



# Project Report

## Intelligent Customer Helpdesk with Smart Document Understanding

*Project Report submitted to fulfill the requisites for the award of Smart Bridge Internship Certificate*

**Work Submitted by :**

N Pavan kumar

## Table of Contents

Project Details	-----	3
acknowledgment	-----	4
1. Introduction	-----	5
a. Overview		
b. Purpose		
2. Literature Survey	-----	7
a. Existing Problem		
b. Proposed Solutio		
3. Theoretical Analysis	-----	8
a. Block Diagram		
b. Hardware/Software Designing		
4. Experimental Investigations	-----	10
5. Flowchart	-----	15
6. Result	-----	16
7. Advantages and Disadvantages	-----	17
8. Applications	-----	17
9. Conclusion	-----	18
10. Future Scope	-----	19
11. Bibliography	-----	19
Appendix	-----	19
A. Source code	-----	

## Project Details

<b>Project ID</b>	:	SPS_PRO_99 IISPS_INT_606
<b>Project Title</b>	:	Intelligent Customer Helpdesk With Smart Document Understanding
<b>Duration</b>	:	4 Weeks
<b>Project Manager</b>	:	Pavan Kumar Nalajala
<b>Project Support</b>	:	The SmartBridge Educational Services
<b>Project Mentor</b>	:	Mr. Hemant Kumar Gahlot Mr. Dugaprasad
<b>Kickoff Date</b>	:	May 5th, 2020
<b>Finish Date</b>	:	June 4th, 2020

## ACKNOWLEDGEMENT

This project has taken a considerable amount of time and resources. I would like to acknowledge the help of all of those who have made this project possible. In final I would like to thank my supervisor Mr. Hemant Kumar Gahlot for his time, patience and guidance, and also for allowing the idea to be pursued primitively. I would also like to thank Mr. Durgaprasad for his help. Further to these people I would like to thank the members of the Smartbridge career workshop for their technical help in setting up various codes and faults. Also, I would like to thank all of my co-interns who have worked on the Open Source projects without whose efforts this project would not have been possible.

# 1. Introduction

## 1.1 Overview

Build a chatbot that uses various Watson AI Services (Watson Discovery, Watson Assistant, Watson Cloud Functions and Node-Red) to deliver an effective Web based UI through which we can chat with the assistant. Also integrate the Watson Discovery service with Watson Assistant using webhooks.

The main objective of the project is to create an Intelligent Customer Help Desk with smart document understanding using different AI Services (Watson Discovery, Watson Assistant, Cloud function and Node Red). With this project, one will get an insight on how to build interactive information retrieval systems by combining various Watson Services.

### **Project Scope: -**

- Project Requirements : Node-RED, IBM Cloud, IBM Watson, Node JS
- Functional Requirements : IBM Cloud
- Technical Requirements : AI, Watson AI, Node JS
- Software Requirements : Watson Assistant, Watson Discovery, Cloud Functions  
Node-red
- Project Deliverables : Intelligent Chatbot with Smart  
Document Understanding
- Project Team : Yashika Mendhekar
- 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. So unless and until customer specifically asks for a customer representative the bot will try to solve all your queries.

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. Then using Watson actions as webhook, Watson Discovery can be integrated with Watson assistant. Finally using Node-Red, Watson assistant can be integrated with a web UI. This UI can then be used to connect with Watson assistant and chat with it.

### **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

The problem that we are here trying to solve is to ensure minimum involvement of customer agent ,hereby giving satisfactory answers to all of the customer's questions with the help of an AI powered chatbot without having to return a recorded statement of "would you like to connect with an agent?".

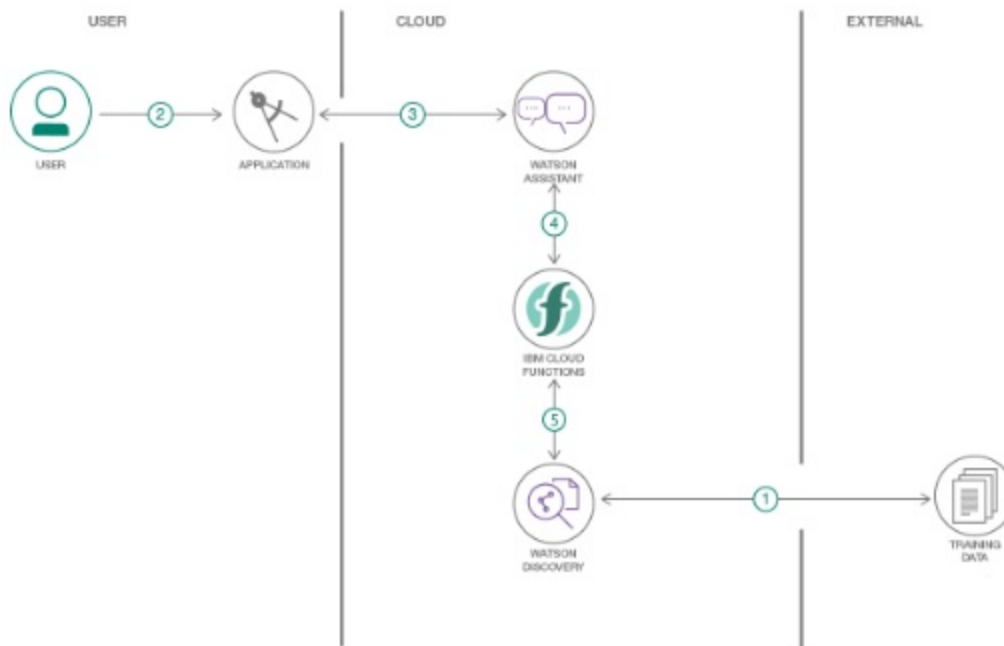
The typical customer care chatbot can answer simple questions, such as store locations and hours, directions, and maybe even making appointments. When a question falls outside of the scope of the pre-determined question set, the option is typically to tell the customer the question isn't valid or offer to speak to a real person.

### 2.2 Proposed Solution

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. So, unless and until customer specifically asks for a customer representative the bot will try to solve all your queries. 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.

# 3. Theoretical Analysis

## 3.1 Block / Flow Diagram



## 3.2 Hardware / Software Designing

1. Create necessary Watson Services.
2. Configure Watson Discovery.
3. Create Watson Cloud Functions Action.
4. Configure Watson Assistant.
5. Integrate Watson Discovery with Watson Assistant using webhook.
6. Build Node-RED flow to integrate Watson Assistant and Web Dashboard.



# 4. EXPERIMENTAL INVESTIGATIONS

## 4.1 CREATE IBM SERVICES

*a. Watson Discovery*

*b. Watson Assistant*

*c. Node-Red*

The screenshot displays the IBM Cloud Resource List page. The browser tabs include 'Student Dashboard', 'IISPS\_JNT\_606\_Intelligent Custom...', 'Slack | ! Hemant Kumar Gahlot |', and 'Resource list - IBM Cloud'. The URL is 'cloud.ibm.com/resources'. The page header shows 'IBM Cloud' and a search bar. The main content area is a table with columns: Name, Group, Location, Status, and Tags. The table lists several services, including 'Cloud Foundry services (1)', 'Services (4)', 'Continuous Delivery', 'Discovery-2n', 'My Assistant', 'global-connect-cloudant-1591069983203', 'Storage (0)', 'Network (0)', 'Cloud Foundry enterprise environments (0)', 'Functions namespaces (0)', 'Apps (1)', and 'Global Connect'. The 'My Assistant' service is highlighted. The 'Status' column shows 'Active' for most services. The 'Tags' column shows 'vers...' for some services. A 'FEEDBACK' button is visible on the right side of the table.

Name	Group	Location	Status	Tags
Cloud Foundry services (1)				
Services (4)				
Continuous Delivery	Default	Dallas	Active	—
Discovery-2n	Default	Dallas	Active	vers...
My Assistant	Default	Dallas	Active	vers...
global-connect-cloudant-1591069983203	Default	Chennai 01	Active	—
Storage (0)				
Network (0)				
Cloud Foundry enterprise environments (0)				
Functions namespaces (0)				
Apps (1)				
Global Connect	Default	Global	—	vers...

## 4.2 Configuration of Watson discovery

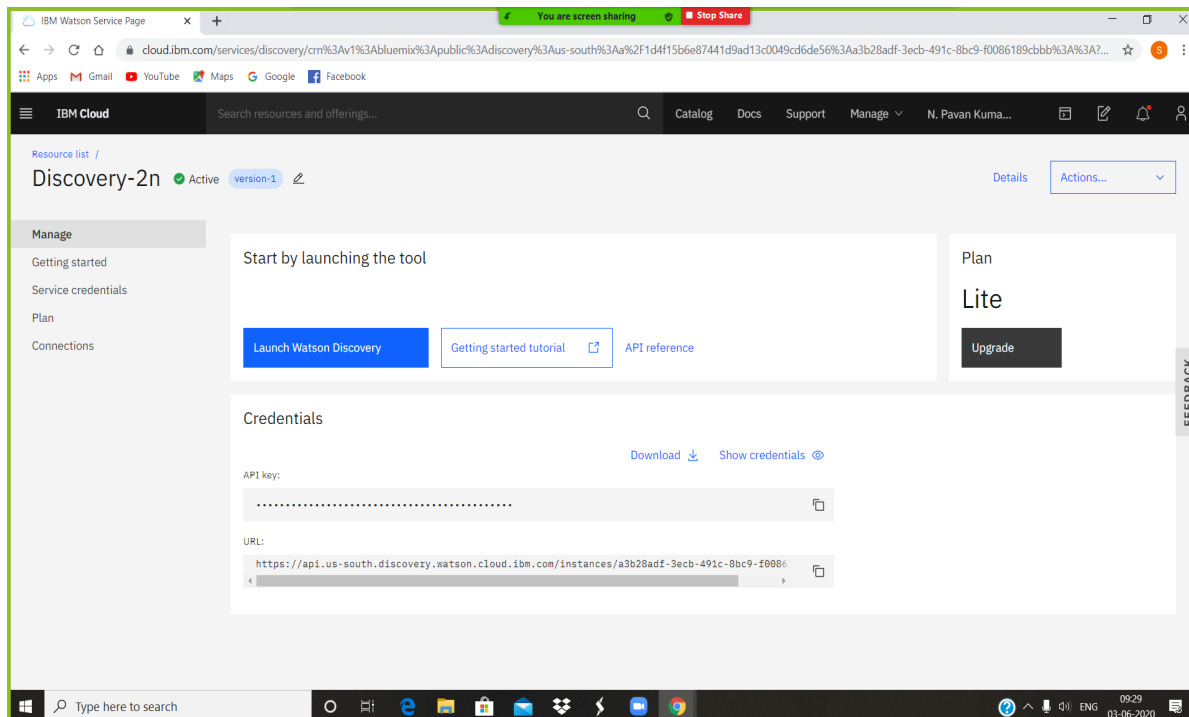
After creating and launching the discovery from the resources we have to attach the document i.e,ecobee3\_UserGuide document from the local documents.

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 the SMART DOCUMENT the user can get the accurate result.This can be done easily by clicking on the configure setting and the labelling each word or element present in the document as their respective label such as title, subtitle,text,images 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.

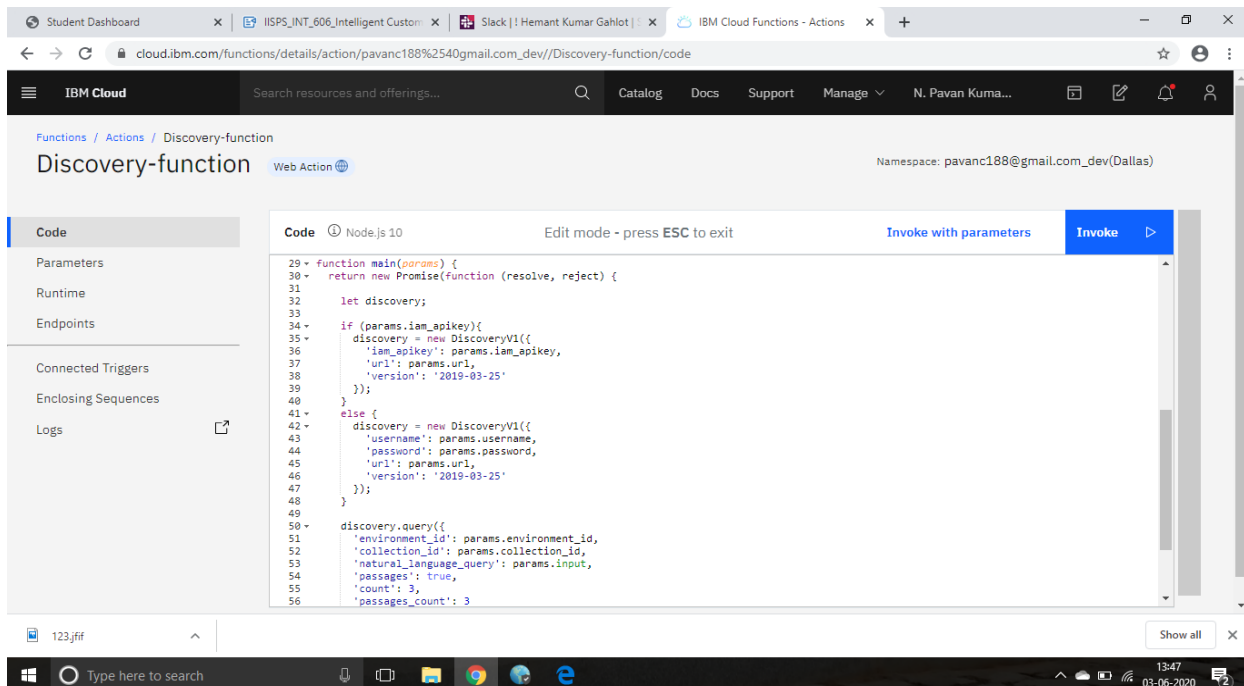
### **follow the below mentioned steps:**

-> 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.

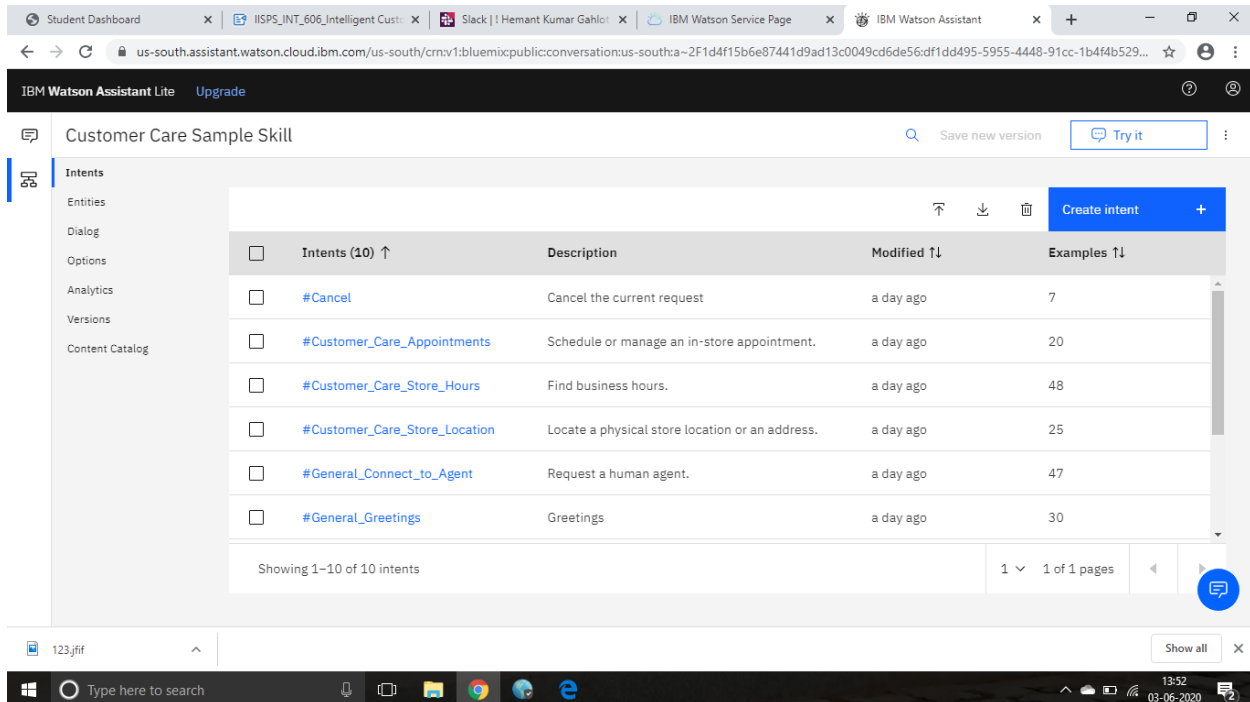
These credentials will be used in further steps.



## 4.3 Integrate Discovery With Cloud Functions



## 4.4 Creating Watson Assistant Skill

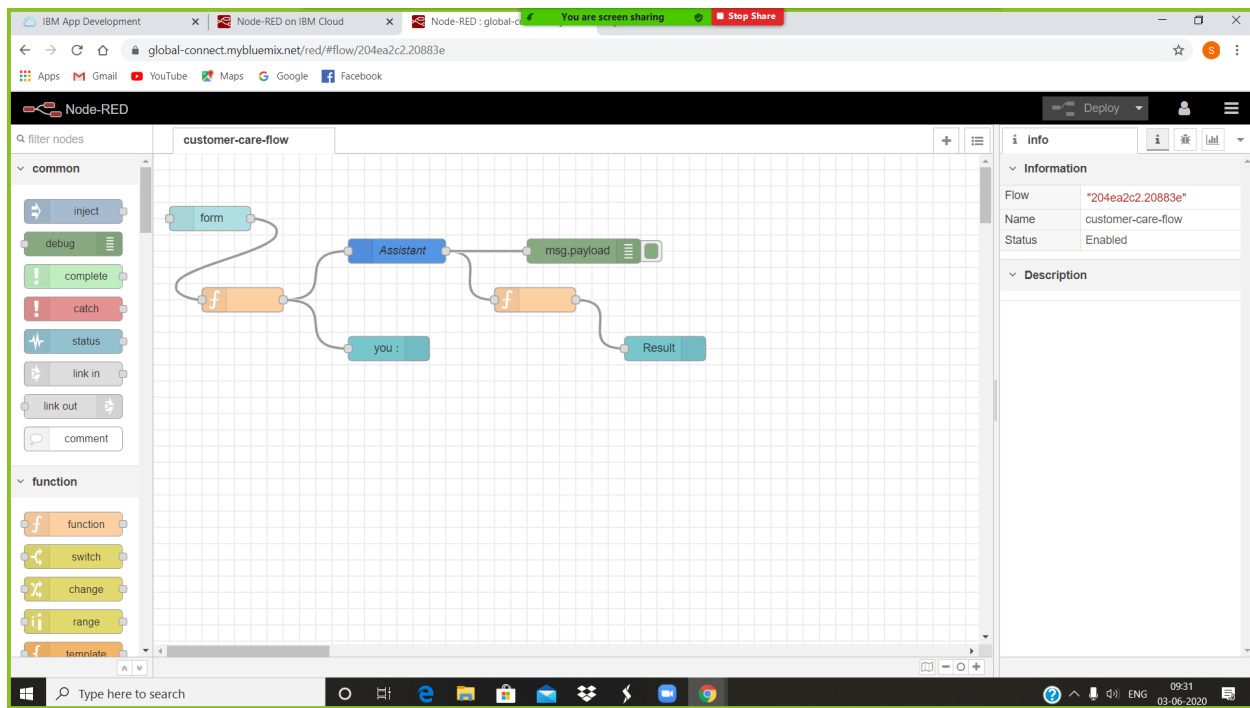


The screenshot displays the IBM Watson Assistant interface for a skill named "Customer Care Sample Skill". The left sidebar contains navigation options: Intents, Entities, Dialog, Options, Analytics, Versions, and Content Catalog. The main area shows a table of 10 intents. The table has columns for a checkbox, the intent name, a description, the modification time, and the number of examples.

<input type="checkbox"/>	Intents (10) ↑	Description	Modified ↑↓	Examples ↑↓
<input type="checkbox"/>	#Cancel	Cancel the current request	a day ago	7
<input type="checkbox"/>	#Customer_Care_Appointments	Schedule or manage an in-store appointment.	a day ago	20
<input type="checkbox"/>	#Customer_Care_Store_Hours	Find business hours.	a day ago	48
<input type="checkbox"/>	#Customer_Care_Store_Location	Locate a physical store location or an address.	a day ago	25
<input type="checkbox"/>	#General_Connect_to_Agent	Request a human agent.	a day ago	47
<input type="checkbox"/>	#General_Greetings	Greetings	a day ago	30

Showing 1-10 of 10 intents

## 4.5 Designing Node Flow using Node-Red



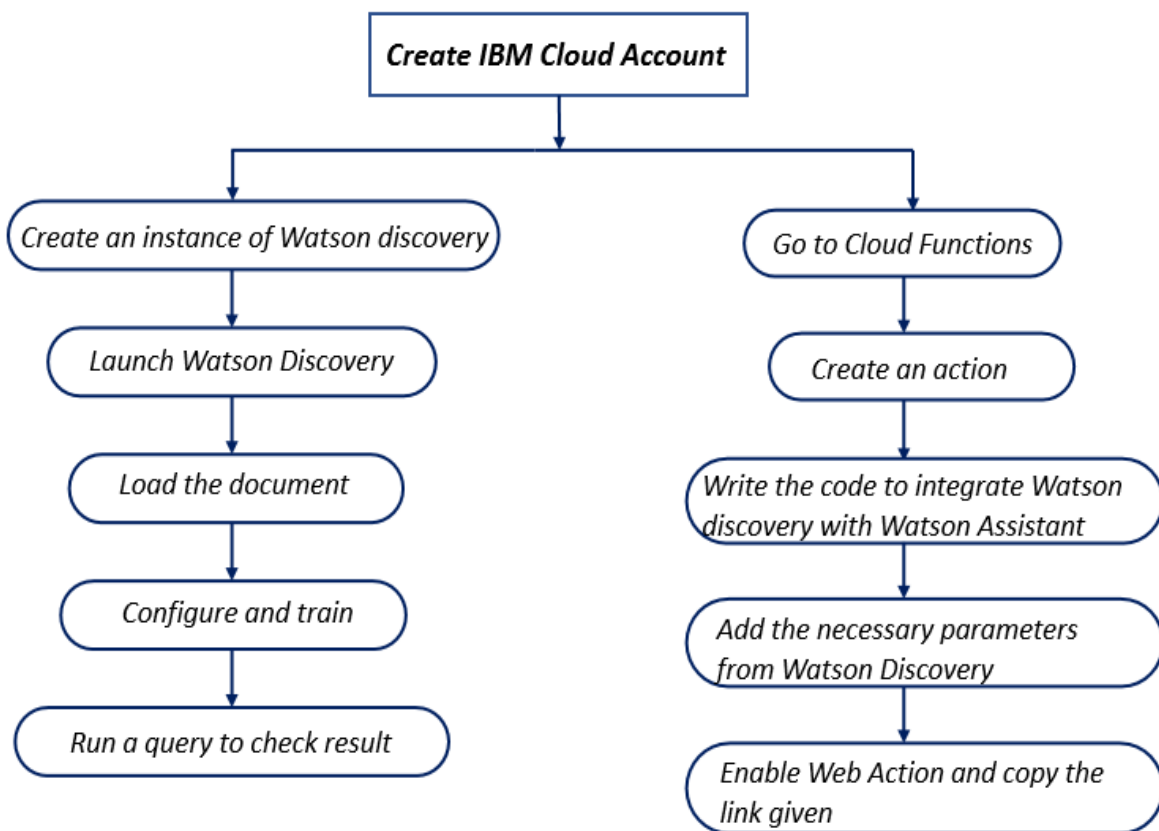
The screenshot shows the Node-RED interface with a flow named "customer-care-flow". The flow starts with a "form" node, which connects to a "you:" node. The "you:" node connects to an "Assistant" node. The "Assistant" node connects to a "msg.payload" node, which then connects to a "Result" node. The interface includes a sidebar with "common" and "function" node categories, a top bar with "Deploy" and "Stop Share" buttons, and a right sidebar with "Info" and "Description" tabs.

## 5.Flow Chart

### 5.1 Watson Discovery And Cloud Functions

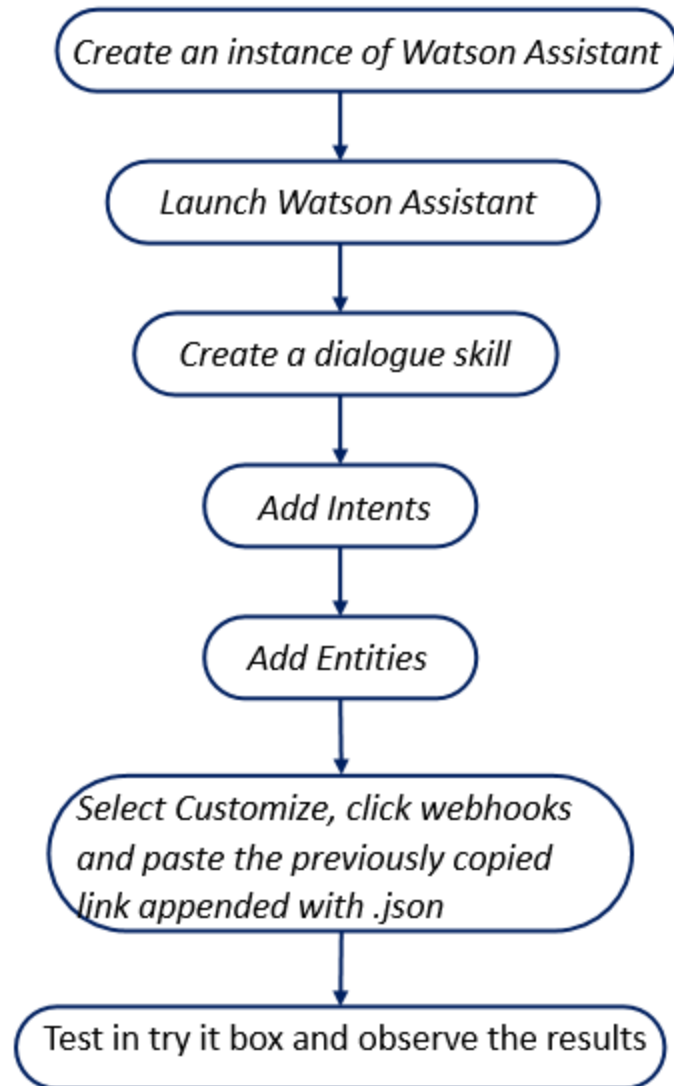
The Flow Chart given below will Guide us to

- Create a Watson Discovery Service
- Load And Train the Sample Data
- Integrating Discovery with Assistant Using IBM Cloud Functions



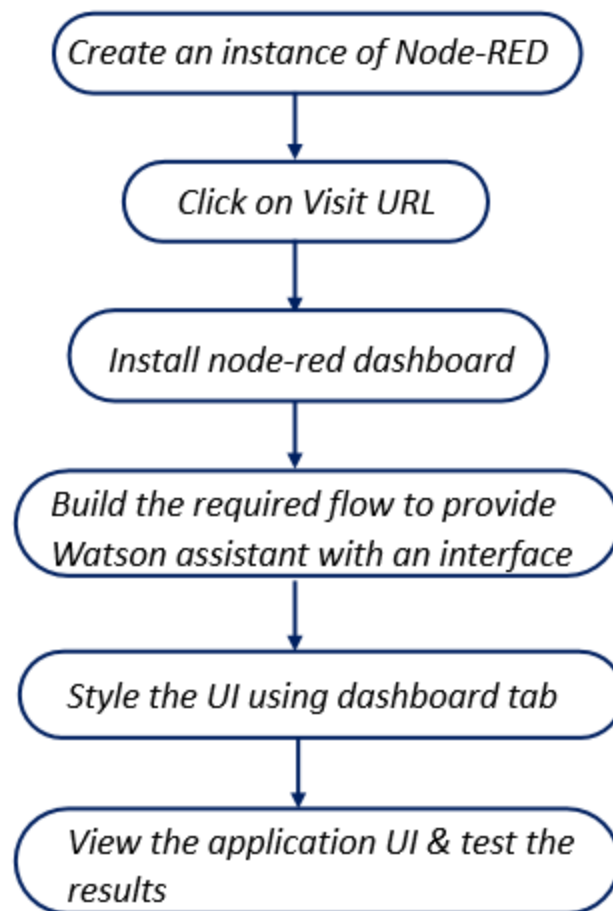
## 5.2 Watson Assistant

The Flow Chart Given below Will help us to Create the Customer Care Dialog Skill And adding intents to that skill.

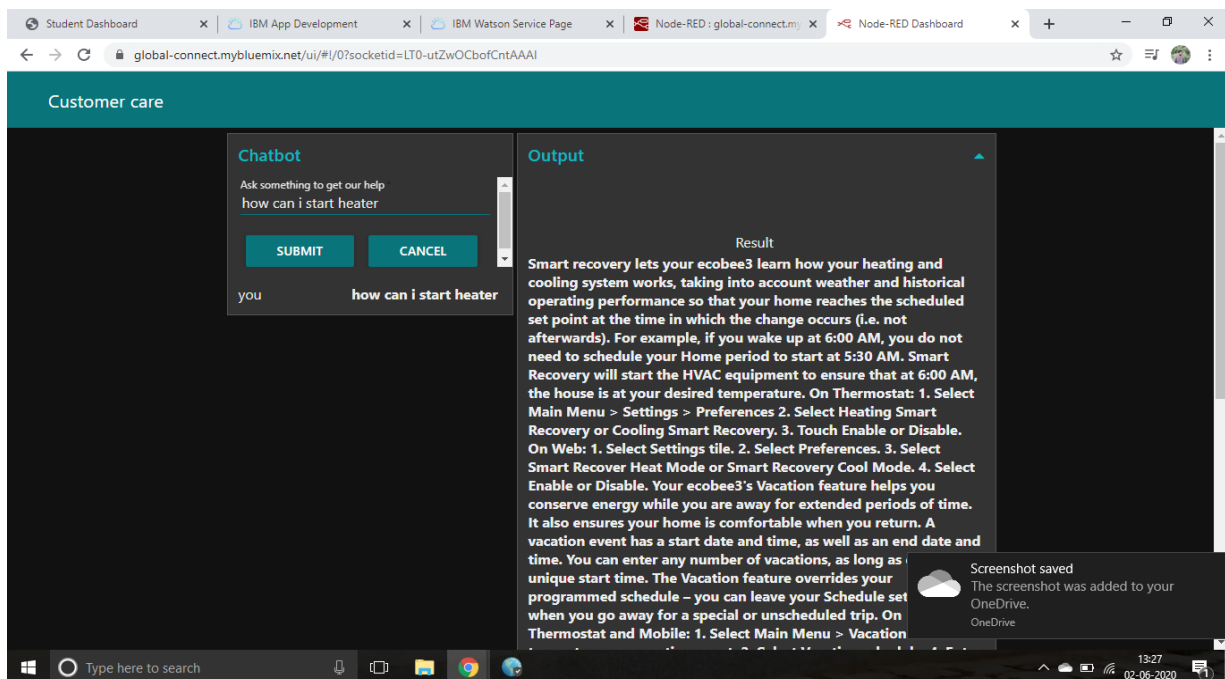
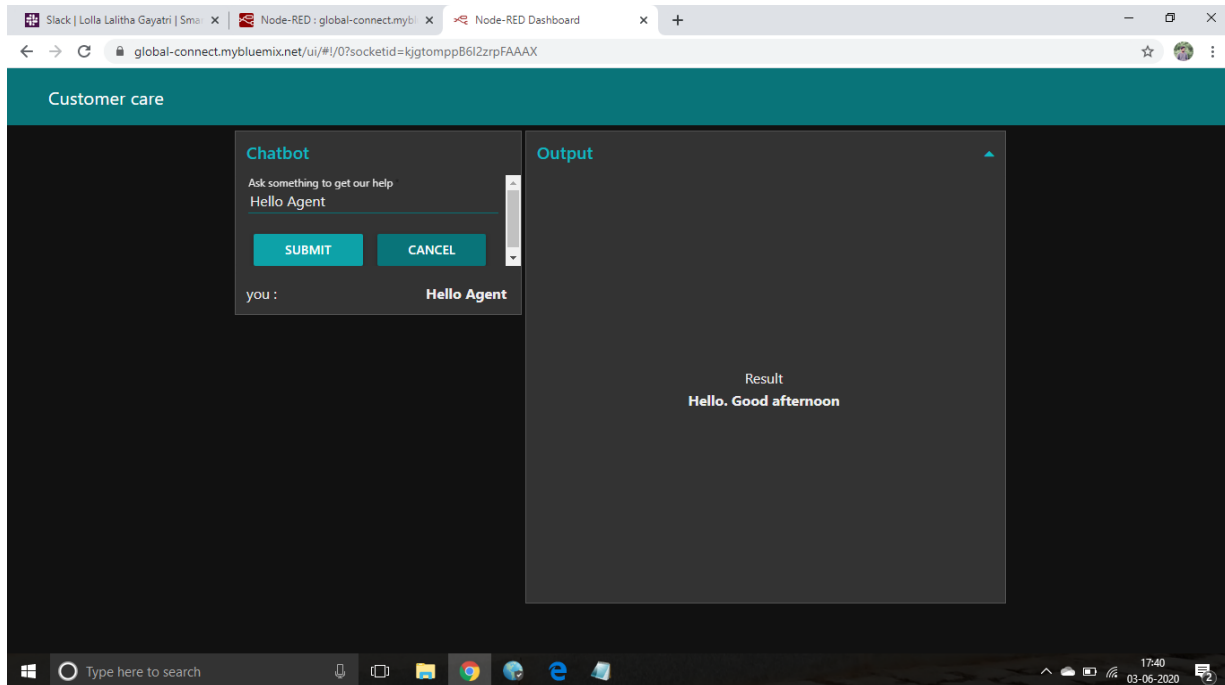


## 5.3 Node Red

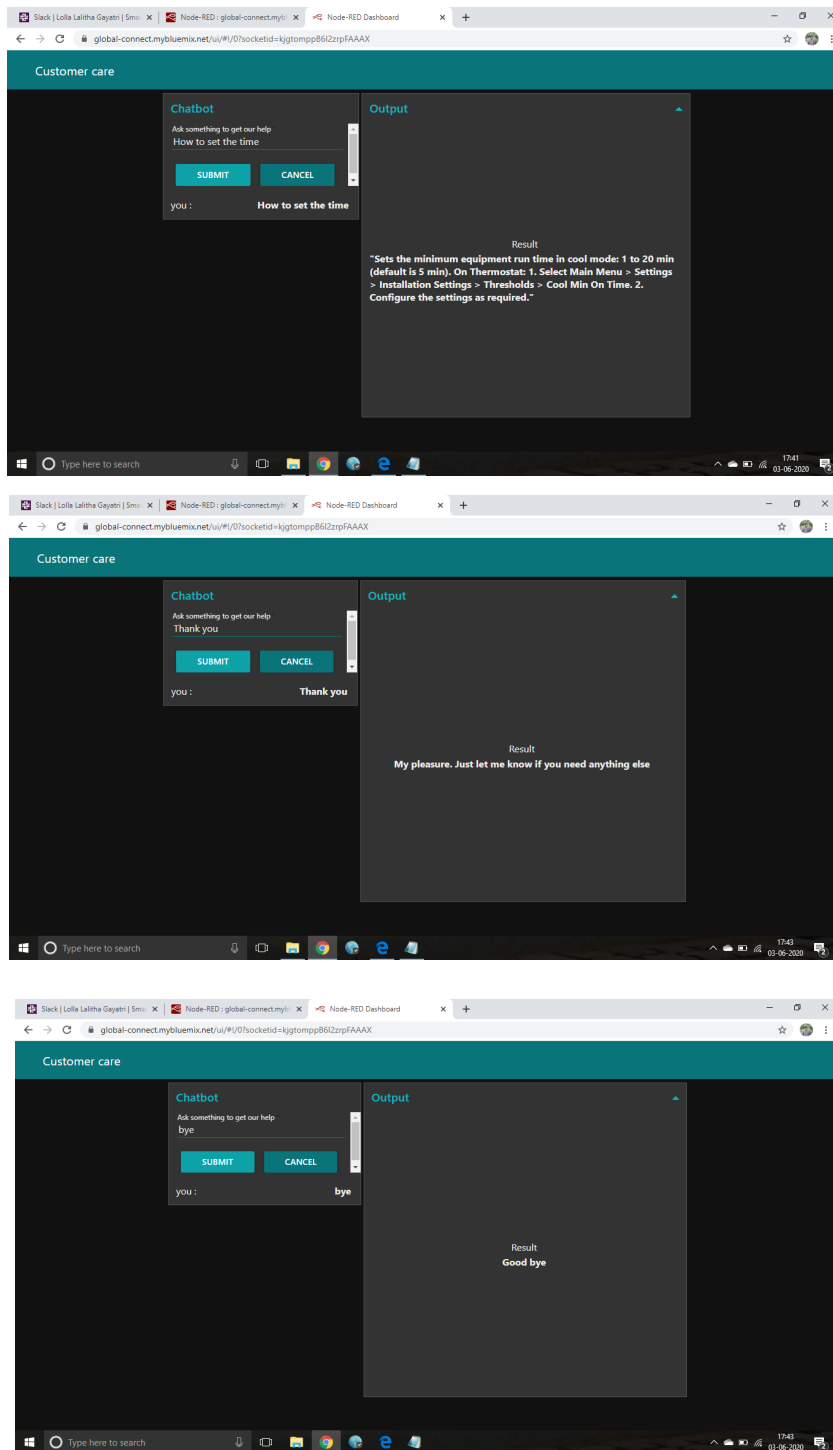
The Given Flow Chart will Guide us in creating Node Red ui and adding Watson Assistant API details to the flow.



## 6. Results







Web based UI was developed by integrating all the services using Node-RED.

URL for UI Dashboard : <https://global-connect.mybluemix.net/ui>

# 7. Advantages and Disadvantages

## 7.1 Advantages

- Companies can deploy chatbots to rectify simple and general human queries.
- Reduces man power .
- Cost efficient.
- No need to divert calls to customer agents and customer agents can look at other works.
- 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.

## 7.2 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.
- The data is not trained properly then the result will not be accurate.
- It may also give same answers to different queries.
- Training a chat bot to work efficiently is a tedious task.

## 8. Applications

- Used in chat applications like Facebook, Messenger, Slack and Telegram.
- Provide Customer Service.
- 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.
- This chatbot can be deployed to various websites as it can solve a lot of basic questions.
- It can be used to deploy as Customer Helpdesk for small scale products as their manual usually has the solution for the user's problems.
- The primary function of the chatbot is to be a virtual companion – To speak with senior people on general topics like the weather, nature, hobbies, movies, music, news, etc.

## 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. An Intelligent Customer Helpdesk Chatbot was created using various Watson services like Watson Discovery, Watson Assistant, Watson Cloud Functions and Node-RED. This internship proved to be fruitful and I thoroughly enjoyed every part of this journey.

## 10. Future Scope

In the future, various other Watson services like Text-To-Speech and Speech-To-Text can be integrated in the chatbot. This can make the chatbot Hands-free. 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.

We can improve our model by including more intents and entities in the Watson dialogue skill. Also we can add speech to text and text to speech services to improve customer experience.

## 11. Bibliography

1. Node-RED Starter Application:

<https://developer.ibm.com/tutorials/how-to-create-a-node-red-starter-application/>

2. Build your own AI assistant:

<https://www.youtube.com/watch?v=hitUOFNne14>

3. How to use Watson Assistant with Webhooks:

<https://www.youtube.com/embed/5z3i5lsBVnk>

4. Watson Discovery:

<https://developer.ibm.com/articles/introduction-watson-discovery/>

[https://www.ibm.com/cloud/architecture/tutorials/cognitive\\_discovery](https://www.ibm.com/cloud/architecture/tutorials/cognitive_discovery)

5. GitHub

<https://github.com/IBM/watson-discovery-sdu-with-assistant>

# Source Code

## Cloud Functions-Actions

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

```

## Node Red-Nodes

Input Function      - msg.payload=msg.payload.input;  
                           return msg;

Output Function    - msg.payload=msg.payload.output.text[0];  
                      return msg;

## Node Flow

```
[
  {
    "id": "204ea2c2.20883e",
    "type": "tab",
    "label": "customer-care-flow",
    "disabled": false,
    "info": ""
  },
  {
    "id": "c0b9c54c.1b3468",
    "type": "ui_form",
    "z": "204ea2c2.20883e",
    "name": "",
    "label": "",
    "group": "de455d1d.ba9e3",
    "order": 1,
    "width": 6,
    "height": 2,
    "options": [
      {
        "label": "Ask something to get our help",
        "value": "input",
        "type": "text",
        "required": true,
        "rows": null
      }
    ],
    "formValue": {
      "input": ""
    },
    "payload": "",
    "submit": "submit",
  }
]
```

```

        "cancel": "cancel",
        "topic": "",
        "x": 70,
        "y": 80,
        "wires": [
            [
                "2c12dd6a.8a5362"
            ]
        ]
    },
    {
        "id": "2c12dd6a.8a5362",
        "type": "function",
        "z": "204ea2c2.20883e",
        "name": "",
        "func": "msg.payload = msg.payload.input;\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 110,
        "y": 180,
        "wires": [
            [
                "96bc19a8.90b628",
                "18ef277.eda6fd9"
            ]
        ]
    },
    {
        "id": "96bc19a8.90b628",
        "type": "watson-conversation-v1",
        "z": "204ea2c2.20883e",
        "name": "Assistant",
        "workspaceid": "2d8c735e-35a6-4dcc-aa3c-623cf076b40b",
        "multiuser": false,
        "context": true,
        "empty-payload": false,
        "service-endpoint":
        "https://api.us-south.assistant.watson.cloud.ibm.com/instances/df1dd495-5955-4448-

```



```

91cc-1b4f4b529dd2",
    "timeout": "",
    "optout-learning": false,
    "x": 300,
    "y": 120,
    "wires": [
        [
            "ddc9cf9.a57f43",
            "556733ec.efa41c"
        ]
    ]
},
{
    "id": "ddc9cf9.a57f43",
    "type": "function",
    "z": "204ea2c2.20883e",
    "name": "",
    "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.webhook_result_1.res
ults[i].text;\\n}\\n    msg.payload=msg.payload.text;\\n}\\nelse\\nmsg.payload =
msg.payload.output.text[0];\\nreturn msg;",
    "outputs": 1,
    "noerr": 0,
    "x": 470,
    "y": 180,
    "wires": [
        [
            "4e05526f.db39dc"
        ]
    ]
},
{
    "id": "18ef277.eda6fd9",
    "type": "ui_text",
    "z": "204ea2c2.20883e",
    "group": "de455d1d.ba9e3",
    "order": 2,

```

```
    "width": 0,
    "height": 0,
    "name": "",
    "label": "you",
    "format": "{{msg.payload}}",
    "layout": "row-spread",
    "x": 290,
    "y": 240,
    "wires": []
  },
  {
    "id": "4e05526f.db39dc",
    "type": "ui_text",
    "z": "204ea2c2.20883e",
    "group": "b349541a.e97d28",
    "order": 1,
    "width": 10,
    "height": "20",
    "name": "",
    "label": "Result",
    "format": "{{msg.payload}}",
    "layout": "col-center",
    "x": 630,
    "y": 240,
    "wires": []
  },
  {
    "id": "556733ec.efa41c",
    "type": "debug",
    "z": "204ea2c2.20883e",
    "name": "",
    "active": true,
    "tosidebar": true,
    "console": false,
    "tostatus": false,
    "complete": "false",
    "x": 530,
```

```

        "y": 120,
        "wires": []
    },
    {
        "id": "de455d1d.ba9e3",
        "type": "ui_group",
        "z": "",
        "name": "Chatbot",
        "tab": "638d1ba1.d9f494",
        "order": 1,
        "disp": true,
        "width": "6",
        "collapse": false
    },
    {
        "id": "b349541a.e97d28",
        "type": "ui_group",
        "z": "",
        "name": "Output",
        "tab": "638d1ba1.d9f494",
        "order": 2,
        "disp": true,
        "width": "10",
        "collapse": true
    },
    {
        "id": "638d1ba1.d9f494",
        "type": "ui_tab",
        "z": "",
        "name": "Customer care",
        "icon": "dashboard",
        "disabled": false,
        "hidden": false
    }
]

```