Intelligent Customer Help Desk with Smart Document Understanding

A project report

by

SWEETY KAKADIYA

Category: Machine Learning

Application ID: SPS_APL_20200000616

Project ID: SPS_PRO_99

Internship at TheSmartBridge

May 2020

Sweety Kakadiya

sweetykakadiya18@gmail.com

https://github.com/Sweety2399/Intelligent-Customer-Help-Desk-with-Smart-Document-Understanding

https://www.youtube.com/watch?v=KAjizLcrxCk

Table of Contents

1. Introduction

1.1 Overview

A chat bot is a software application used to conduct an on-line chat conversation via text or text-to-speech, in lieu of providing direct contact with a live human agent. The typical customer care chat bot 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 to the helpdesk 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. It returns relevant sections of the owner's manual to help solve the customers' problems. The project uses 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 Purpose

The purpose of the project is to enhance the customer helpdesk so that it can answer operational questions which might not be answered by a typical chatbot or otherwise require a representative or a consultant to answer them.

2. Literature Survey

2.1 Existing System Study

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.

Thus, the existing chatbots can offer little or no assistance with respect to the operational aspect of any product. Advancements in Artificial Intelligence and mainly Natural Language Processing have made way for systems that not only understand human queries but also effectively answer them.

2.2 Proposed System

The proposed solution is a chatbot that is built using IBM Cloud Services. It is a web application that utilizes multiple IBM Watson services to create a better customer care experience. Using the Watson Discovery Smart Document Understanding (SDU) feature, we will enhance the Discovery model so that queries will be better focused to only search the most relevant information found in a typical owner's manual. Using Watson Assistant, we will use a standard customer care dialog to handle a typical conversation between a customer and a company representative. When a customer question involves operation of a product, the Assistant dialog will communicate with the Discovery service using a webhook. The webhook will be created by defining a web action using IBM Cloud Functions.

Smart Document Understanding (SDU)

SDU trains Watson Discovery to extract custom fields in your documents. Customizing how your documents are indexed into Discovery will improve

the answers returned from queries. With SDU, you annotate fields within your documents to train custom conversion models. As you annotate, Watson is learning and will start predicting annotations. SDU models can also be exported and used on other collections. Current document type support for SDU is based on your plan.

Lite plans: PDF, Word, PowerPoint, Excel, JSON, HTML.

Advanced plans: PDF, Word, PowerPoint, Excel, PNG, TIFF, JPG, JSON, HTML.

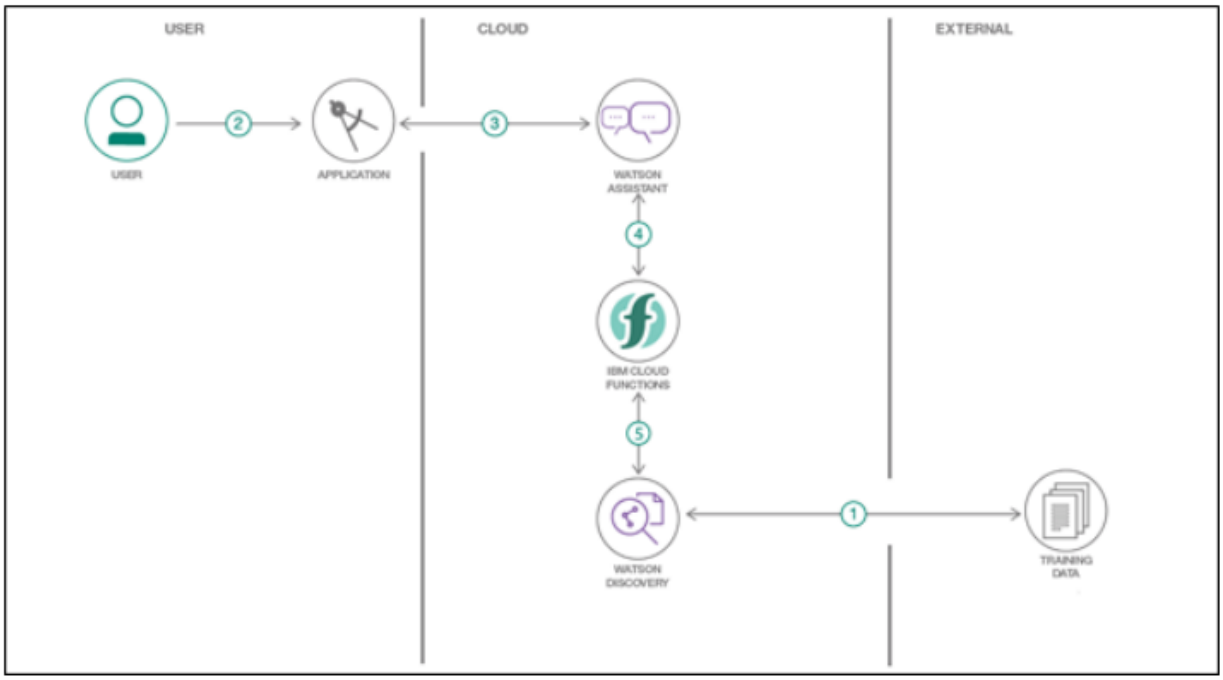
Webhook

A webhook is a mechanism that allows you to call out to an external program based on something happening in your program. When used in a Watson Assistant dialog skill, a webhook is triggered when the Assistant processes a node that has a webhook enabled. The webhook collects data that you specify or that you collect from the user during the conversation and save in context variables, and sends the data to the Webhook request URL as an HTTP POST request. The URL that receives the webhook is the listener. It performs a predefined action using the information that is provided by the webhook as specified in the webhook definition, and can optionally return a response.

In the proposed system, the webhook will communicate with an IBM Cloud Functions web action, which is connected to the Watson Discovery service.

3. Theoretical Analysis

3.1 Block Diagram



- 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.

3.2 Hardware/Software Designing

- Create an IBM Cloud account.

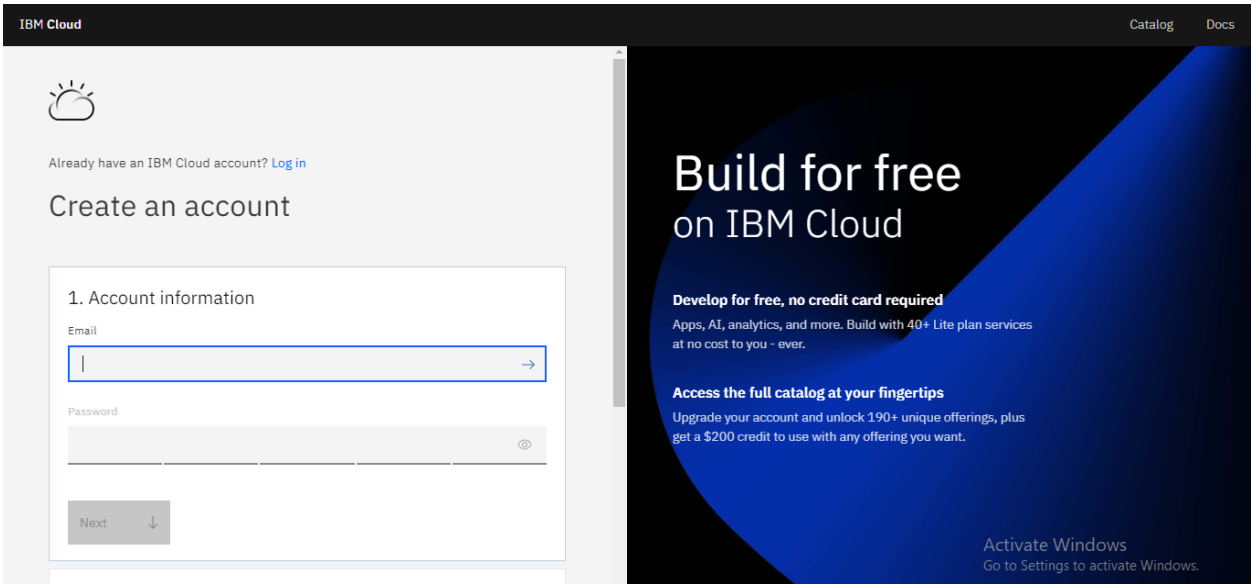
- Configure Watson Assistant.

- Configure Watson Discovery.
- Create Cloud Function.
- Create Node-RED application to integrate all services.
- Deploy and run the application.

4. Experimental Investigations

A. Create an IBM Cloud account.

- Create an account by signing up on cloud.ibm.com or use an existing account.
- For this project, I created an IBM Cloud Lite account.

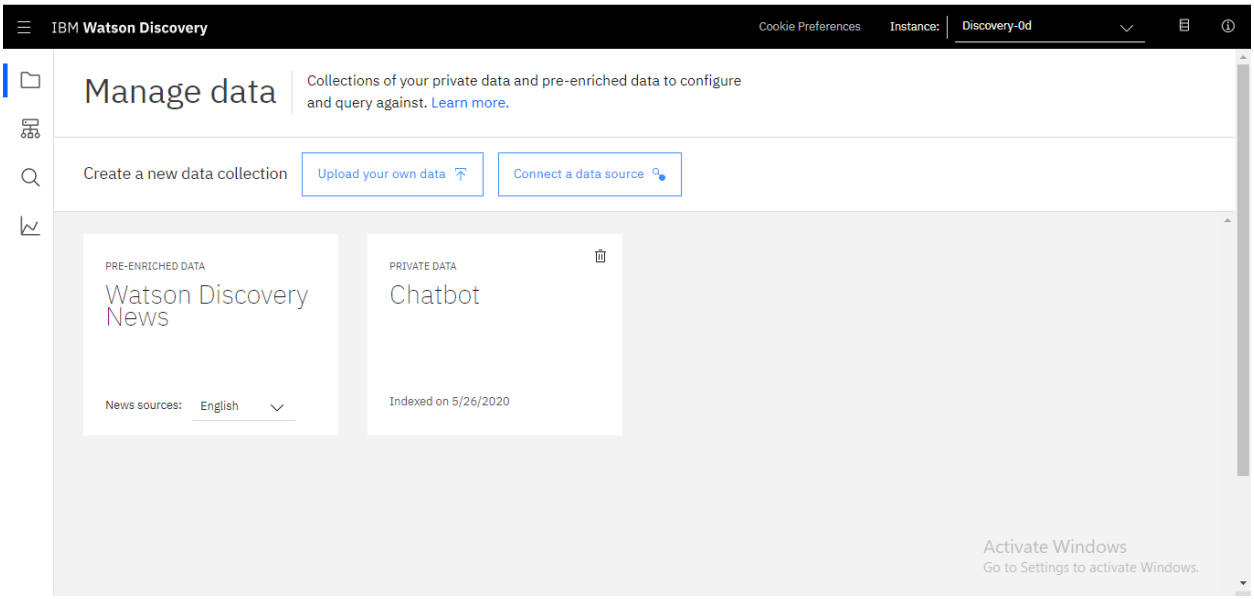


B. Create the necessary services.

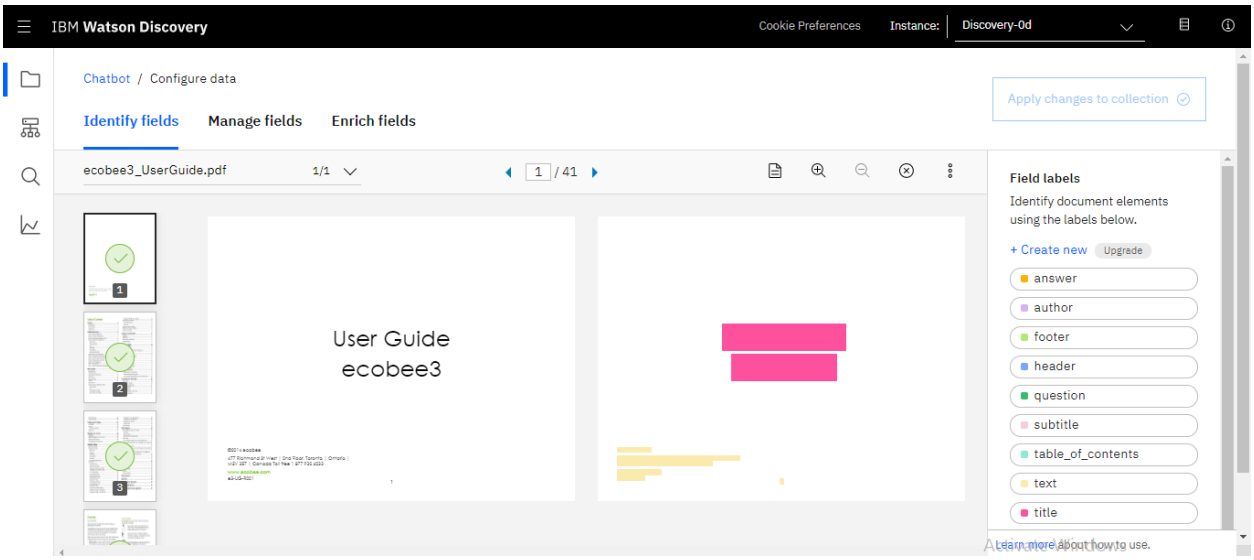
- Create a Watson Assistant instance
- Create a Watson Discovery instance
- Create a Node-RED starter application

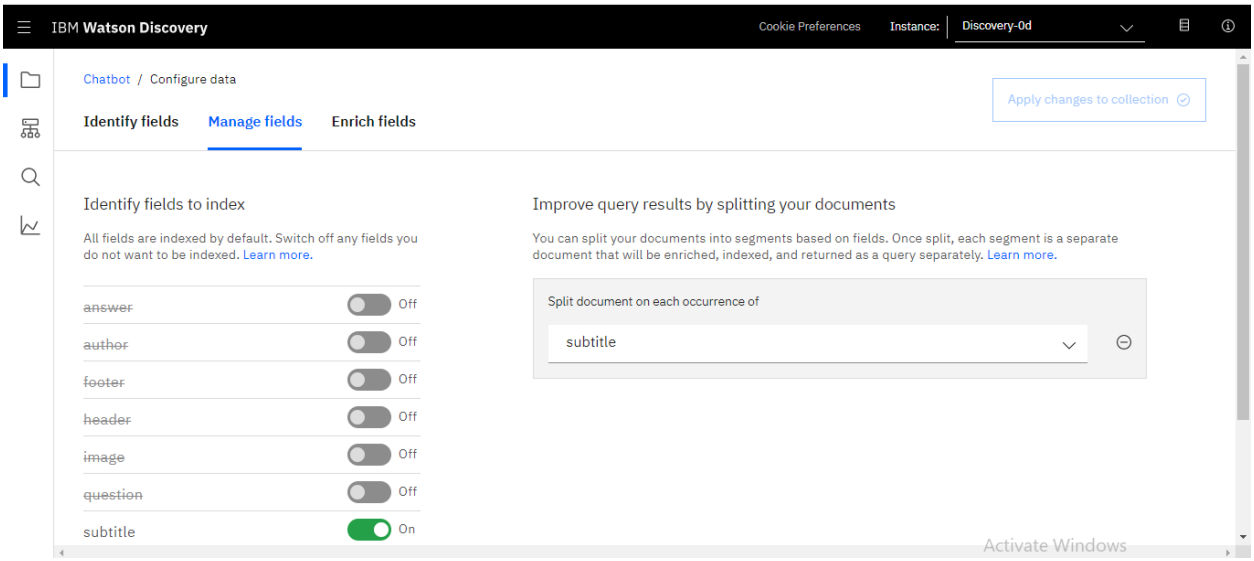
C. Configure Watson Discovery.

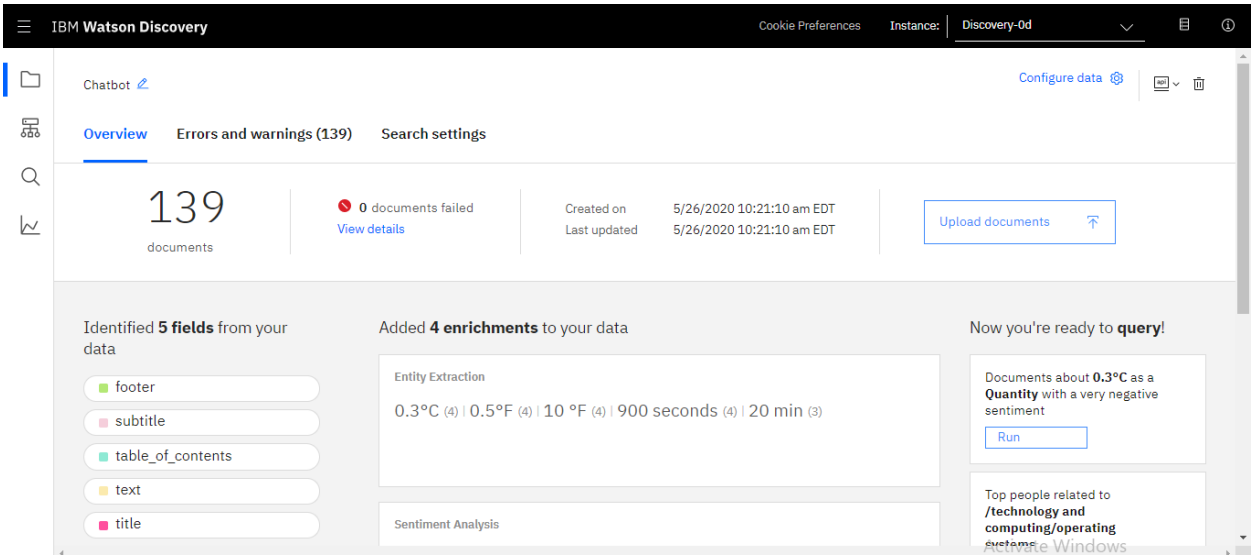
- Launch the Watson Discovery tool and create a new data collection by selecting the upload your own data option.



- Upload the Ecobee thermostat manual.
- Apply Smart Document Understanding to the document.
    - 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.
    - As you go through the annotations one page at a time, Discovery is learning and should start automatically updating the upcoming pages.
    -







D. Create IBM Cloud Functions action.

- Start the IBM Cloud Functions service by selecting Create Resource from the IBM Cloud dashboard.
- 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.

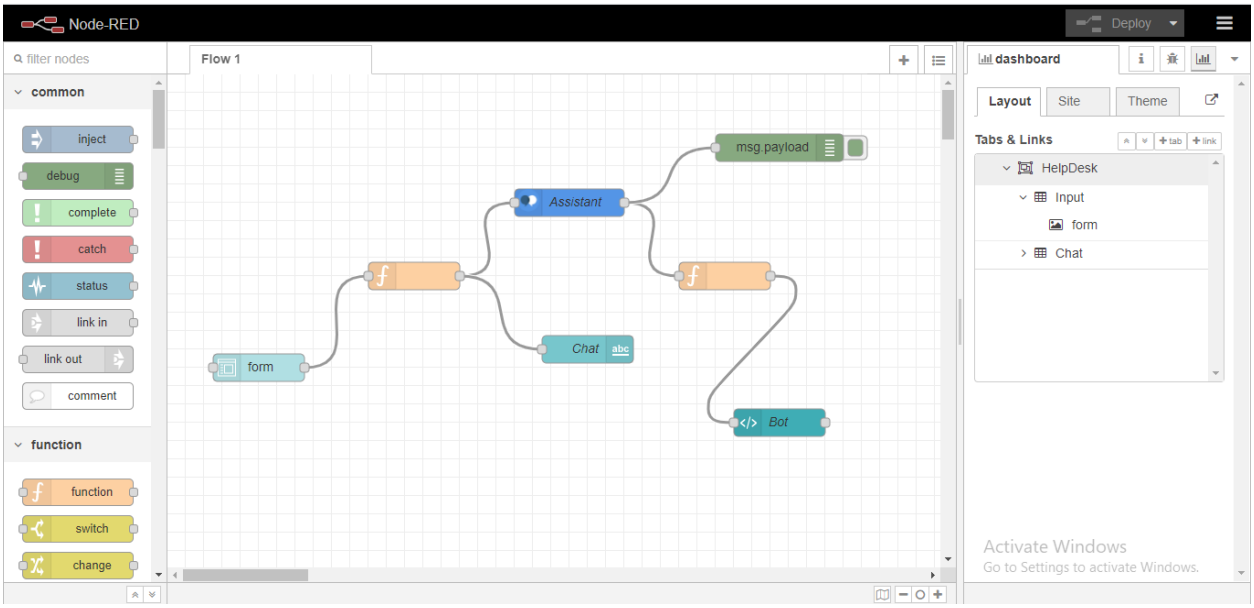
- Type the appropriate code.

E. Configure Watson Assistant.

- Launch the Watson Assistant Tool and create a new dialog skill. Select the use sample skill option as the starting point.
- 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.
- Click the Create intent button. Name the intent #Product_Information, and enter the example questions to be associated with it.
- Create a new Dialog node, name it “ask about product” and assign it our new intent.
- Now select the Options tab and enable webhook for the Cloud Function created above.

F. Build a Node-RED Flow to integrate all services.

- The form node takes an input query from the user and parses it to the assistant node using the function node.
- The assistant (Watson Assistant instance created above) processes the query along with Watson Discovery and returns the result.
- The result is parsed using the function node and displayed using a template node. A debug node is also used.



G. Configure the nodes and build a web dashboard in Node-RED.

- Configure the nodes by writing the appropriate code.
- Add a text field to get user input using the form node.
- Code for input_fn

```
msg.payload="You: "+msg.payload.q;
return msg;
```

- Code for output_fn

```
msg.payload.text="";
if(msg.payload.context.webhook_result_1){
        for(var i in msg.payload.context.webhook_result_1.results){
                msg.payload.text=msg.payload.text+"\n"+msg.payloa
        d.context.webhook_result_1.results[i].text;
        }
        msg.payload="Bot: "+msg.payload.text;
}
else
        msg.payload="Bot: "+msg.payload.output.text[0];
return msg;
```

- Code for bot_response - to display bot's response on the UI

```
<div style="height:440px">
        {{msg.payload}}
```

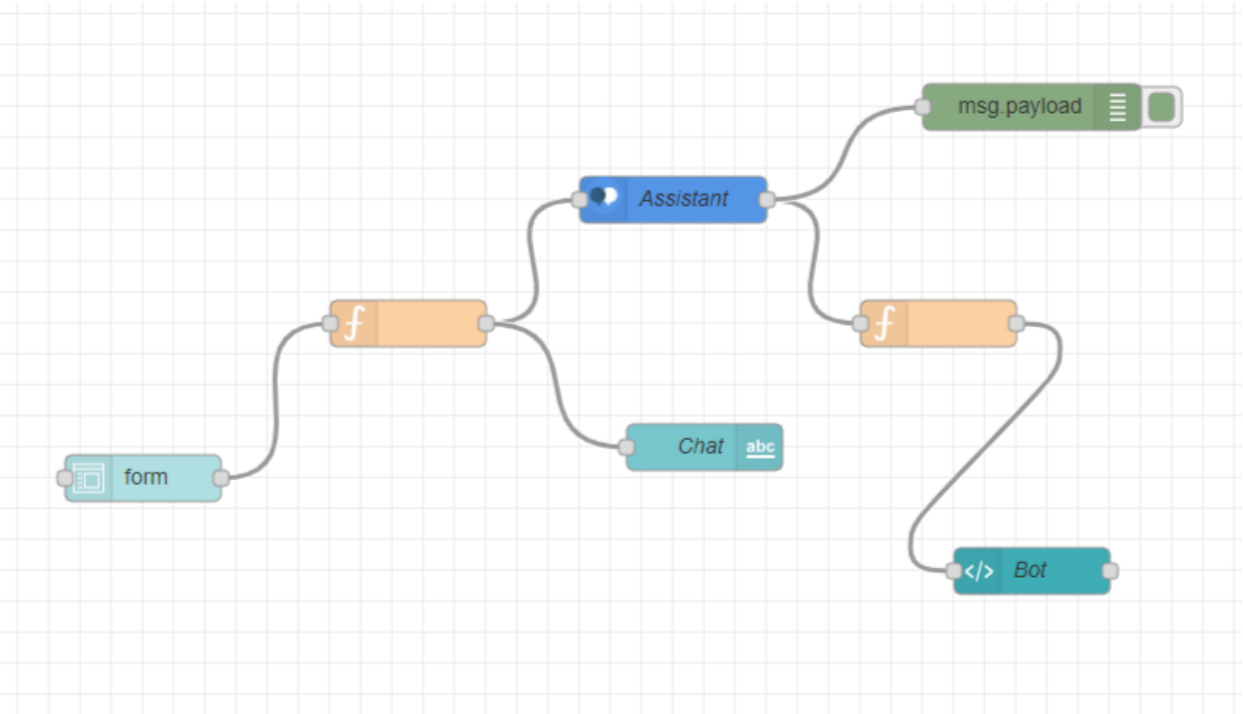
* *

- Configure the assistant by adding the following details: API Key, Service Endpoint, Workspace ID.

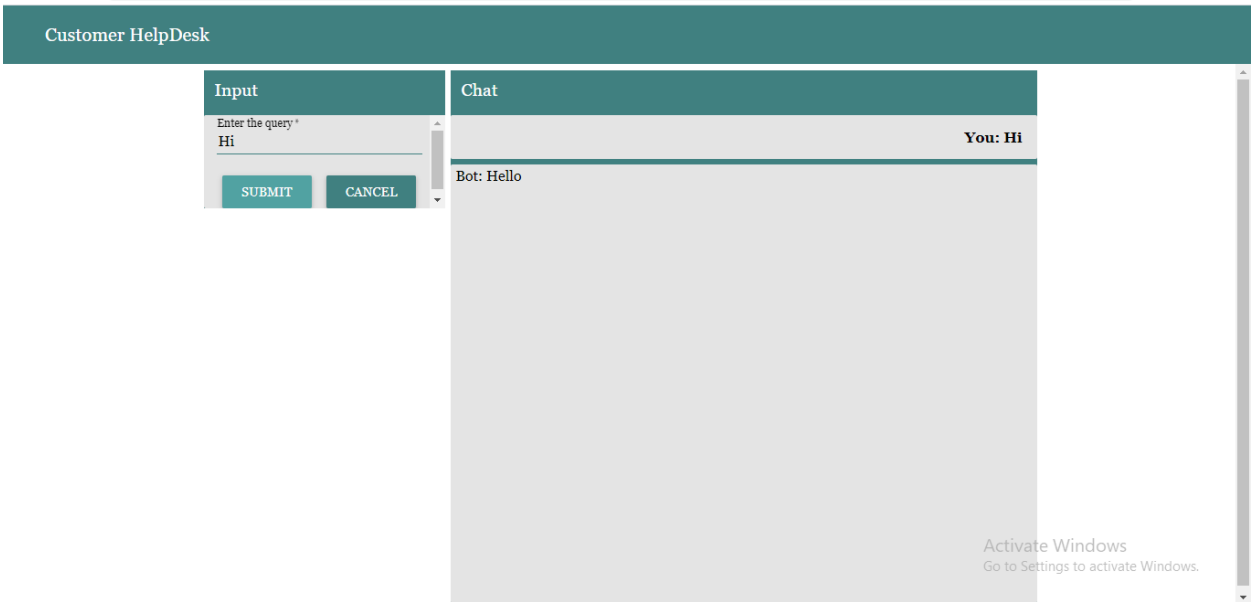
H. Deploy and run the application.

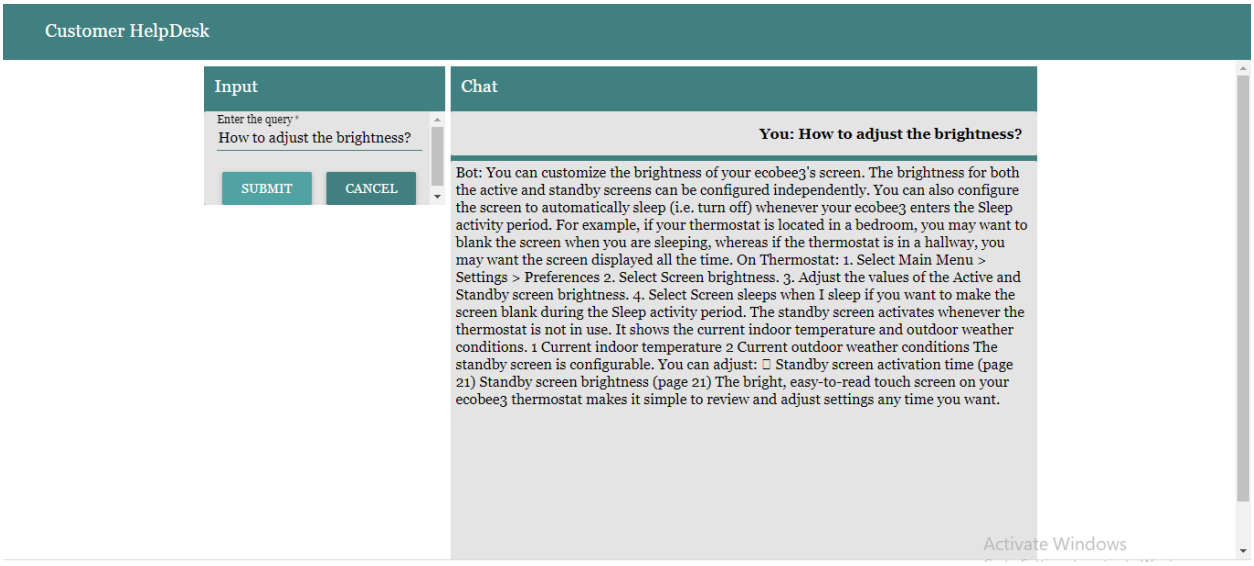
- Deploy the application by using the deploy button on the top right of Node-RED editor.
- Go to the UI to test the application.

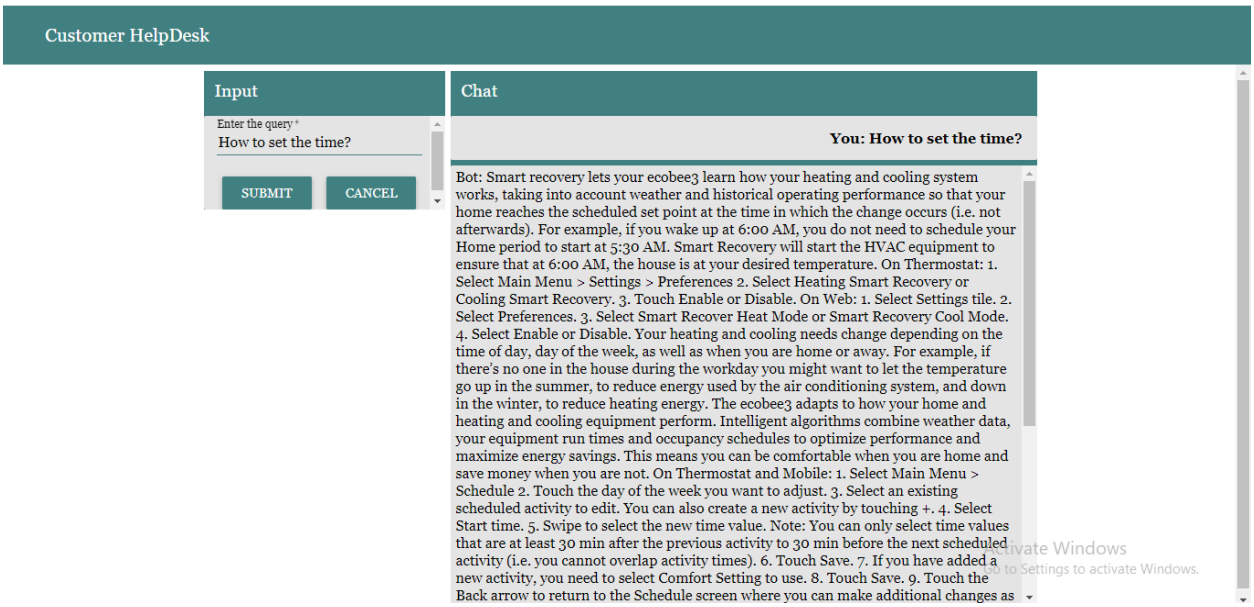
5. Flowchart



6. Result


Customer HelpDesk
Input
Enter the query *
Hi
SUBMIT
CANCEL
Chat
You: Hi
Bot: Hello
Activate Windows
Go to Settings to activate Windows.


Customer HelpDesk
Input
Enter the query *
How to adjust the brightness?
SUBMIT
CANCEL
Chat
You: How to adjust the 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. For example, if your thermostat is located in a bedroom, you may want to blank the screen when you are sleeping, whereas if the thermostat is in a hallway, you may want the screen displayed all the time. On Thermostat: 1. Select Main Menu > Settings > Preferences 2. Select Screen brightness. 3. Adjust the values of the Active and Standby screen brightness. 4. Select Screen sleeps when I sleep if you want to make the screen blank during the Sleep activity period. The standby screen activates whenever the thermostat is not in use. It shows the current indoor temperature and outdoor weather conditions. 1 Current indoor temperature 2 Current outdoor weather conditions The standby screen is configurable. You can adjust: □ Standby screen activation time (page 21) Standby screen brightness (page 21) The bright, easy-to-read touch screen on your ecobee3 thermostat makes it simple to review and adjust settings any time you want.
Activate Windows



The proposed system is able to satisfactorily respond to user queries about the operation of the thermostat among other queries.

7. Advantages & Disadvantages.

Advantages:

- Faster customer service.
- Increased customer satisfaction.
- Lower labour costs.
- Variety of uses.
- 24x7 availability.
- Multiple customer handling.
- Can be integrated with various external services.

Disadvantages:

- Limited responses for customers.
- Customers could become frustrated.
- Maintenance required.
- They aren't human ultimately so cannot be reliable in case of emergencies.
- Complex chatbots could cost more.
- Not all business can use chatbots.

8. Applications

Chatbots can be used for many purposes. Some of them include:

- **Order Pizza.**

  It's ridiculously easy to order pizza with the help of chatbots. You can order by texting, tweeting, voice, or even from your car. Domino's was one of the early adopters of chatbots. Today, Domino's lets you easily build a new pizza (or reorder your favorite pizza) and track your order all from Facebook Messenger.

- **Product Suggestions.**

  Many consumers know they want to buy some shoes, but might not have a particular item in mind. You can use chatbots to offer product suggestions based on what they want (color, style, brand, etc.)

  It's not just shoes. You can replace "shoes" with any other item. It could be clothes, groceries, flowers, a book, or a movie. Basically, any product you can think of.

  For example, tell H&M's Kik chatbot about a piece of clothing you have and they'll build an outfit for you.

- **Specialized tasks.**

  Tasks like hiring a cab, ordering a product, checking the weather, etc. can be easily accomplished by chatbots. Businesses can use chatbots to manage inventory and purchase orders.

- **Using chatbots on mobile devices.**

  Using chatbots in mobile apps enables the creation of more streamlined user interfaces. It allows users to browse, evaluate, purchase, and get support from a single interface.

- **Scale up operations.**

  Human beings have their limitations; an agent may be able to engage with a maximum of 3 customers at a time, whereas AI-based chatbots have no such limits. By fortifying your workforce with chatbots, you can

interact with more users, helping you increase your customer base and even enter new markets.

- **Assist customers in making the right choice.**
  Choosing can become very difficult when the choices are very close, or the customer has to shell out big bucks. Chat bots can easily guide the customer and help them get the right product or service.
- **Upselling.**
  Customers who have completed a purchase are more likely to purchase again. Chatbots can interact with these customers and leverage the opportunity of upselling to them.
- **Interactive marketing.**
  Chatbots represent an active user experience, unlike websites and apps, which are passive. Not only can you use them for interactive marketing, you can reach out to a larger audience; in fact, several people simultaneously. Using chatbots can also simplify several tasks for you when you use online channels like your social media networks to engage with customers.

9. Future Scope

- **Voice interface**

  If the future demands advanced chatbots that do more than use scripted, single-turn exchanges, then their method of interface will also have to advance. A voice interface can assist users with disabilities or those who are skeptical of technology, but it also requires another layer of NLP development.
- The helpdesk can be extended to fulfil a lot of other functionalities as required by the company.
- It can be trained on more scenarios so that it can responds efficiently.
- It can be integrated with external applications for wider use.

10. Conclusion

Thus, we have successfully created and tested an intelligent customer helpdesk using various IBM Cloud services like IBM Watson Discovery, IBM Watson Assistant and IBM Cloud Function. Now, the chatbot is able to answer various user queries and also queries pertaining to the operations of the Ecobee thermostat unlike typical chatbot. It can be further developed and be used for multiple purposes as per availabilty and requirements.

11. Bibliography

1. https://www.ibm.com/cloud/get-started https://developer.ibm.com/patterns/enhance-customer-help-desk-with-smart-documentunderstanding/
2. https://developer.ibm.com/tutorials/how-to-create-a-node-red-starter-application/
3. https://github.com/IBM/watson-discovery-sdu-with-assistant
4. https://cloud.ibm.com/docs/openwhisk?topic=cloud-functions-getting-started

Appendix

A. Source Code

*Cloud function code for integrating Watson Discovery using webhook:

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