PROJECT REPORT

**Title:** Intelligent Customer Help Desk with Smart Document Understanding

**Project ID:** SPS\_PRO\_99

**Category:** Artificial Intelligence

Internship at [**smartinternz.com@2020**](mailto:smartinternz.com@2020)

**NAME:** Riya Ann Abraham

**EMAIL**: [**riyaann2001@gmail.com**](mailto:riyaann2001@gmail.com)

**CONTENTS**

**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 Hardware / Software designing

**4** EXPERIMENTAL INVESTIGATIONS

**5** FLOWCHART

**6**  RESULT

**7**  ADVANTAGES & DISADVANTAGES

**8**  APPLICATIONS

**9** CONCLUSION

**10** FUTURE SCOPE `

**11** BIBILOGRAPHY

APPENDIX

A. Source code

**1. INTRODUCTION**

**1.1 OVERVIEW**

We will build a chatbot that uses various Watson AI Services such as **Watson Discovery, Watson Assistant, Cloud Functions and NODE -RED** to deliver an effective Web based UI through which we can chat with the assistant. We will integrate the Watson Discovery service with Watson Assistant using **Webhook**.

* 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 Duration: 29 Days

**1.2 PURPOSE**

As we all know that the catalog and owner’s manuals of various products and companies are not actually needed by all the customers, so to make the task easy for the customers and find answers quickly, the chatbot can be introduced. Thus, we are using smart document

understanding technique and training our Watson Assistant to provide as with the appropriate answers.

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 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 section of the owner’s manual to help solve our customer’s 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 future, 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 action 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 contact with Watson Assistant and chat with it.

**2. LITERATURE SURVEY**

**2.1 EXISTING PROBLEM**

The typical customer care chatbot can answer simple questions, such as store location 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**

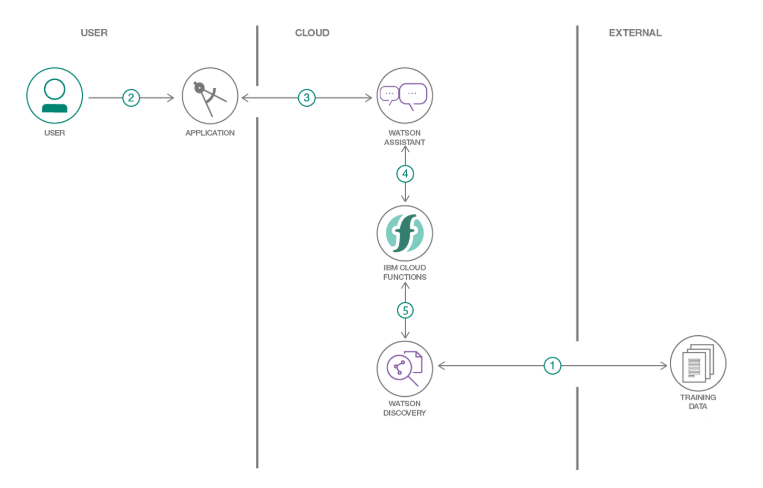
In order to achieve this problem stated above we are going to develop a smart chatbot which will answer mostly all of the customer related queries. This is done by incorporating Watson AI Service. Another benefit is that this can work for 24 hours which a normal person cannot.

**Steps:**

* 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

**3. THEORITICAL ANALYSIS**

**3.1 BLOCK / FLOW DIAGRAM**

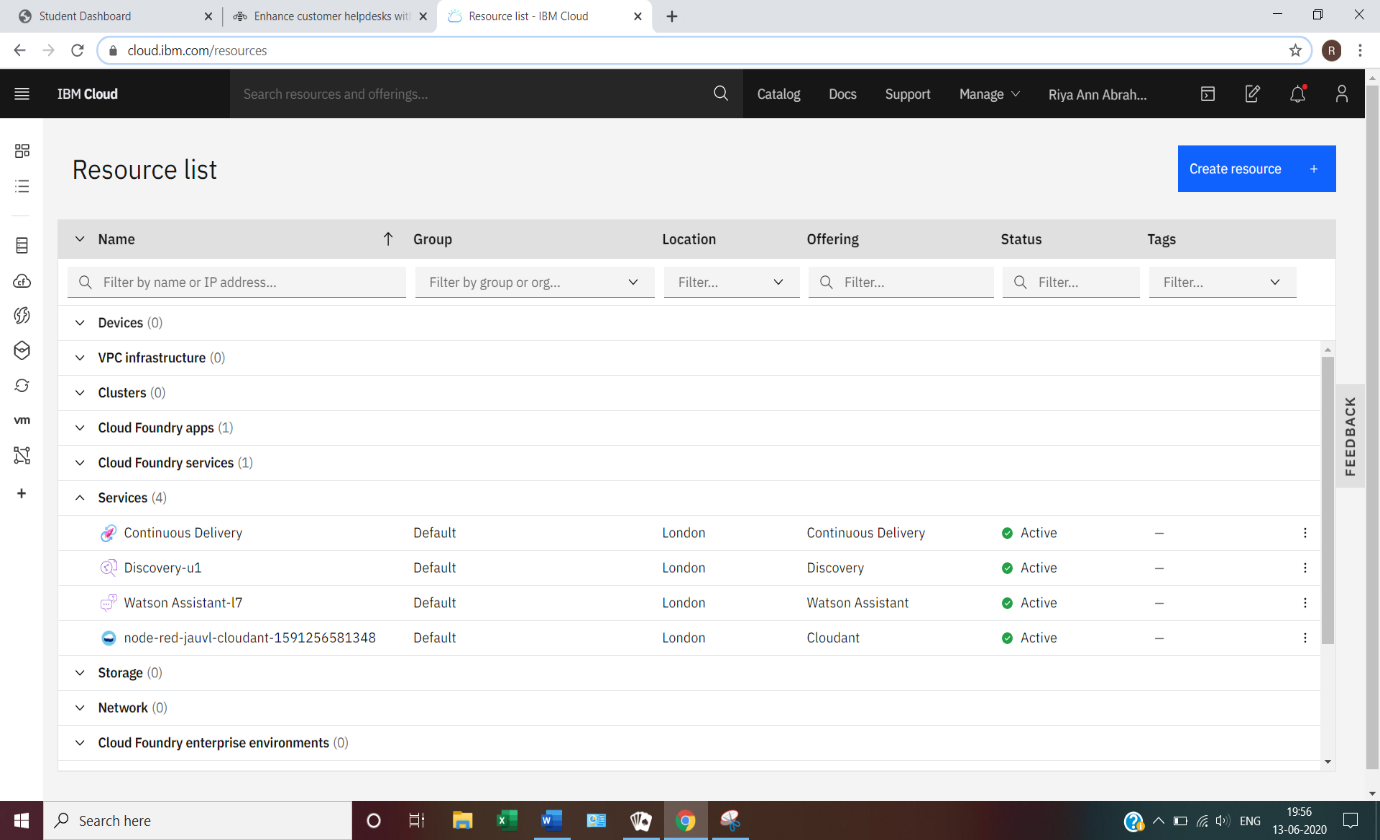


**3.2 HARDWARE / SOFTWARE DESIGNING**

The descriptive steps are as follows:

**1. Create IBM Cloud services**

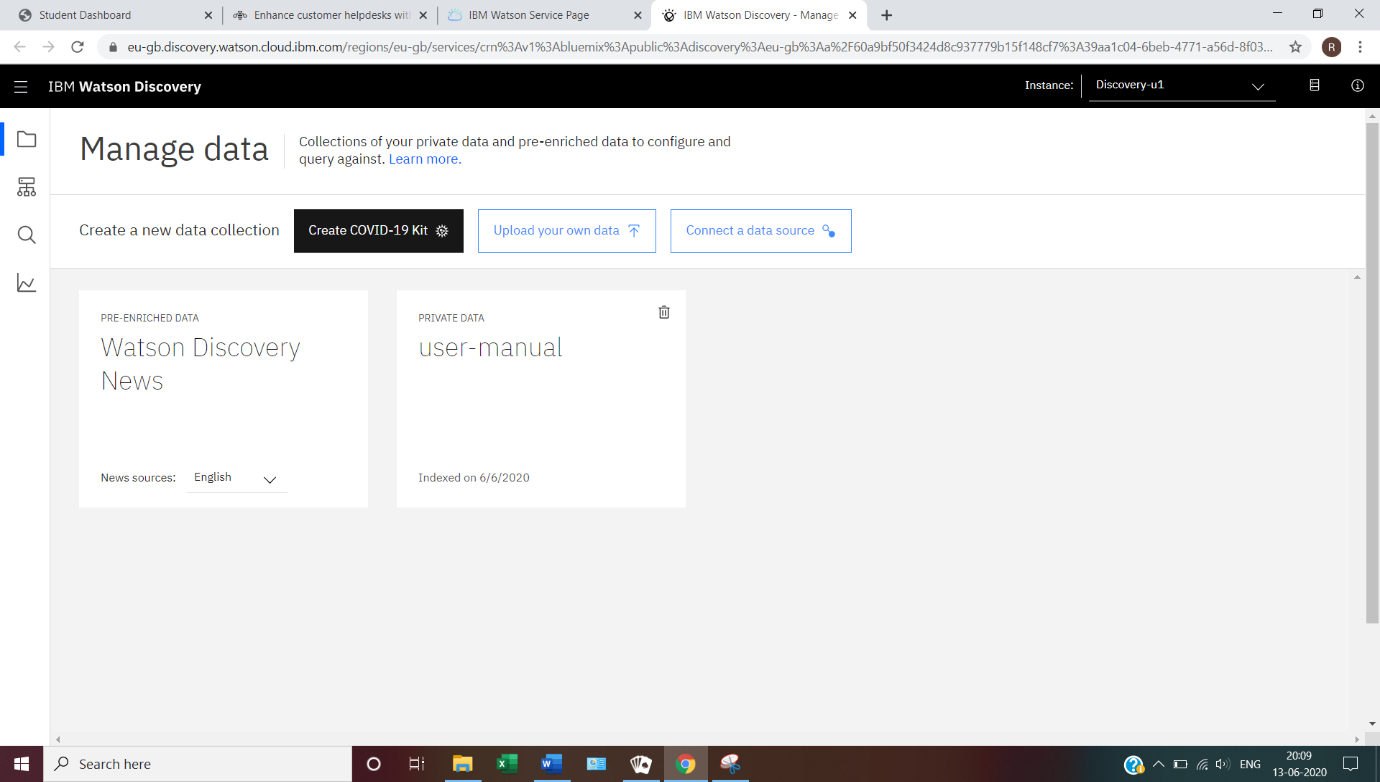
* Watson Assistant
* Watson Discovery
* Node-Red

****

**2. Configure Watson Discovery**

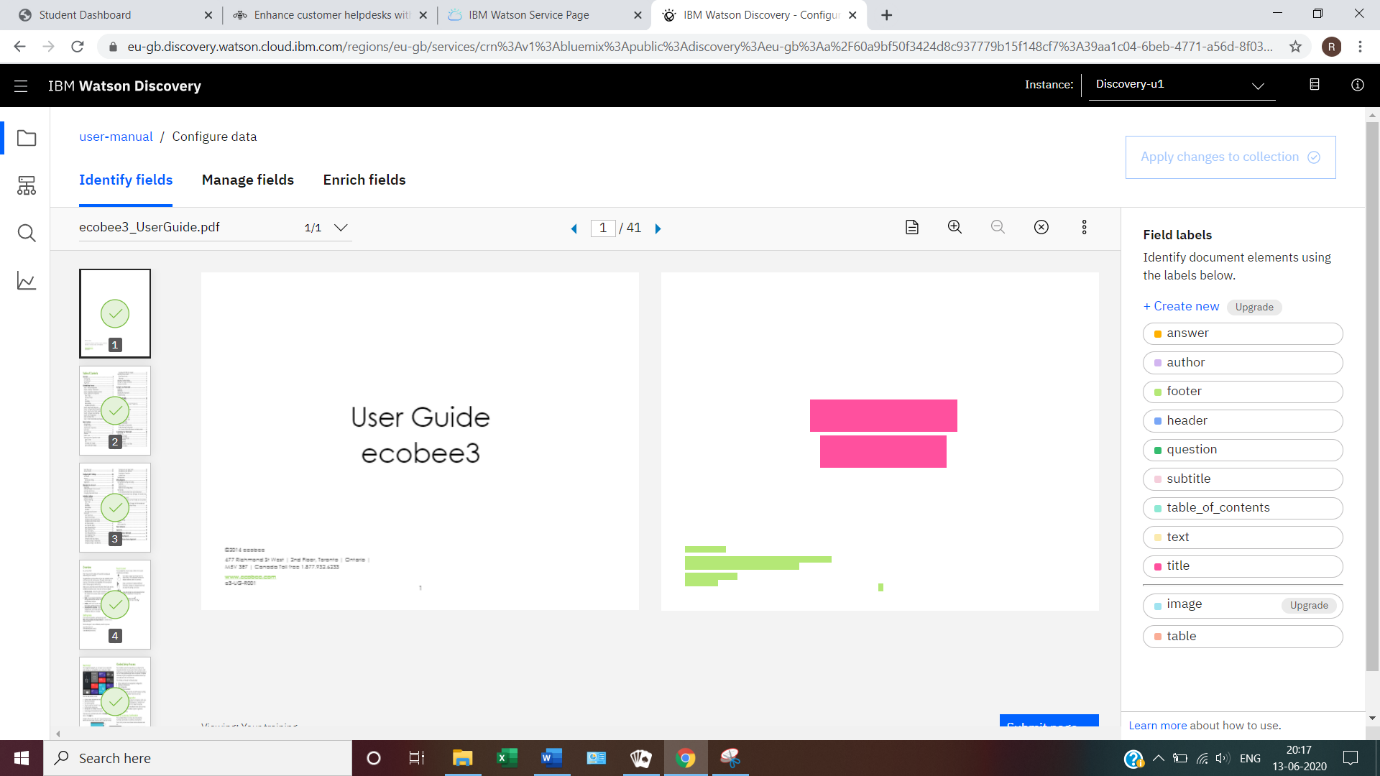
*Import the document*

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 **ecobee-3.pdf** file located in the data directory of your local repo.



*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. The goal is to annotate all of the pages in the document so that Discovery can learn what text is important and what text can be ignored.



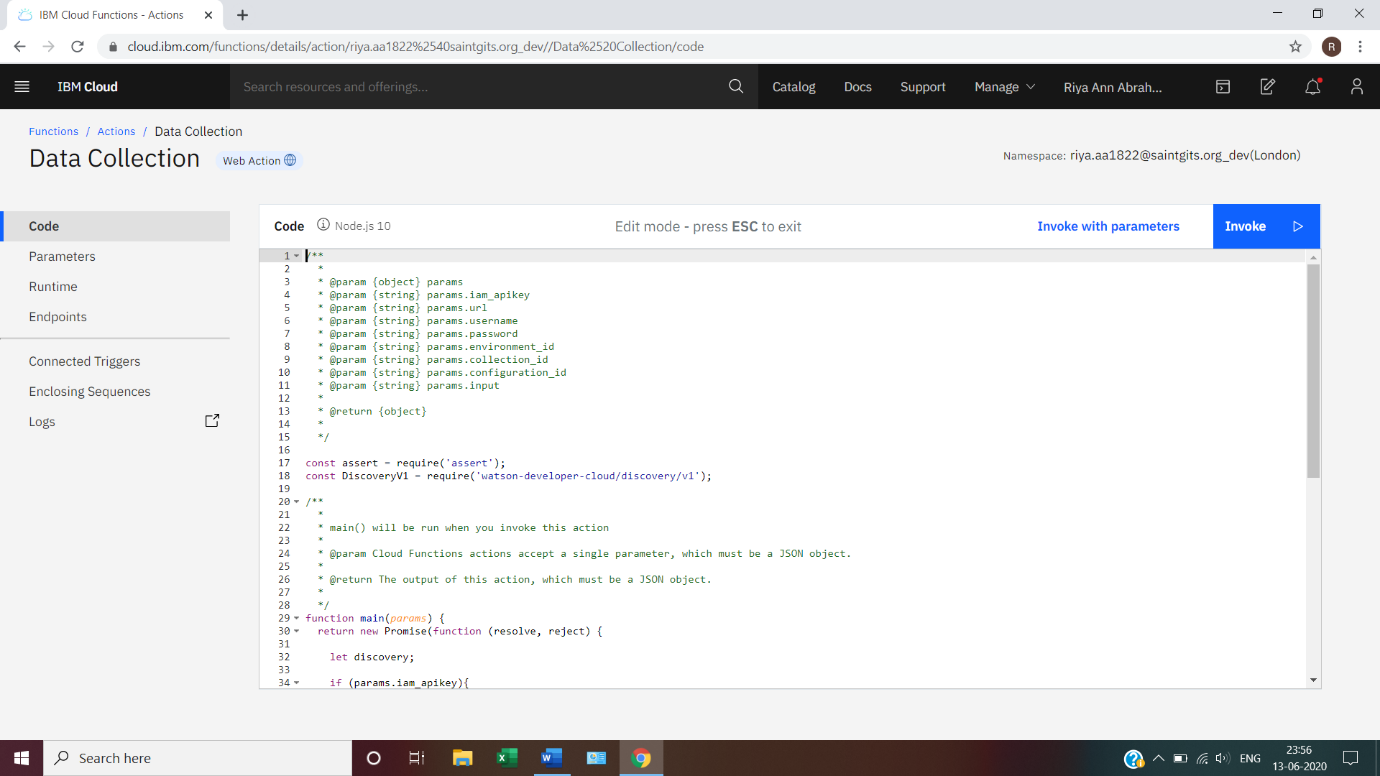
**3. Create IBM Cloud Functions Action**

Now let’s create the web action that will make queries against our Discovery collections.

* + Start the IBM Cloud Functions service by selecting **Create Resources** from the IBM Cloud Dashboard.
  + Enter functions as the filter, the select the Functions card.
  + 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, keep the default package and select the **Node.js 10** runtime.
  + Click the **Create** button to create the action
  + Once your action is created, click on the **Code tab**.
  + In the code editor window, cut and paste in the code from the disco-action.js file found in the action’s directory of your local repository. The code is pretty straight-forward. It simply connects to the Discovery Service, make a query against the collection, then returns the response.
  + If you press the **Invoke** button, it will fail due to credentials not being defined yet. We’ll do this next. Select the **Parameters** tab.

Add the following keys:

* URL
* environment\_id
* collection\_id
* iam\_apikey
* For values, please use the values associated with the Discovery services 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.
* Click the checkbox for enable as Web Action. This will generate a public endpoint URL.
* Take note of the URL value, as this will be needed by Watson Assistant.
* To verify you have entered the correct Discovery parameters, execute the provided command. If it fails, re-check your parameter values.



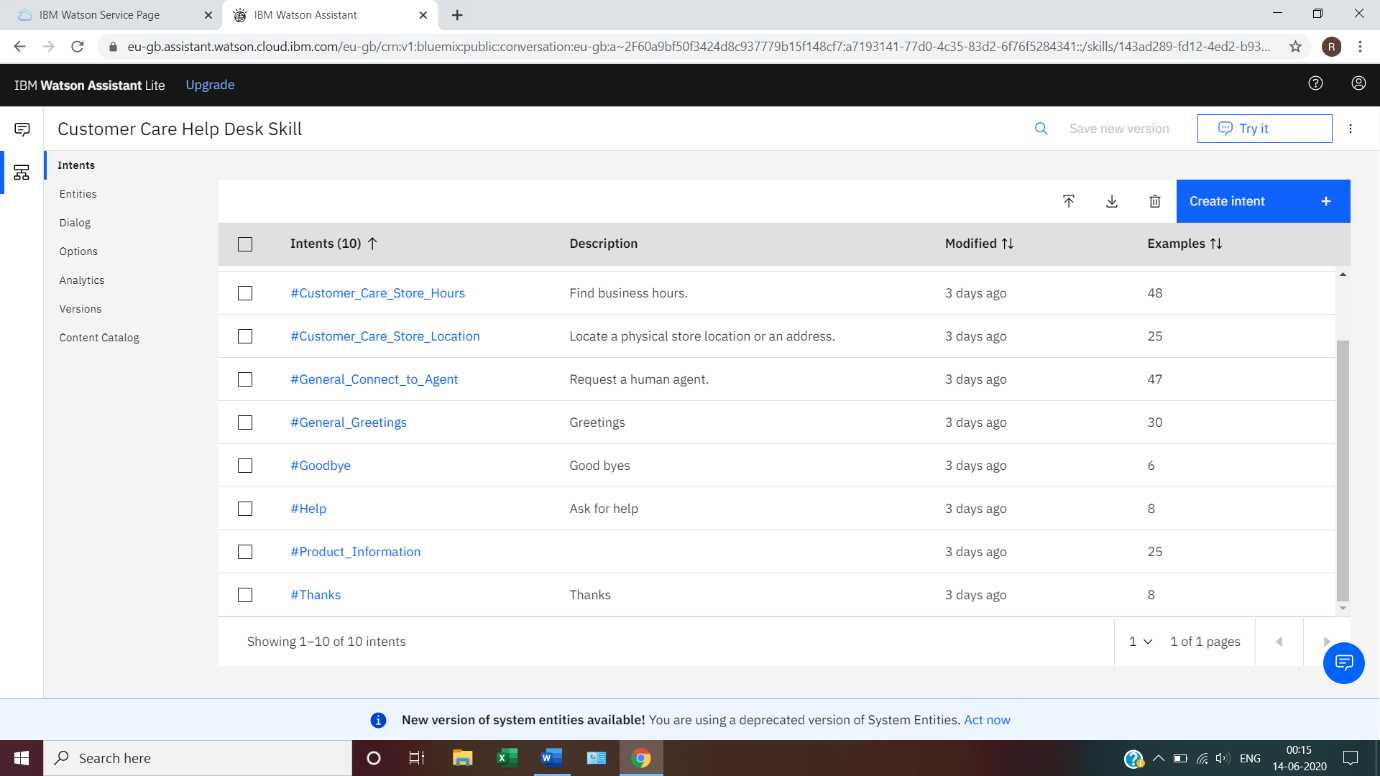
**4. Configure Watson Assistant**

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 a new intent*

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

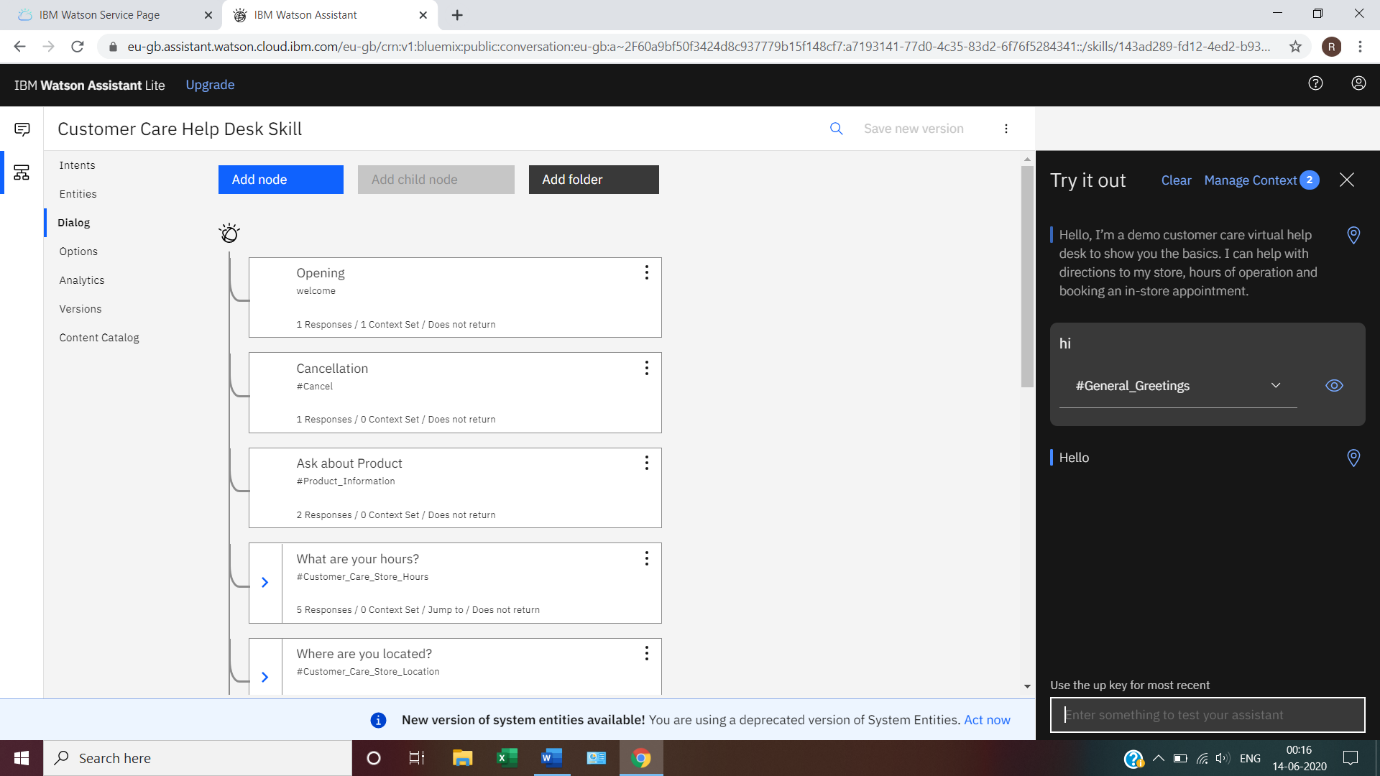
* Create a new intent that can detect when the user is asking about operating the product.
* From the Customer Care Skill panel, select the **Intents** tab.
* Click the **Create Intent** button.
* Name the intent #Product\_Information, and at a minimum, enter the following examples questions associated with it.



*Create new dialog node*

Now we need to add a node to handle our intent.

* Click on the **Dialog** tab, then click on the **drop-down** menu for the Small Talk node.
* Select the **Add node below** option.
* Name the node “Ask about product” and assign it as our new intent. 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.

* Select the **Options** tab.
* Enter the public URL endpoint for your action.
* 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 set automatically to **$webhook\_result\_1**. This is the variable name you can use to access the result from the Discovery Service query.

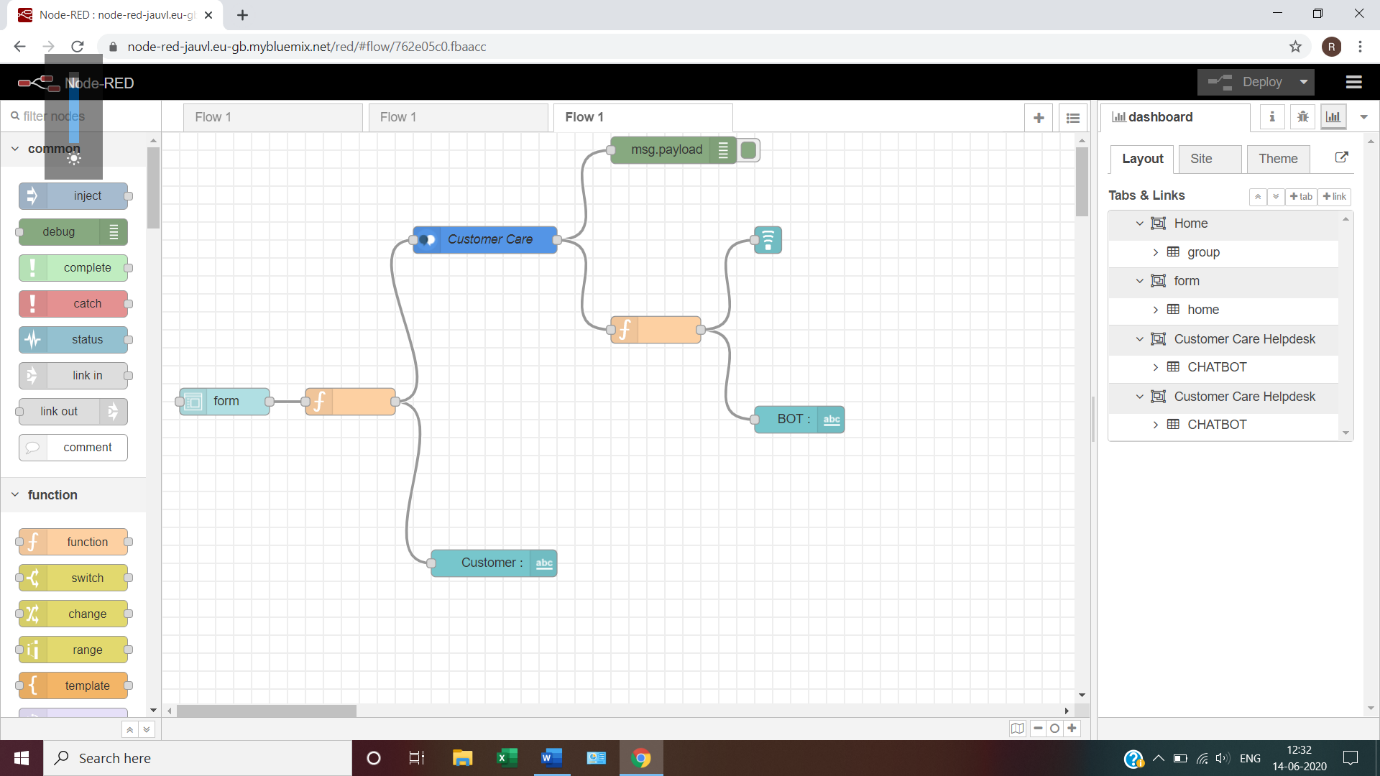
*Test in Assistant Tooling*

* From the Dialog panel, click **Try it** button located at the top right side of 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. As 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**

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

* Inject
* Assistant
* Debug
* Function
* Ui\_Form
* Ui\_Test



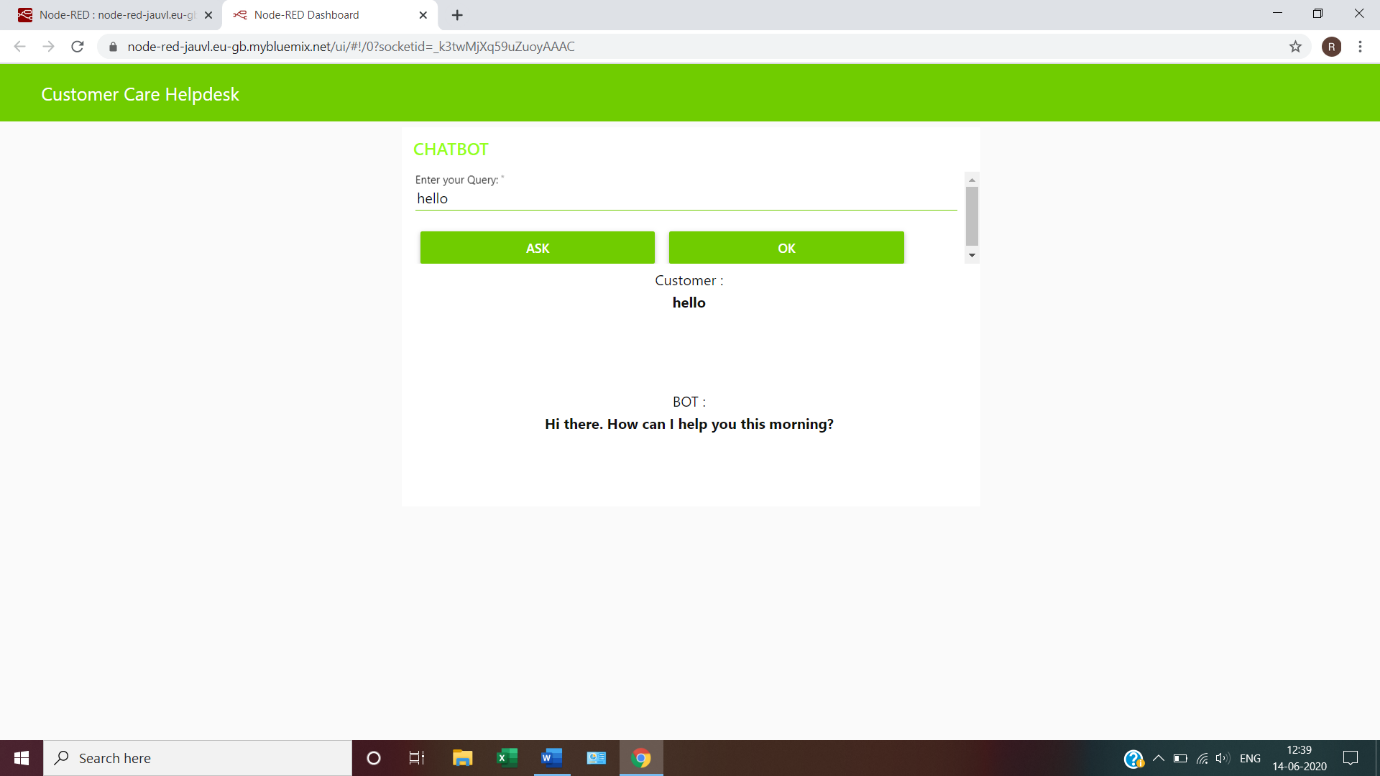
**6. Deploy and run Node-RED app**

Deploy the Node-RED flow.

Then copy the link URL up to **.net/** and paste at a new tab by adding **ui** at the end of the URL,

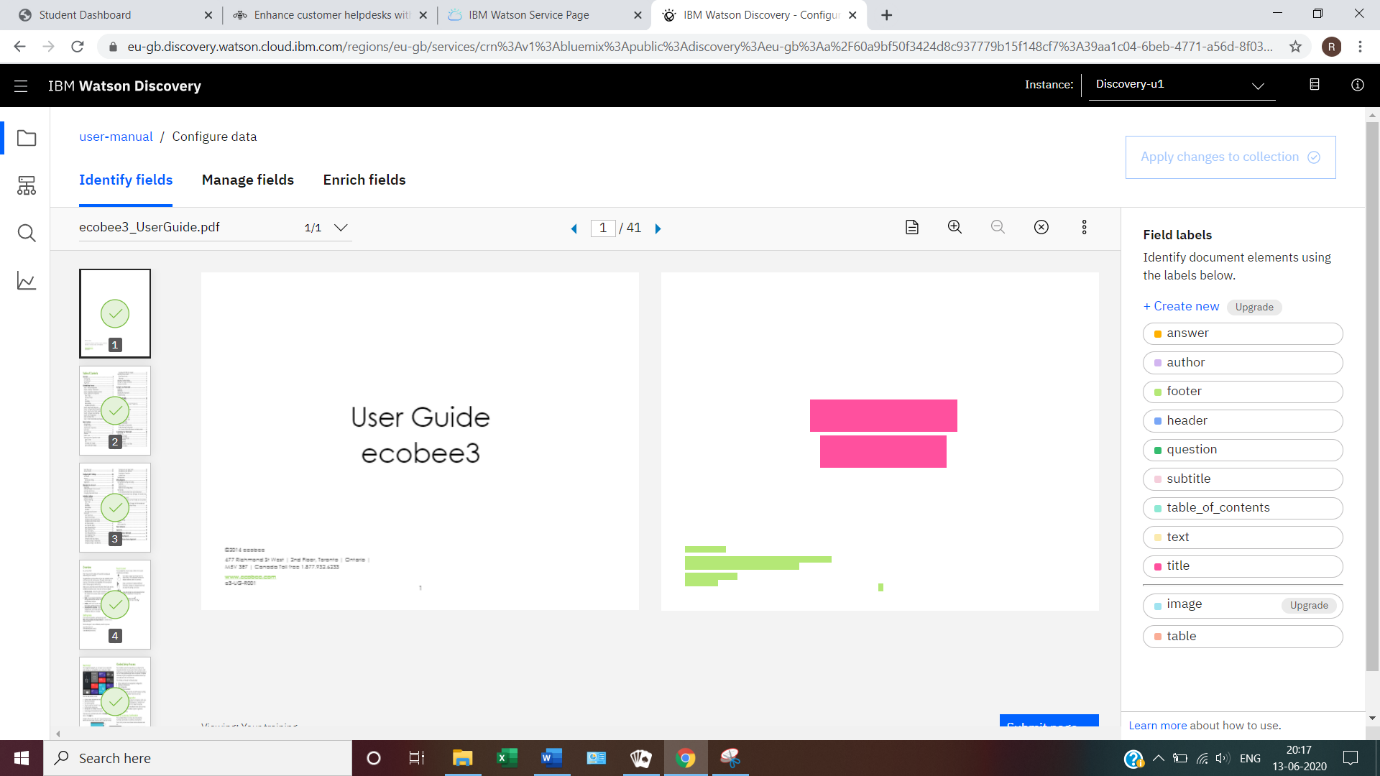
<https://node-red-jauvl.eu-gb.mybluemix.net/ui>

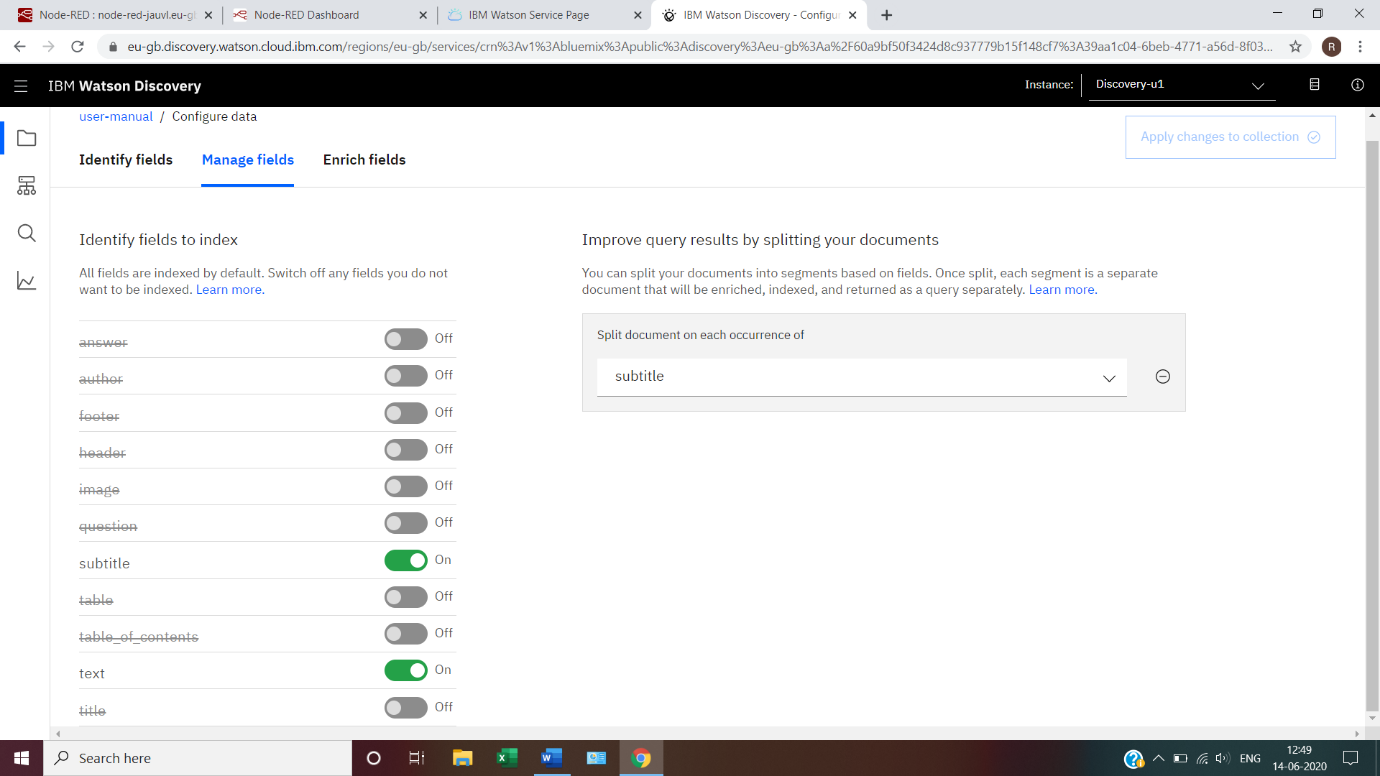
Or within the Node-RED window on the right side ribbon on dashboard tab, click the icon with outward arrow.

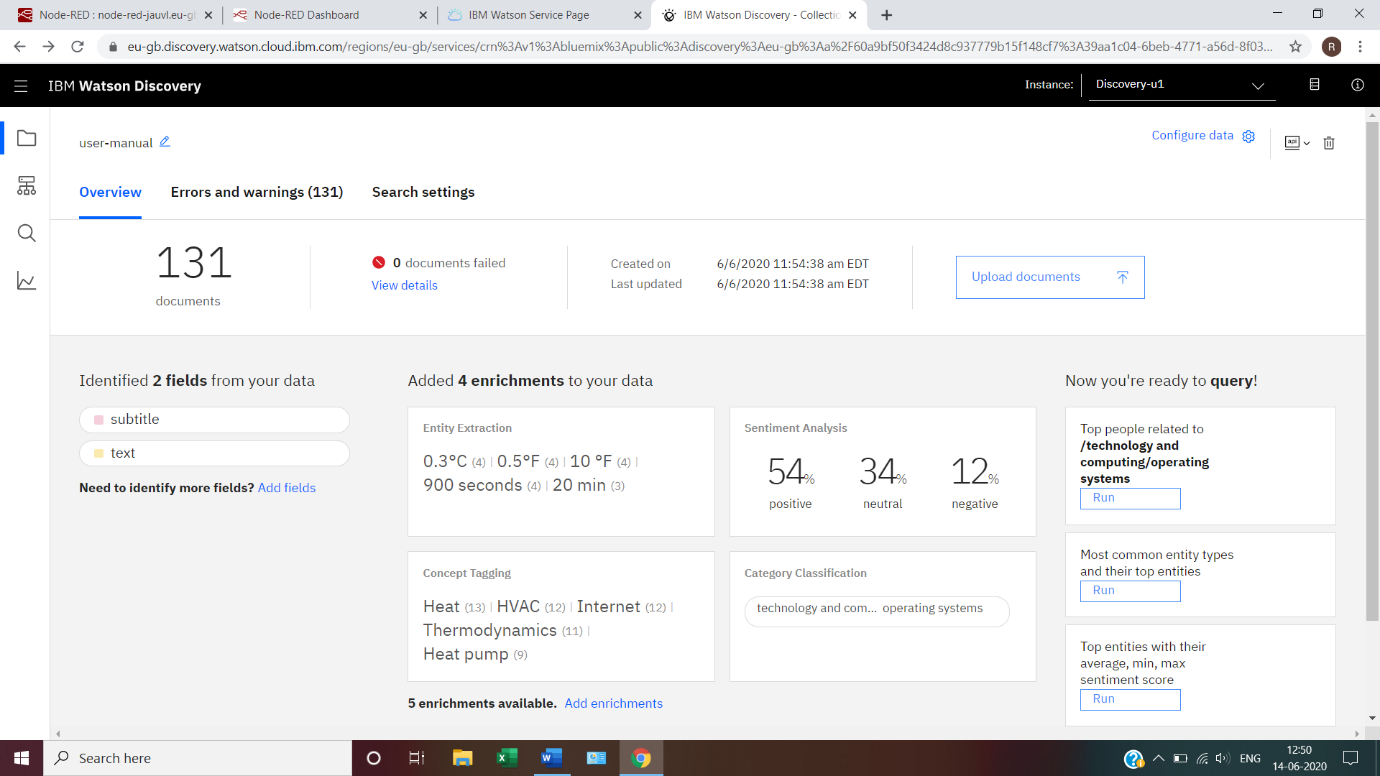


**4. EXPERIMENTAL INVESTIGATIONS**

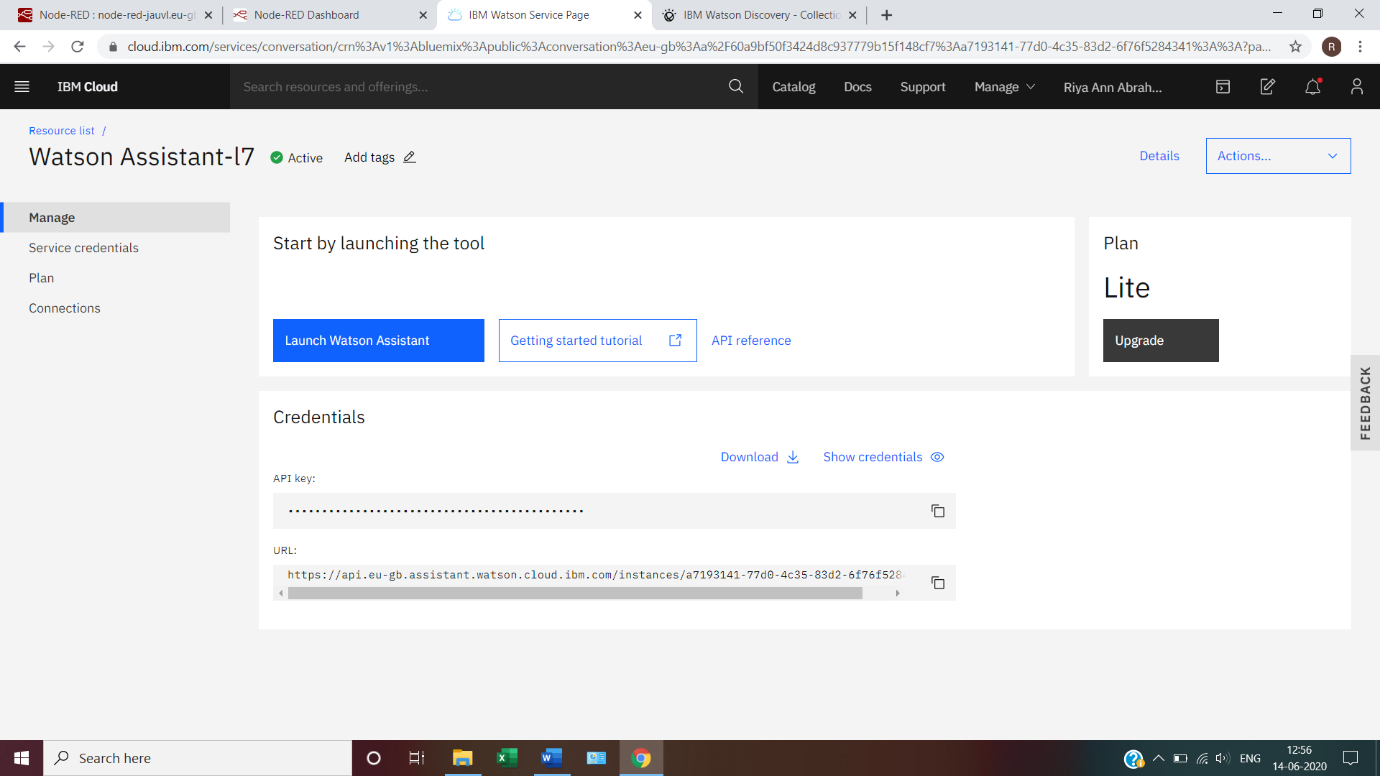
**WATSON DISCOVERY**

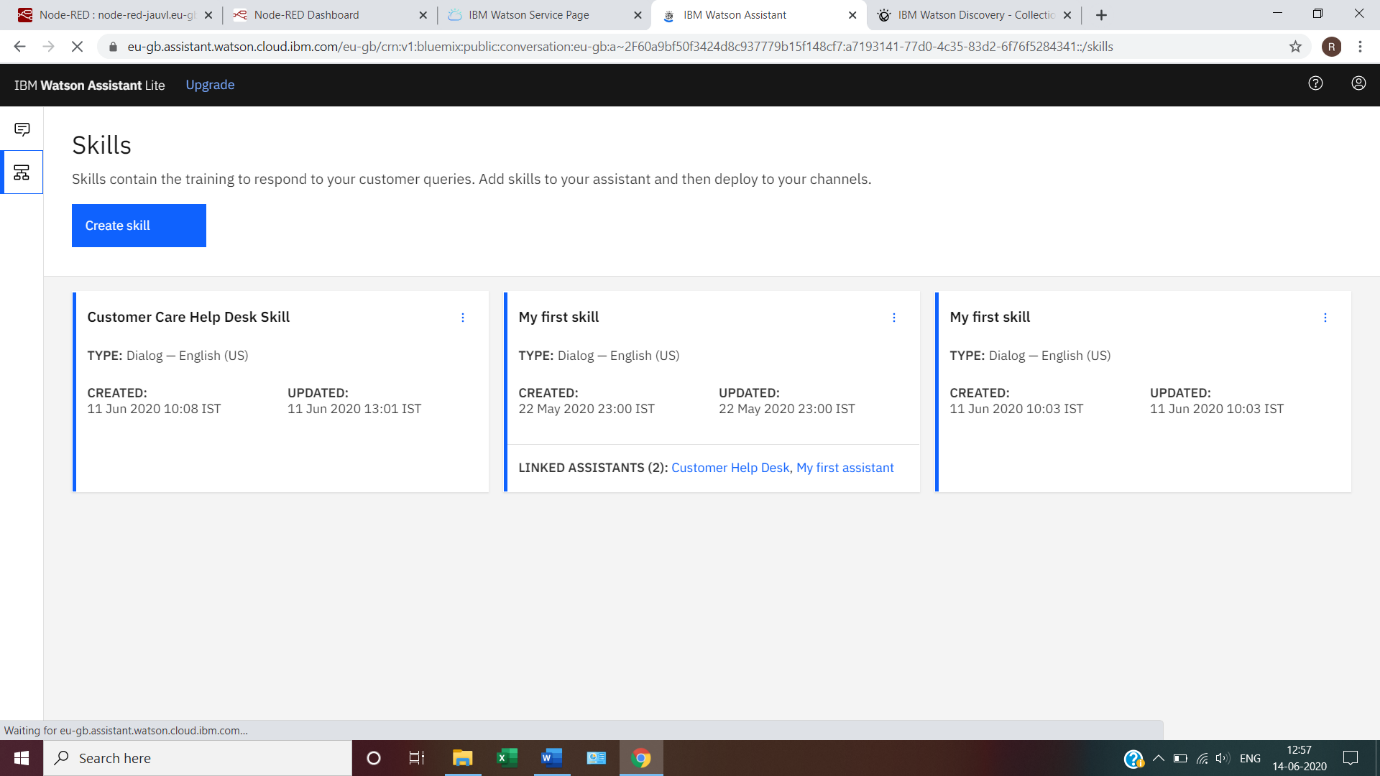
****

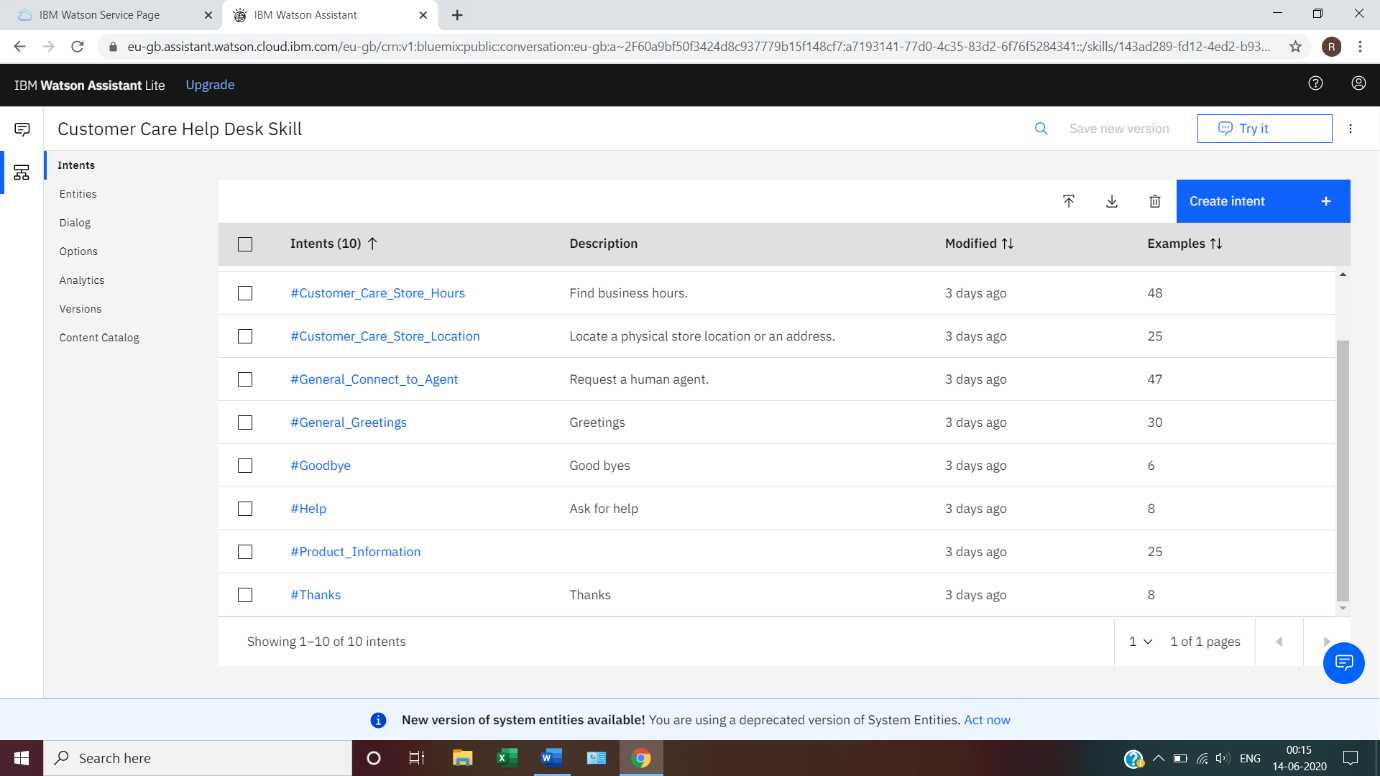
****

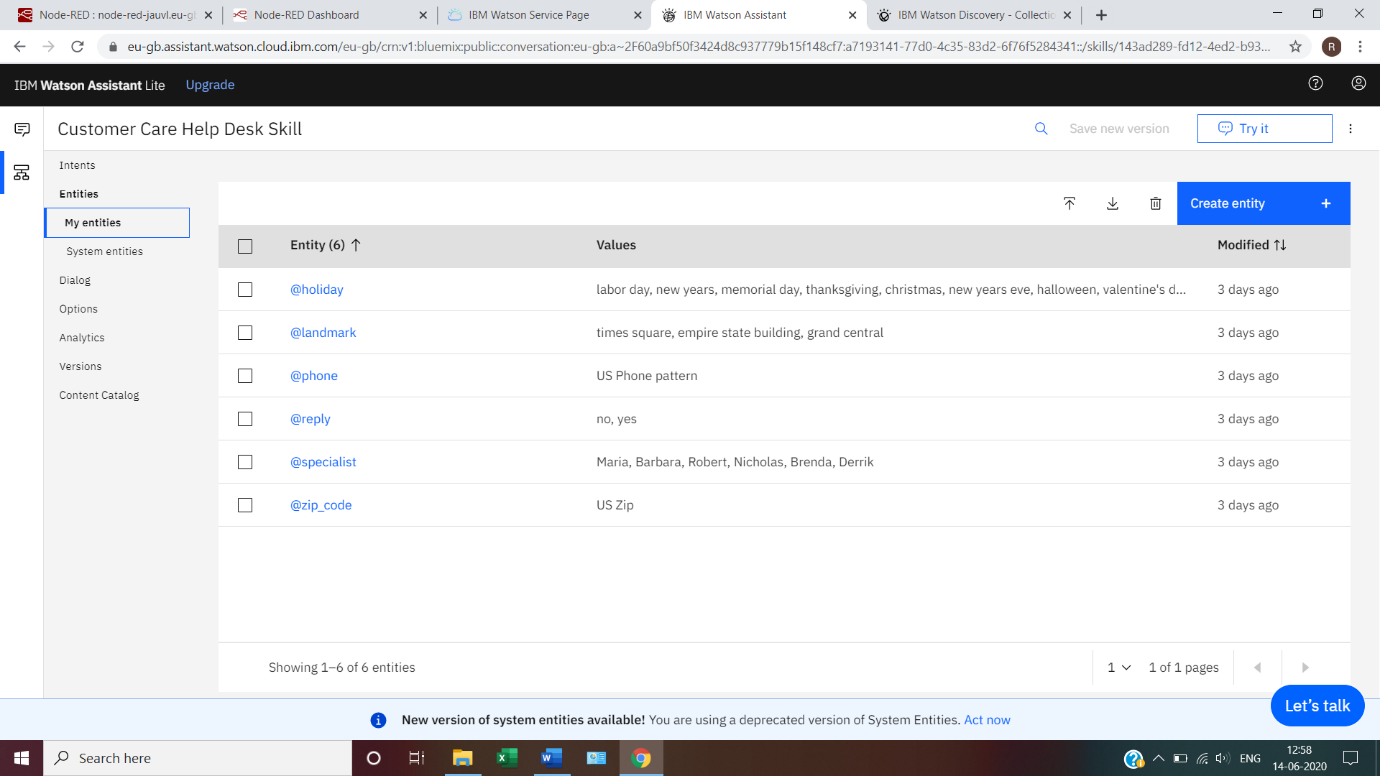
****

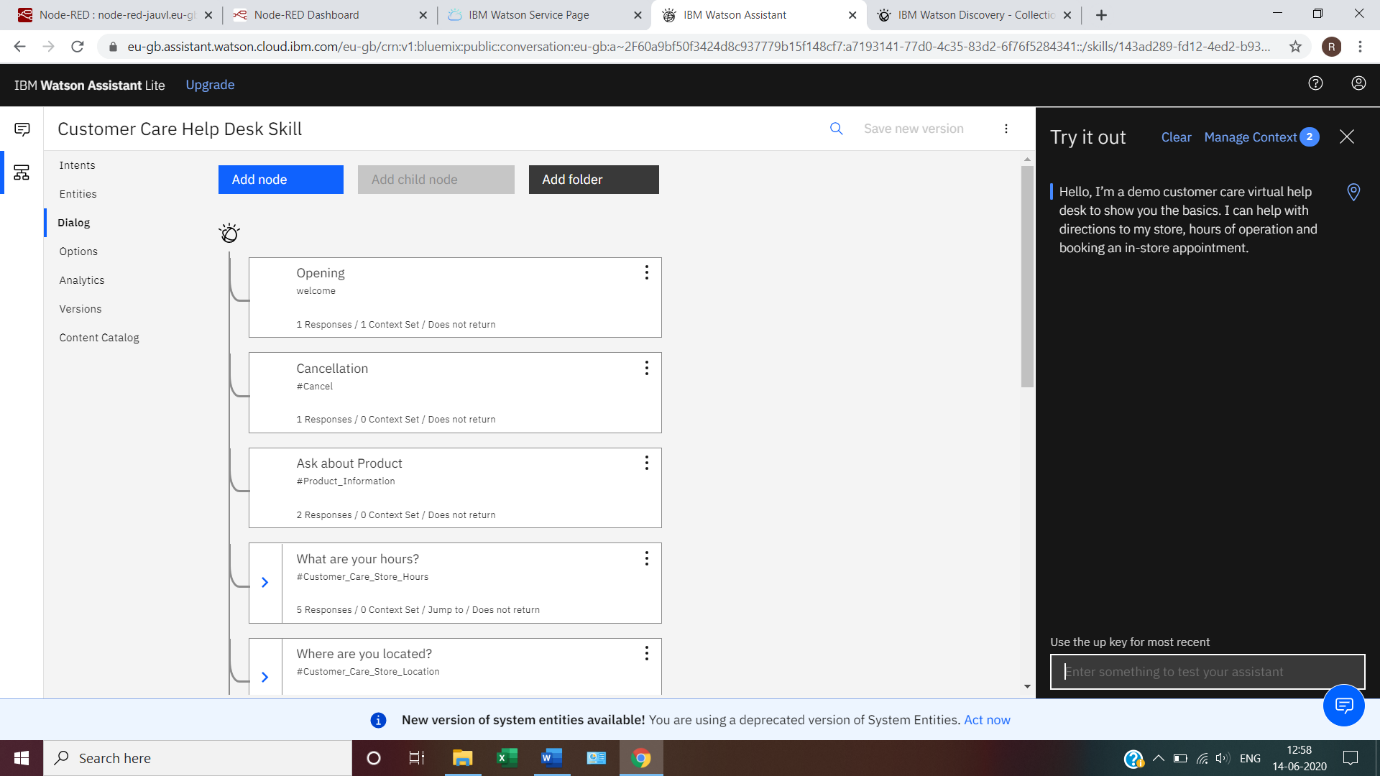
**WATSON ASSISTANT**

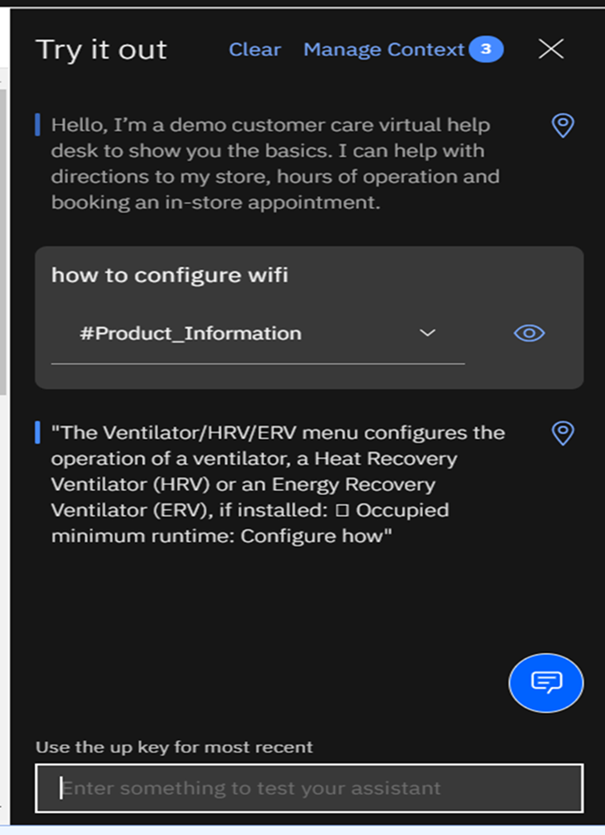
****

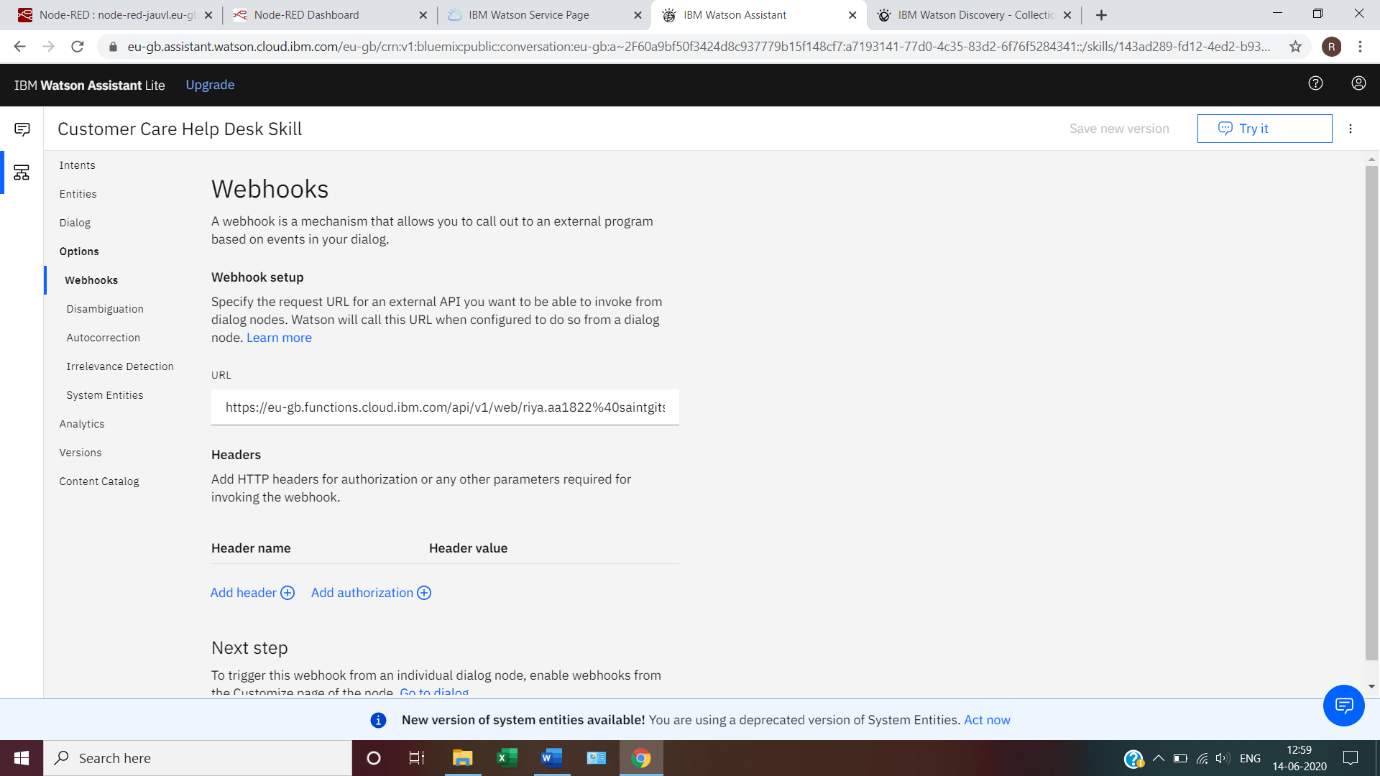


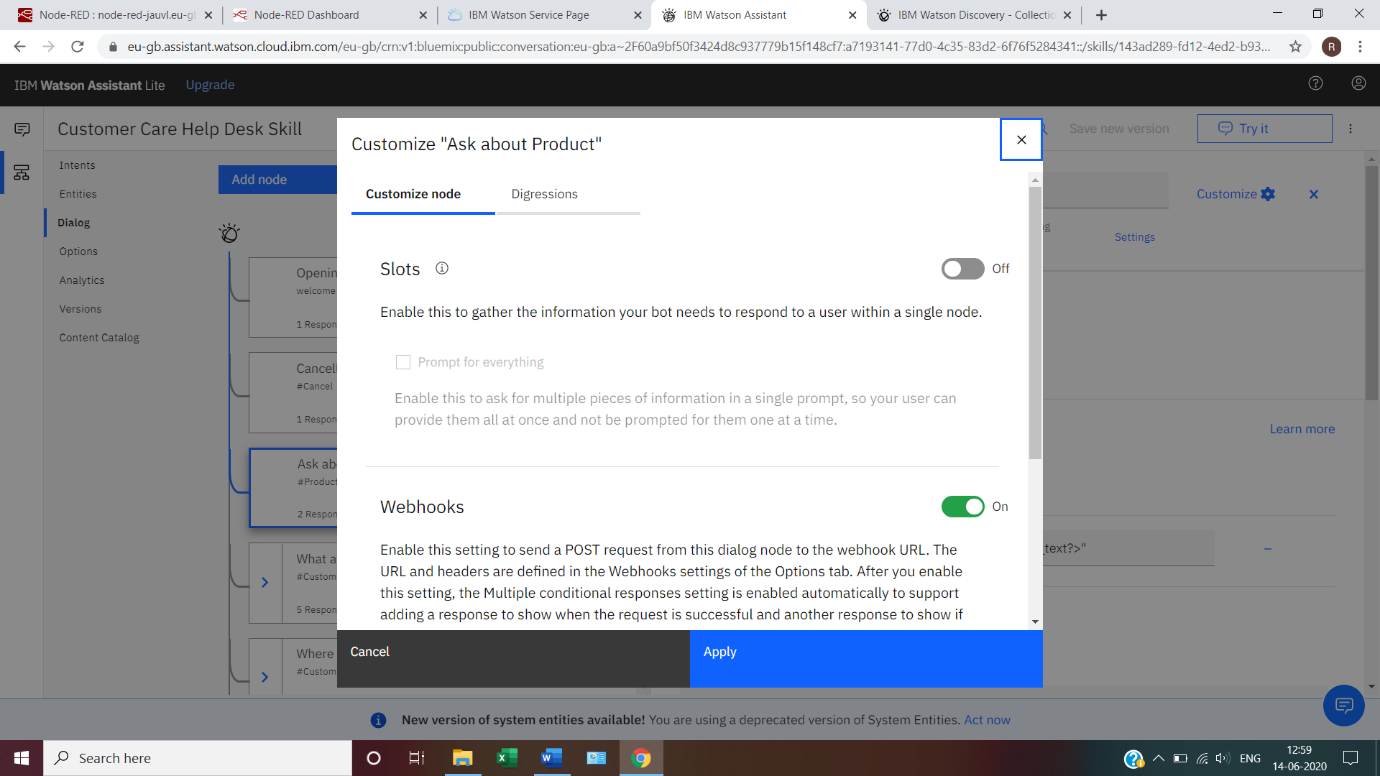




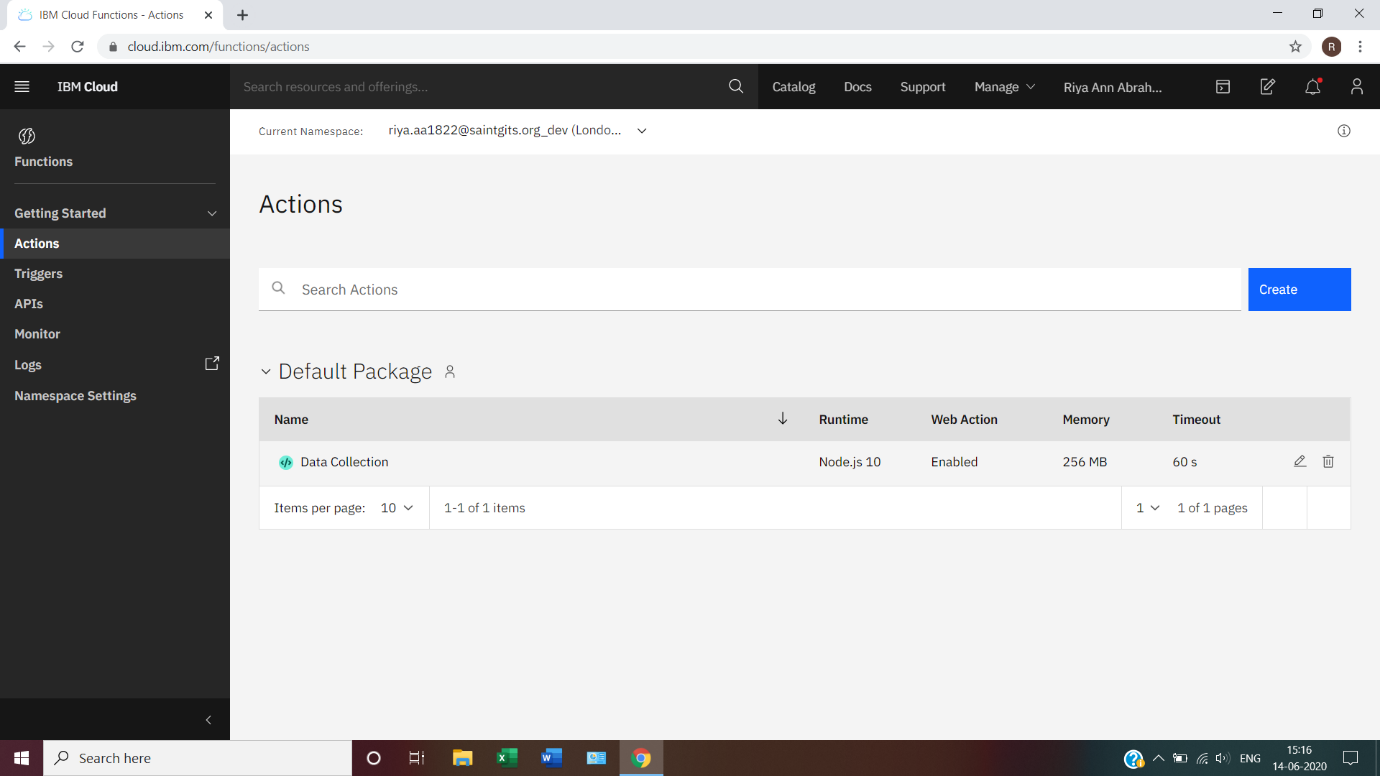


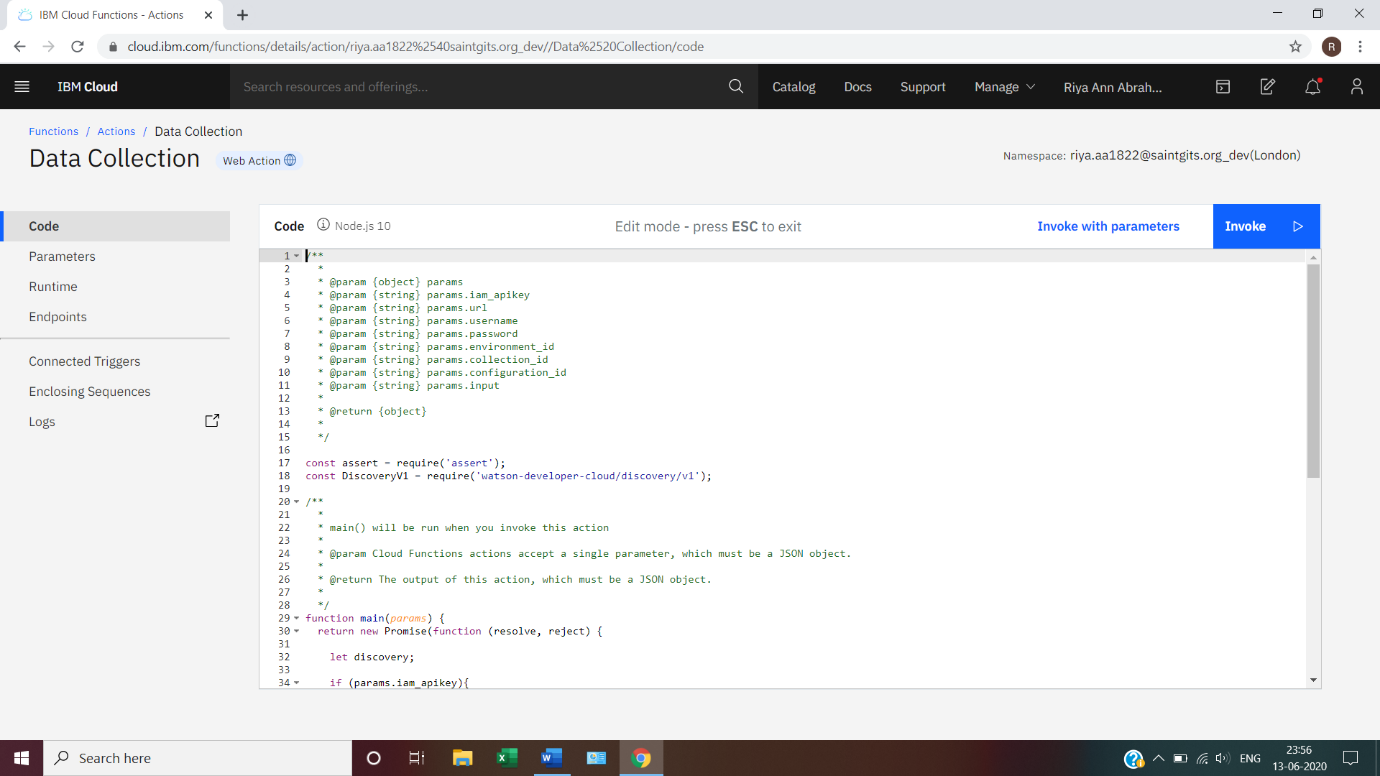


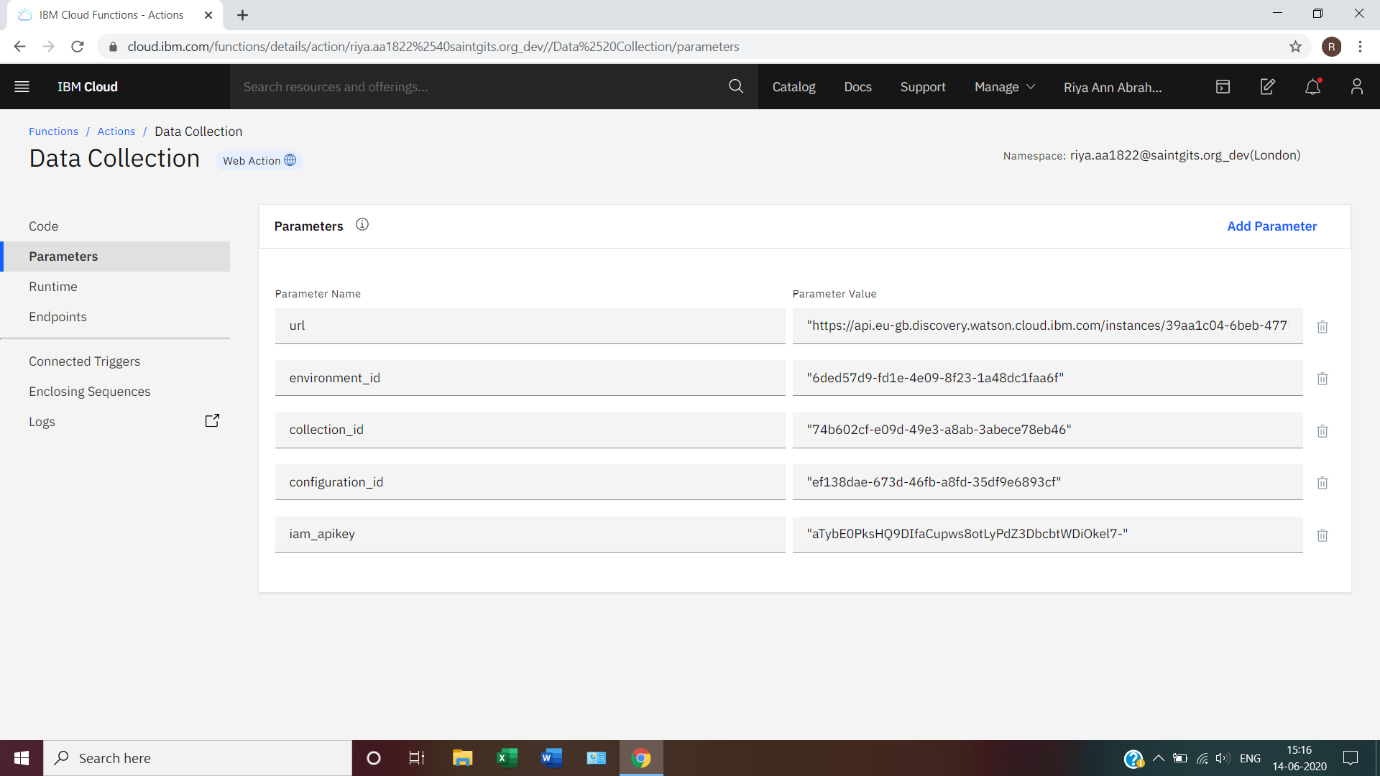




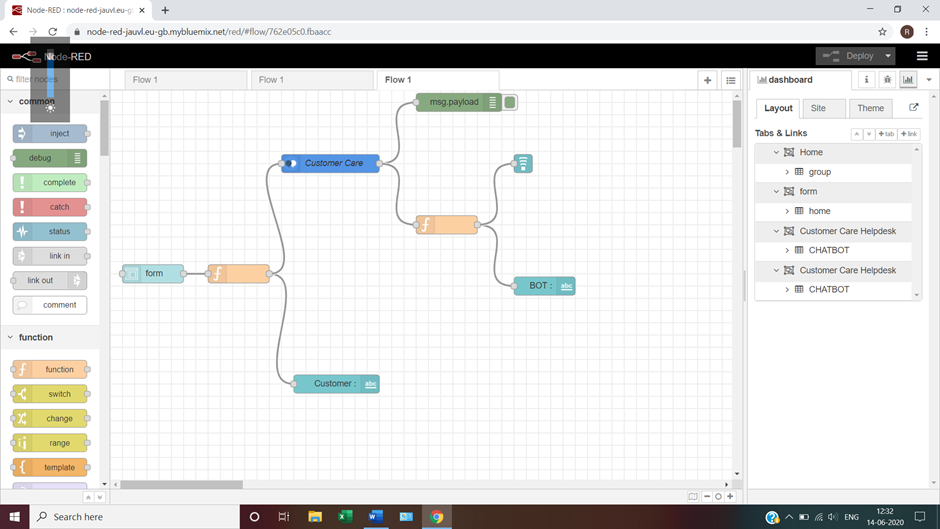
**CLOUD FUCTIONS**



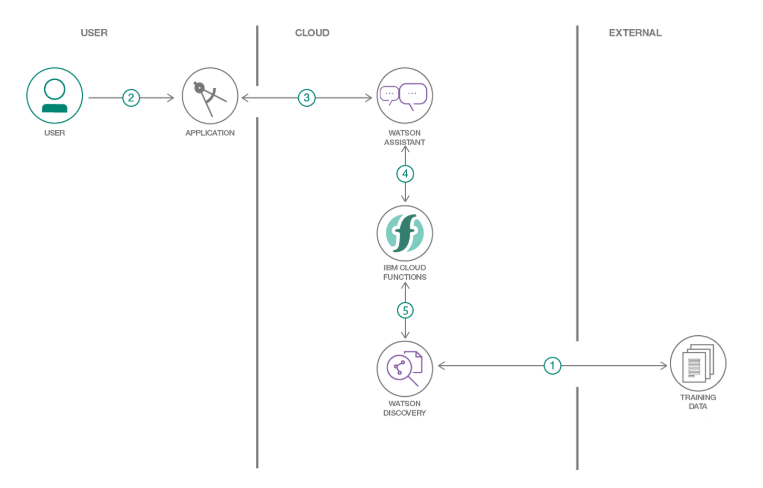
****

****

**NODE-RED FLOW**



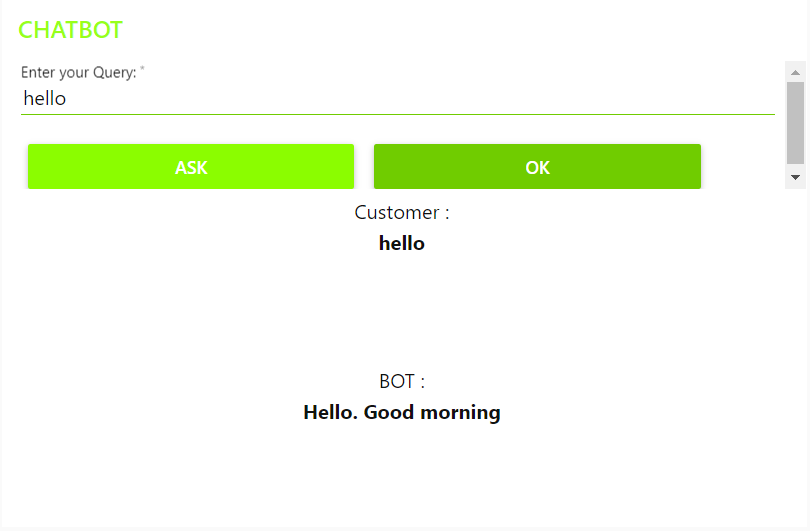
**5. FLOWCHART**

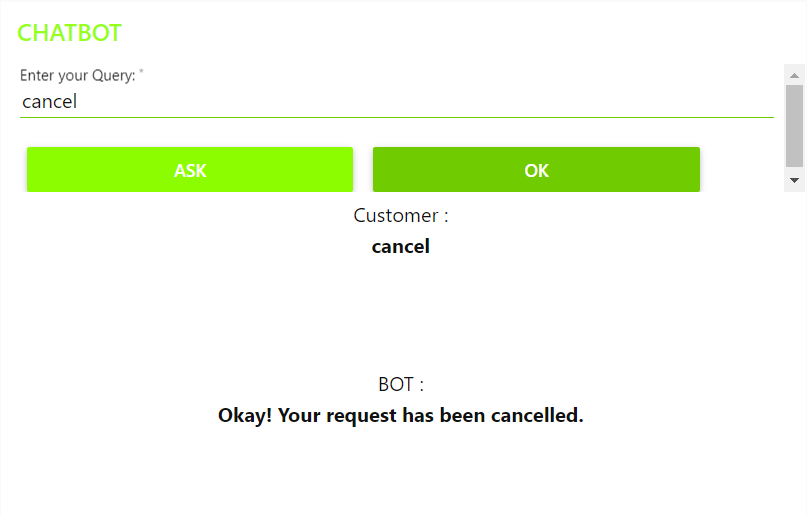


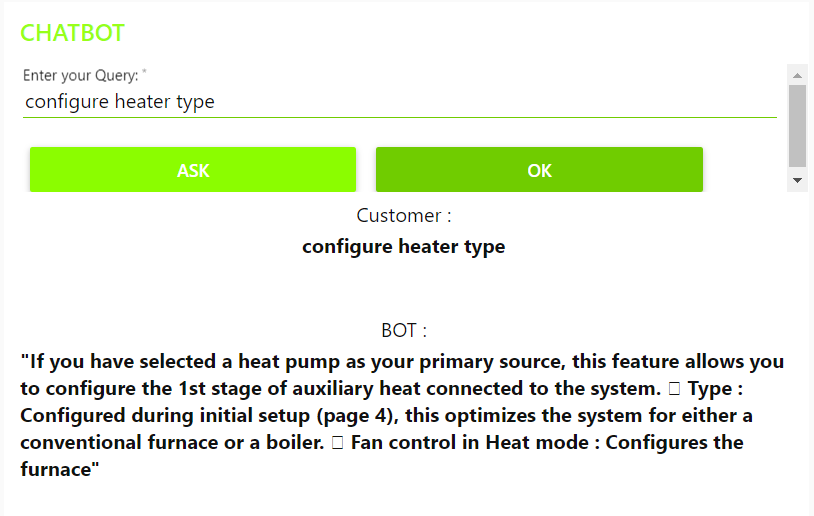
* The document is annotated using Watson Discovery SDU.
* The user interacts with the backend server via the app UI. The frontend app UI is a chatbot that engages the user in a conversation.
* Dialog between the user and backend server is coordinated using a Watson Assistant dialog skill.
* If the user asks a product operation question, a search query is passed to a predefined IBM Cloud Functions action.

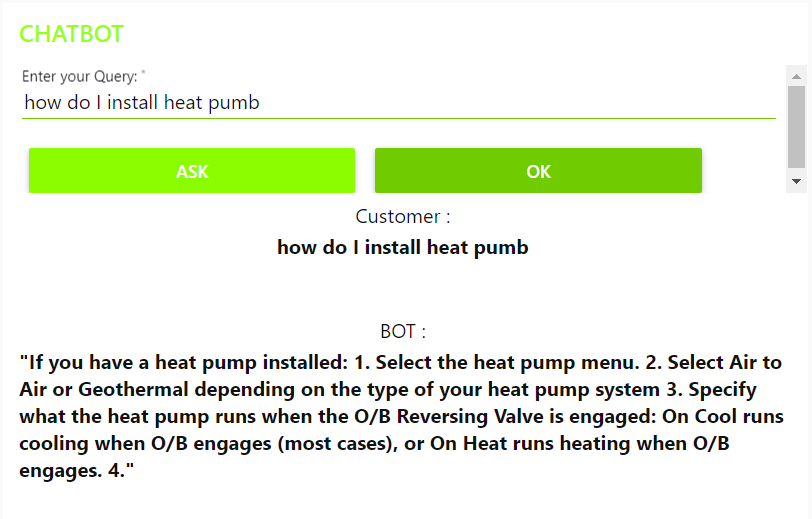
The Cloud Functions action will query the Watson Discovery Service and return the results.

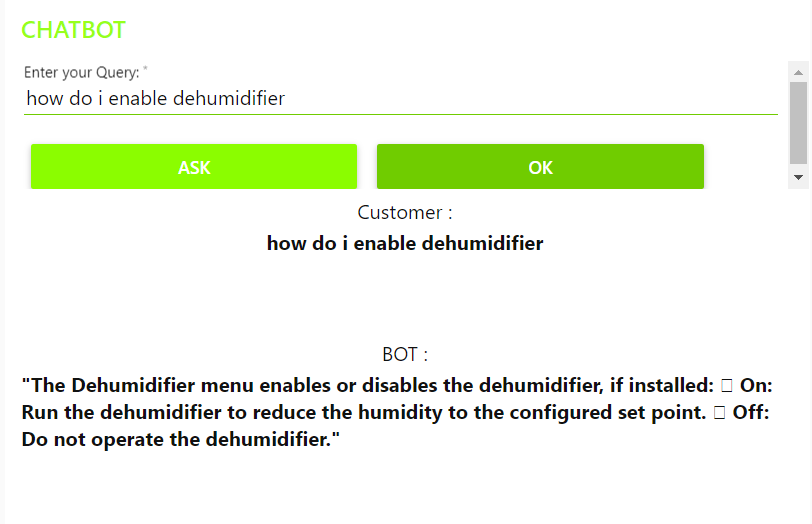
**6. RESULT**

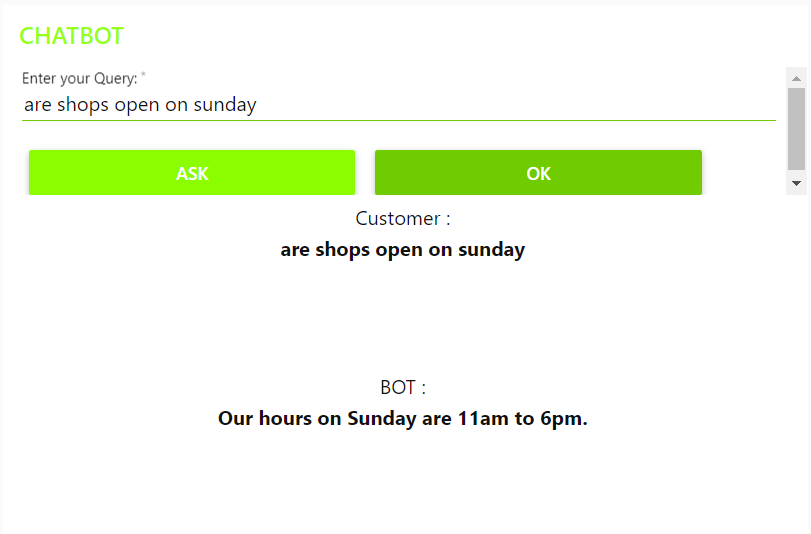
****

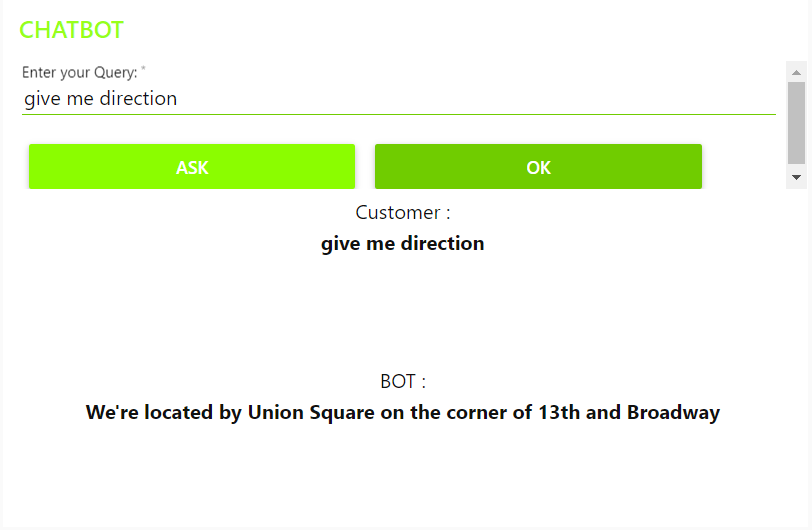
****

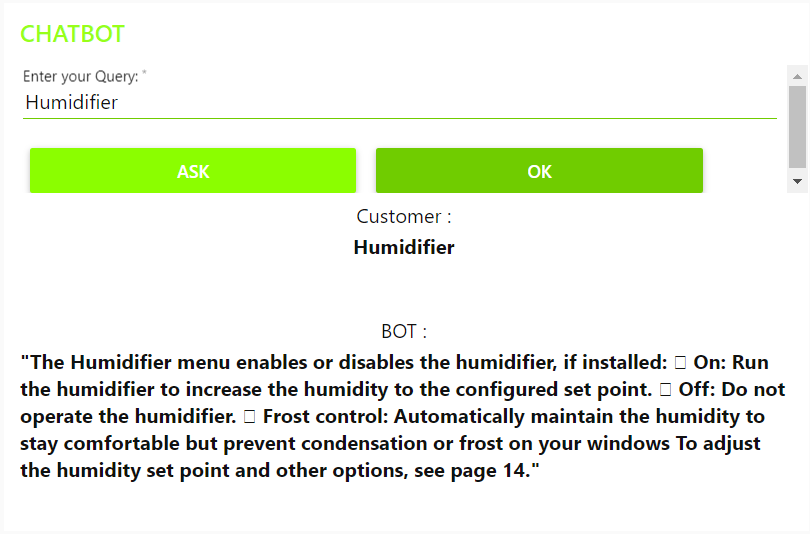
****

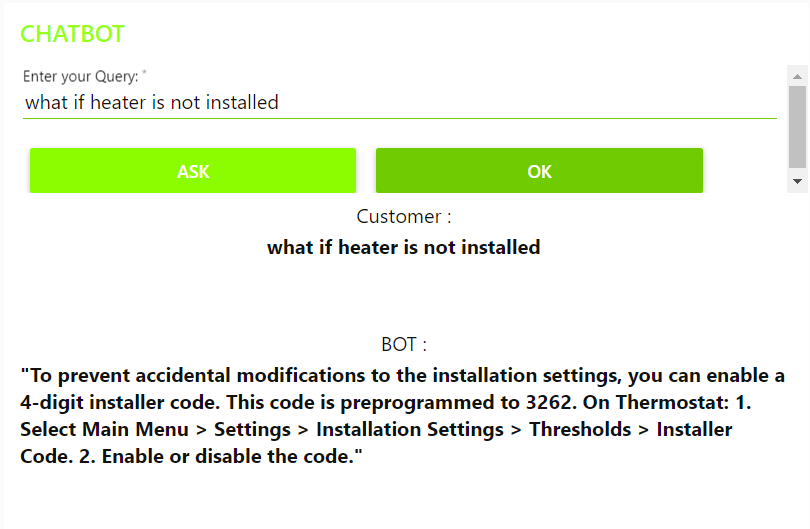
****

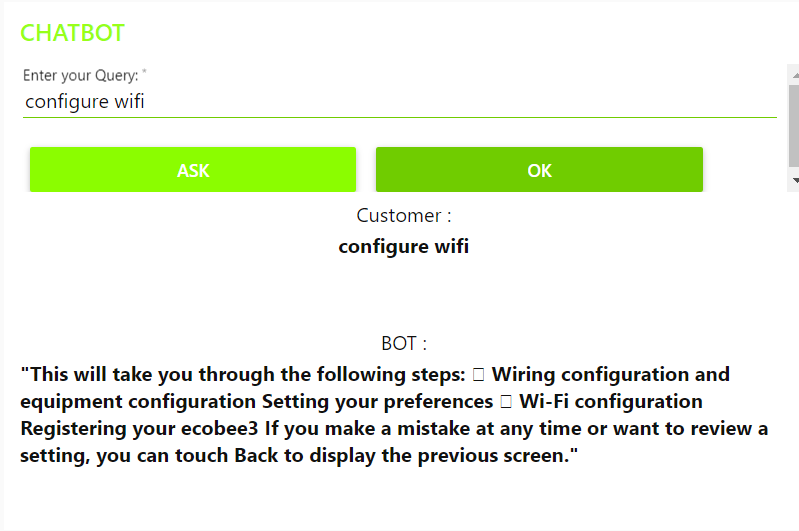
****

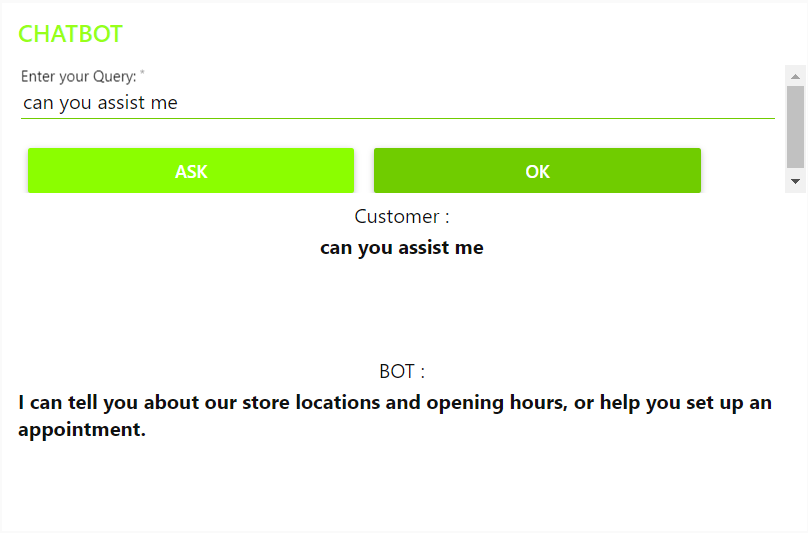
****

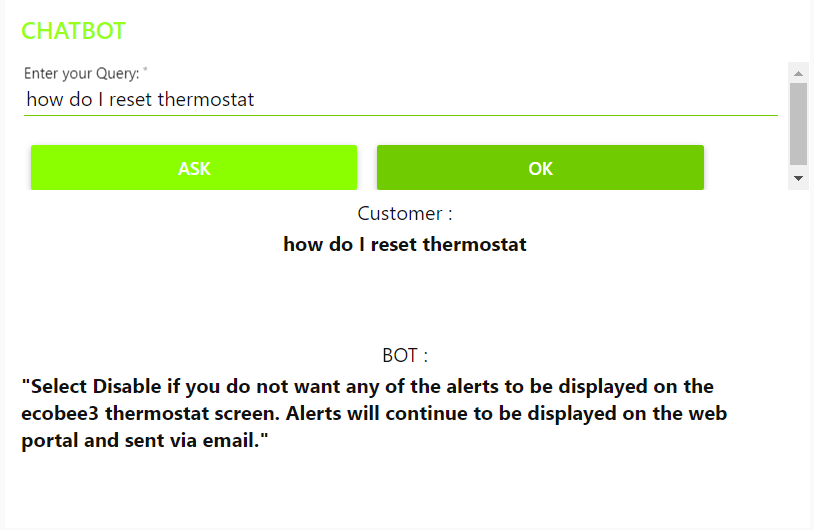
****

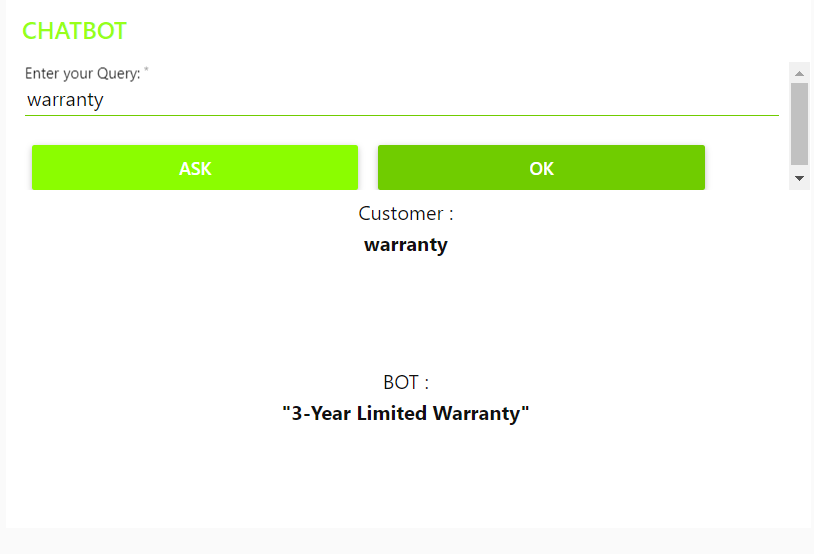
****

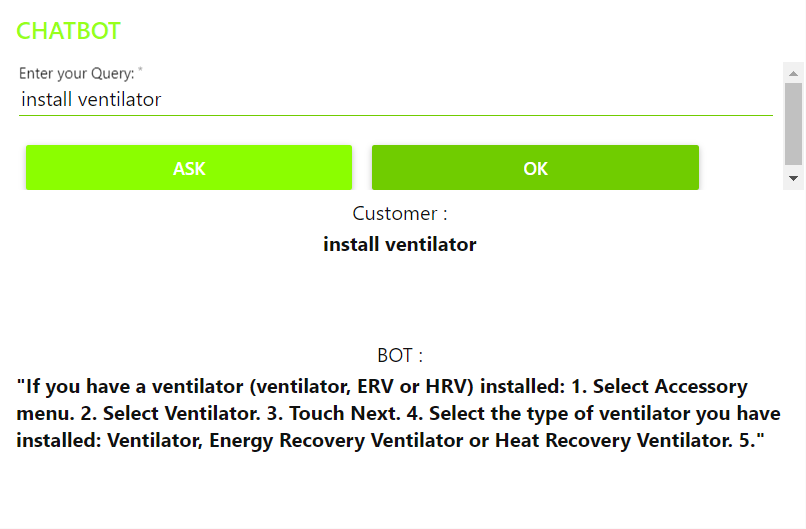
****

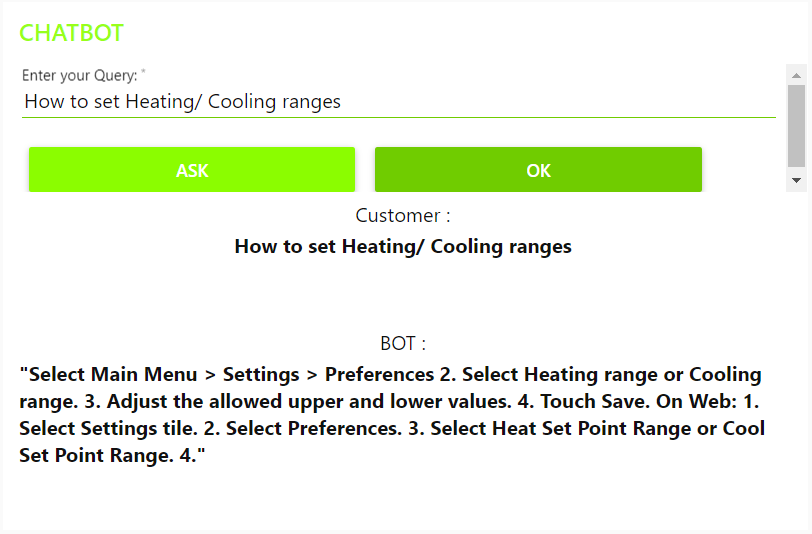
****

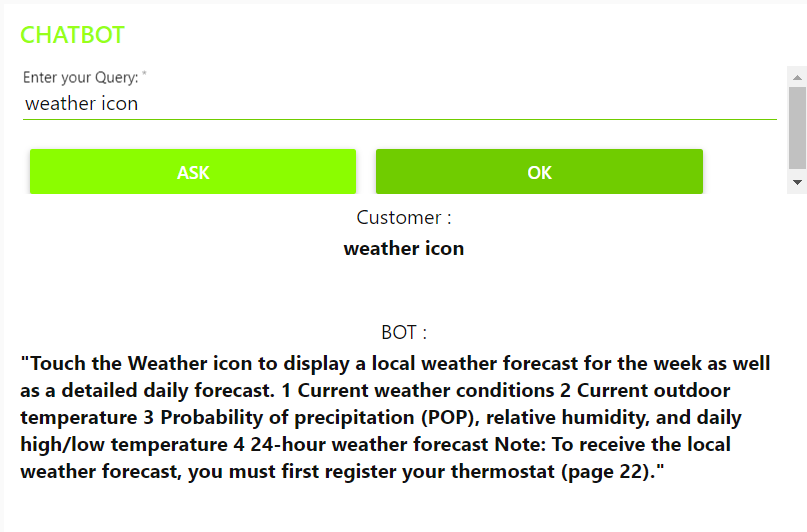
****

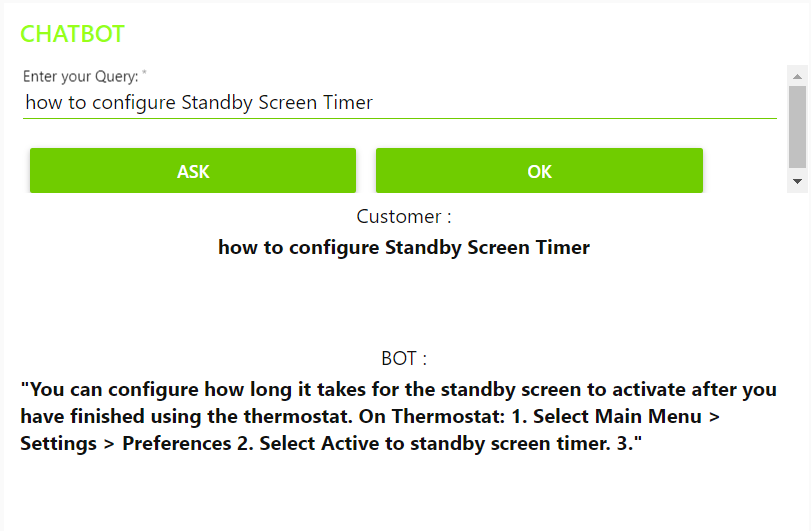


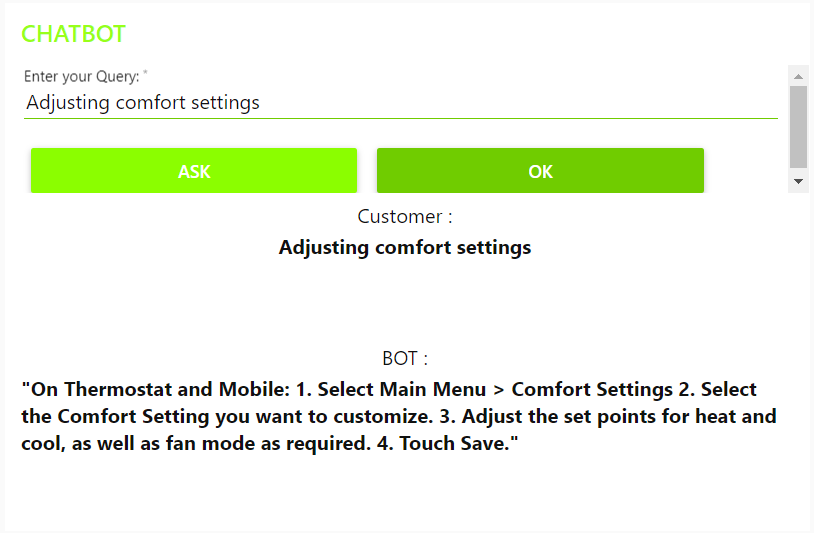


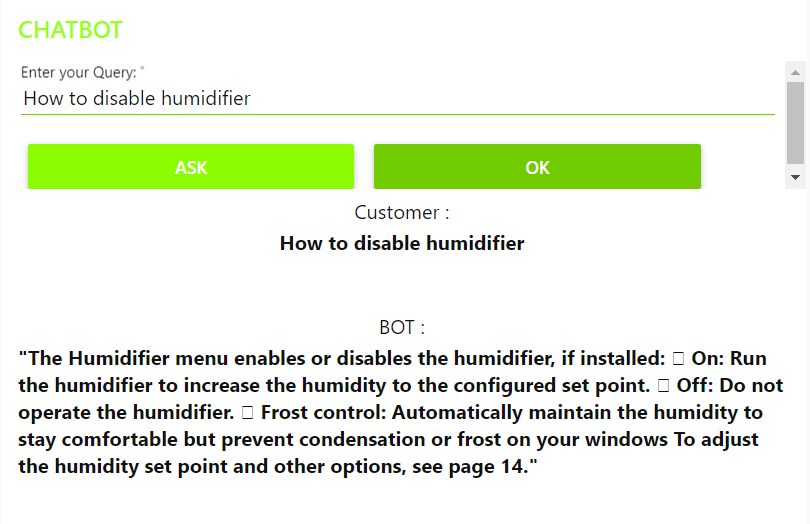


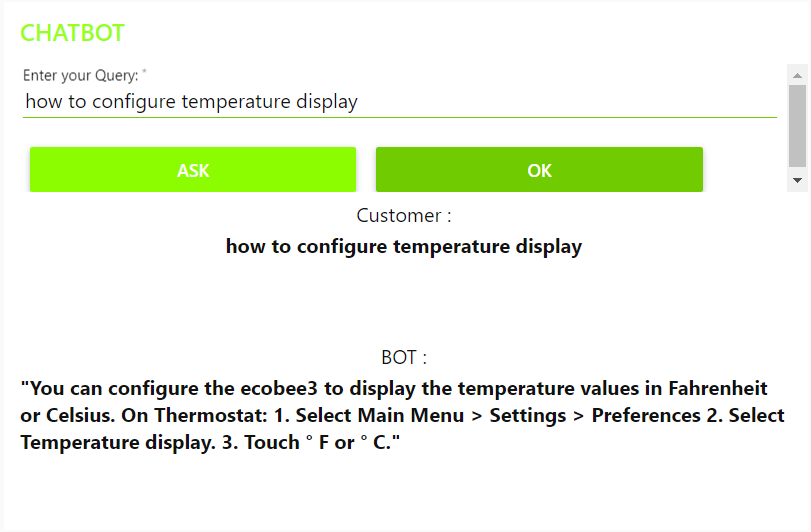


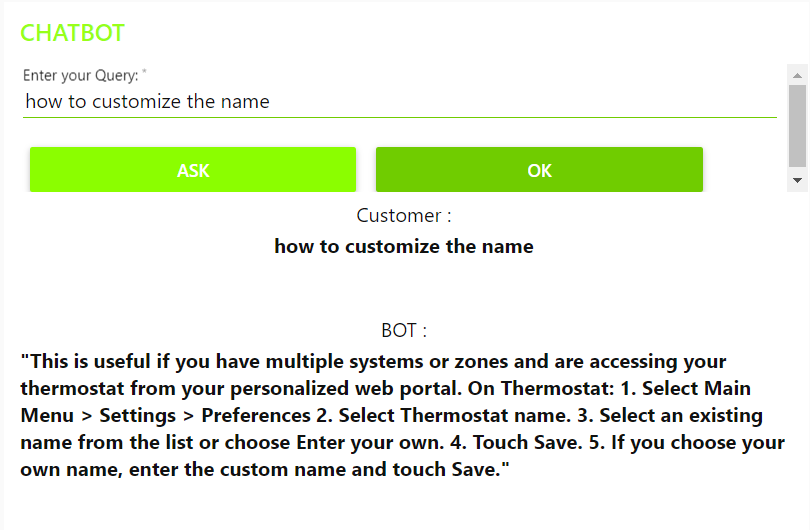












**7. ADVANTAGES & DISADVANTAGES**

**Advantages**

1. Cost efficient: With the help of chatbots we can reduce the human labor required in companies that receive multiple queries at once.
2. 24/7 Availability: Unlike humans, once we installed a chatbot, it can handle queries at any time of the day. This also allows companies to monitor customers <<traffic>> during nonworking hours and contact them later
3. Continuous Learning and Updating: AI- based chatbots are able to learn from interactions and update independently. This is one of the main advantages. When you hire a new employee, you have to train them continuously. However, chatbots <<from>> themselves.

**Disadvantages**

1. Sometimes it can mislead customers as it tries to search irrelevant information in the manual.
2. It may also give same answers to different queries.

**8. APPLICATIONS**

1. Companionship: 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. the chatbot asks questions, reacts to the answers, is able to speak on various topics, and share interesting news and facts from Google.
2. Deployment: This chatbot can be deployed to various websites as it can solve a lot of basic question. It can be used to deploy as Customer Help Desk for small scale products as their manual usually has the solution for the user’s problems.

**9. CONCLUSION**

An Intelligent Customer Help Desk Chatbot was created using various Watson services like Watson Discovery, Watson Assistant, Cloud Functions and Node-RED. This chatbot will be useful for the user to ask the assistant the queries related to the Product and will give them clear guidance about the product. If the assistant doesn’t resolve our query then we can ask it to redirect us to a representative.

**10. FUTURE SCOPE**

1. Voice-control: Voice recognition can be added with the virtual assistant. Then the customer can control application by using his voice. Soon, we could be joining meetings with a voice command, instead of dialing in the long meeting ID and password. 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.
2. Smarter Virtual Assistance: Much of what virtual assistance do now are basic skills, such as retrieving data and basic computation. As **Natural Language Processing** (NLP) continues and response capability, allowing for their use to become more widespread and complex. Also, as machine learning progresses, we may see virtual assistance becomes smarter and being to learn and predict customer needs.

**11. BIBLIOGRAPHY**

1. Node-RED Starter Application:

<https://developer.ibm.com/tutorials/how-to-create-a-%20%20node-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/5z3i5IsBVnk>

4.Watson-Discovery:

<https://www.youtube.com/embed/5z3i5IsBVnk>