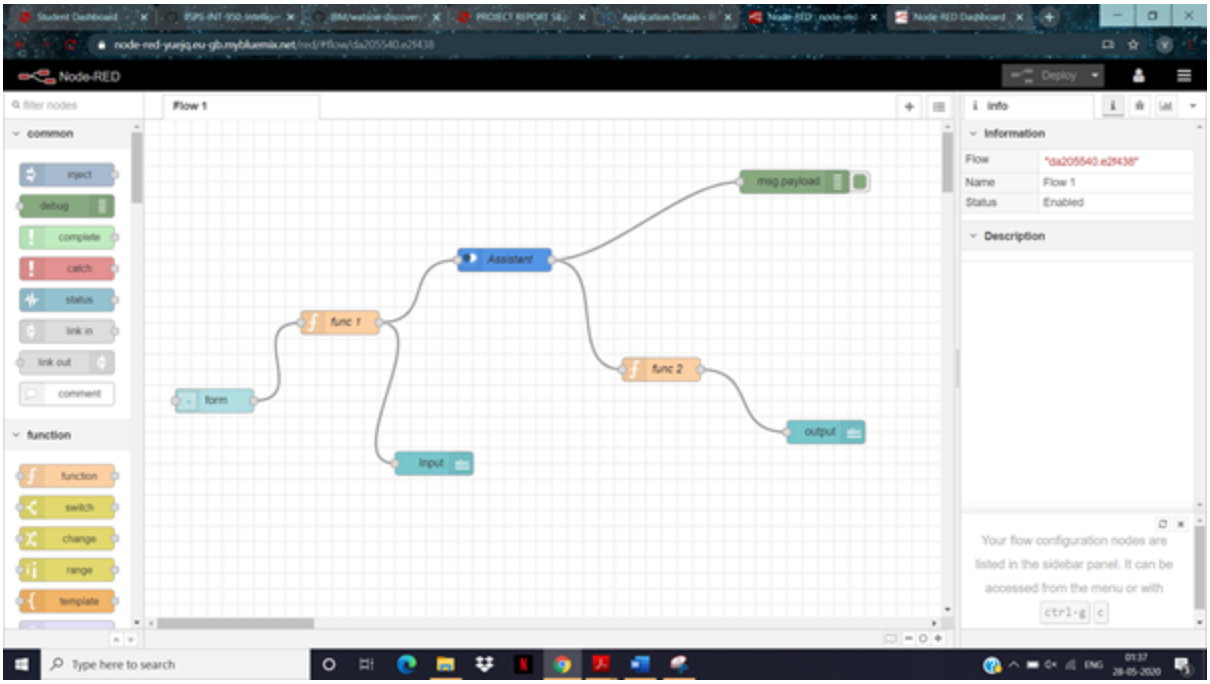




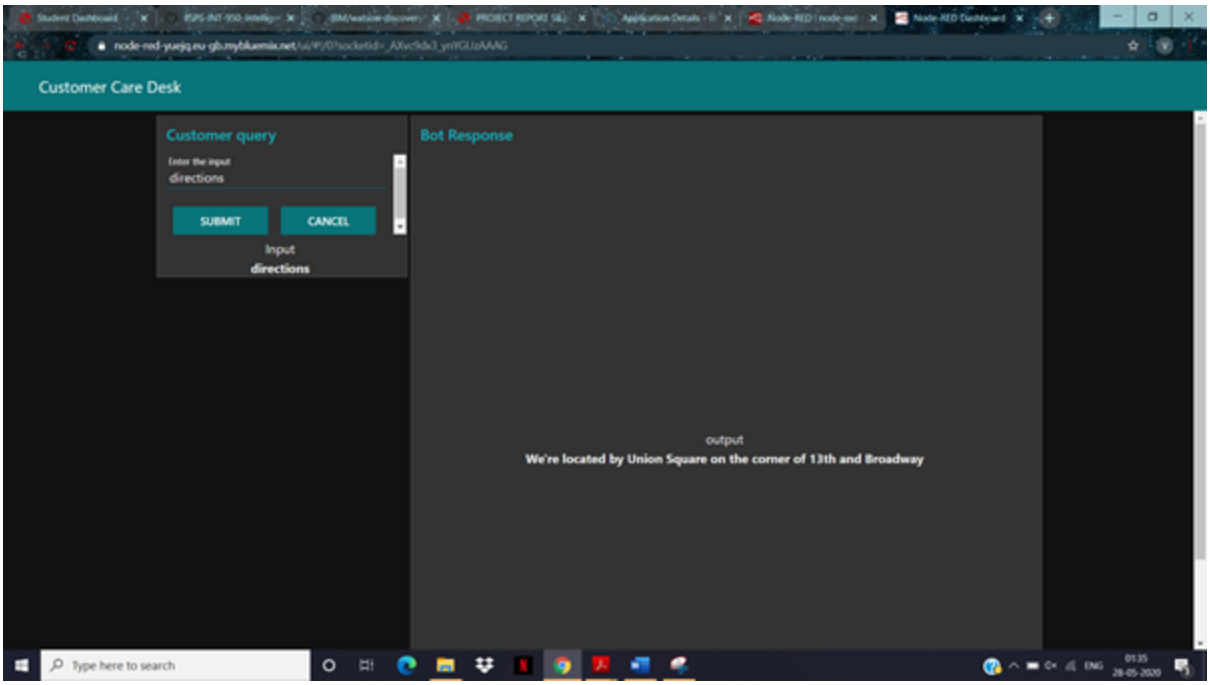
Step 4:- Node-Red

Now we create an instance of node-red, where we will integrate all the necessary services together and also we will create the UI for the chatbot using node-red web dashboard library

This is the flow chart of chatbot.

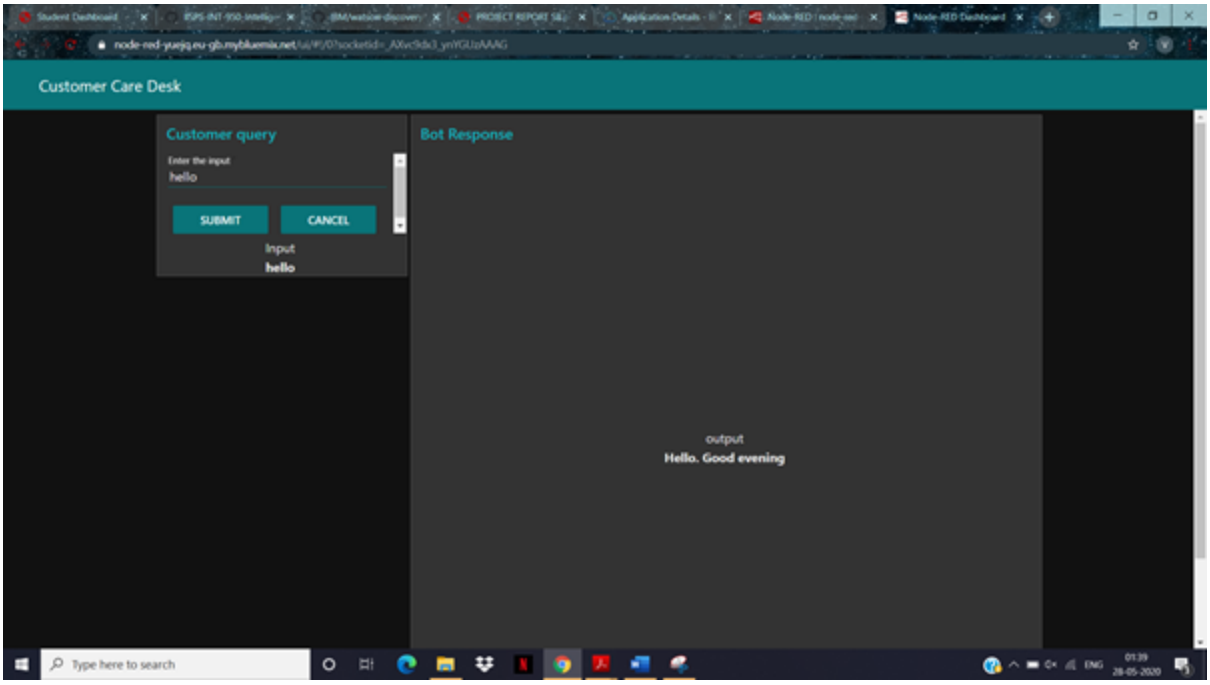


And this is interface of the Customer Chatbot.

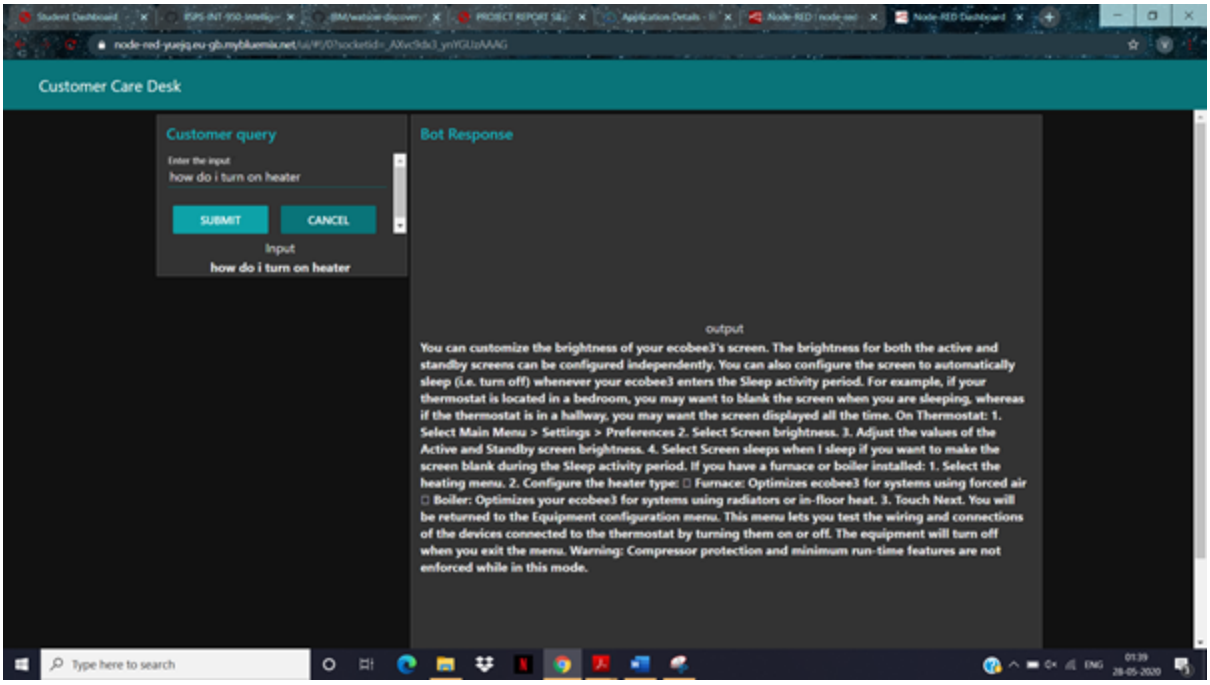


5. Result

This figure shows the pre-determined answer of Bot.



This figure below shows that bot is answering the technical query which are out of its scope.



6. 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**.**
- Faster Customer Service.
- Increased Customer Satisfaction & 24/7 availability.
- Multiple Customer Handling.

Disadvantages :-

- Sometimes chatbot can mislead customers.
- Giving same answer for different sentiments.
- Sometimes cannot connect to customer sentiments and intentions.
- Limited Responses for Customers.
- Maintenance.

7. Applications

These kind of chatbot can be integrated with social networking platforms like Telegram, WhatsApp, WeChat, etc. which can provide businesses to provide 24/7 customer service to the user.Chatbot can be deployed at any website to clear the basic doubts of the customer.

Also, One of the most useful application of these bots is in healthcare department. An average patient spends 30 minutes trying to get to the right service in a local hospital. But deploying conversational chatbots in the healthcare sector can significantly reduce long waits. From registration to coverage and claims.

8. Conclusion

We have created a chatbot which not only engage the customer with small talk but will also provide user with accurate and smart solution. The response will be fast and customer will be satisfied with the service. By integrating all the above mentioned services, a smarter and efficiently working Customer Help Desk has been developed.

9. Future Scope

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

10. Bibliography

- *https://developer.ibm.com/articles/introduction-watson-discovery/*
- *https://developer.ibm.com/tutorials/how-to-create-a-node-red-starter-application/*
- *https://github.com/watson-developer-cloud/node-red-labs*
- *https://developer.ibm.com/components/watson-assistant/series/learning-path-watson-assistant*
- *https://developer.ibm.com/articles/introduction-watson-discovery/*
- https://developer.ibm.com/patterns/enhance-customer-help-desk-with-smart-document-understanding/
- *https://cloud.ibm.com/docs/openwhisk?topic=cloud-functions-getting-started*
- *https://developer.ibm.com/technologies/web-development/articles/ws-restful*

12. Appendix

Code for Cloud Function

```
/**
 *
 * @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);
    });
  });
}
```

Code for Node-Red Flow

```
[
    {
        "id": "da205540.e2f438",
        "type": "tab",
        "label": "Flow 1",
        "disabled": false,
        "info": ""
    },
    {
        "id": "23556120.70ecde",
        "type": "ui_tab",
        "z": "",
        "name": "Customer Care Desk",
        "icon": "dashboard",
```

```
        "disabled": false,
        "hidden": false
    },
    {
        "id": "23c01f98.adefd",
        "type": "ui_group",
        "z": "",
        "name": "Customer query",
        "tab": "23556120.70ecde",
        "order": 1,
        "disp": true,
        "width": 6,
        "collapse": false
    },
    {
        "id": "f33b8e85.e34bd",
        "type": "ui_base",
        "theme": {
            "name": "theme-dark",
            "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": true,
      "reset": 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": "#097479",
        "value": "#097479",
        "edited": false
      },
      "page-titlebar-backgroundColor": {
        "value": "#097479",
        "edited": false
      },
      "page-backgroundColor": {
        "value": "#111111",
```

```
    "edited": false
},
"page-sidebar-backgroundColor": {
    "value": "#000000",
    "edited": false
},
"group-textColor": {
    "value": "#0eb8c0",
    "edited": false
},
"group-borderColor": {
    "value": "#555555",
    "edited": false
},
"group-backgroundColor": {
    "value": "#333333",
    "edited": false
},
"widget-textColor": {
    "value": "#eeeeee",
    "edited": false
},
"widget-backgroundColor": {
    "value": "#097479",
    "edited": false
},
"widget-borderColor": {
```

```
            "value": "#333333",
            "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": "8055355a.c51b18",
    "type": "ui_group",
    "z": "",
    "name": "Bot Response",
    "tab": "23556120.70ecde",
    "order": 2,
    "disp": true,
    "width": 15,
    "collapse": false
},
{
    "id": "f2f2649a.0d0d98",
    "type": "debug",
    "z": "da205540.e2f438",
    "name": "",
    "active": true,
    "console": "false",
    "complete": "false",
    "x": 830,
```

```
    "y": 80,
    "wires": []
},
{
    "id": "e19e969d.7e5f28",
    "type": "ui_form",
    "z": "da205540.e2f438",
    "name": "",
    "label": "",
    "group": "23c01f98.adefd",
    "order": 1,
    "width": 0,
    "height": 0,
    "options": [
        {
            "label": "Enter the input",
            "value": "input",
            "type": "text",
            "required": true,
            "rows": null
        }
    ],
    "formValue": {
        "input": ""
    },
    "payload": "",
    "submit": "submit",
```

```
    "cancel": "cancel",
    "topic": "",
    "x": 90,
    "y": 360,
    "wires": [
        [
            "90365fb6.218f1"
        ]
    ]
},
{
    "id": "90365fb6.218f1",
    "type": "function",
    "z": "da205540.e2f438",
    "name": "func 1",
    "func": "msg.payload=msg.payload.input;\nreturn msg;",
    "outputs": 1,
    "noerr": 0,
    "x": 250,
    "y": 260,
    "wires": [
        [
            "1c468b56.c66165",
            "24a18ce.1ccff74"
        ]
    ]
},
```

```
    {
        "id": "1c468b56.c66165",
        "type": "ui_text",
        "z": "da205540.e2f438",
        "group": "23c01f98.adefd",
        "order": 2,
        "width": 0,
        "height": 0,
        "name": "",
        "label": "Input",
        "format": "{{msg.payload}}",
        "layout": "col-center",
        "x": 370,
        "y": 440,
        "wires": []
    },
    {
        "id": "24a18ce.1ccff74",
        "type": "watson-conversation-v1",
        "z": "da205540.e2f438",
        "name": "Assistant",
        "workspaceid": "8d6f7af1-729e-4ebb-8cf8-f10b00c00e6d",
        "multiuser": false,
        "context": false,
        "empty-payload": false,
        "service-endpoint":
"https://api.eu-gb.assistant.watson.cloud.ibm.com/instances/9d9f9fd1-
```

```
1955-4f84-9eaa-d56677ae5d1f",
        "timeout": "",
        "optout-learning": false,
        "x": 460,
        "y": 180,
        "wires": [
            [
                "f2f2649a.0d0d98",
                "75df196e.c9a928"
            ]
        ]
    },
    {
        "id": "75df196e.c9a928",
        "type": "function",
        "z": "da205540.e2f438",
        "name": "func 2",
        "func":
"msg.payload.text=\"\";\nif(msg.payload.context.webhook_result_1){\n
for(var i in msg.payload.context.webhook_result_1.results){\n
msg.payload.text=msg.payload.text+\"\\n\"+msg.payload.context.webh
ook_result_1.results[i].text;\n    }\n
msg.payload=msg.payload.text;\n}\nelse{\n    msg.payload =
msg.payload.output.text[0];\n}\n\nreturn msg; ",
        "outputs": 1,
        "noerr": 0,
        "x": 660,
```

```
        "y": 320,
        "wires": [
            [
                "b945629b.d3d94"
            ]
        ]
    },
    {
        "id": "b945629b.d3d94",
        "type": "ui_text",
        "z": "da205540.e2f438",
        "group": "8055355a.c51b18",
        "order": 1,
        "width": "15",
        "height": "14",
        "name": "",
        "label": "output",
        "format": "{{msg.payload}}",
        "layout": "col-center",
        "x": 870,
        "y": 400,
        "wires": []
    }
]
```