PROJECT REPORT

1. Introduction

1.1 Overview

The aim of this project is to build a smart customer help desk system that uses smart document understanding. The user search for their query on the website, the Watson assistant will then either reply to the query based on the entities / dialogues set up, or will look for a relevant answer for the query inside the Device Manual and return the required extract. The search inside the document will be carried out with the help of the Watson Discovery Service. The link between the Watson Discovery Service and the Watson Assitant will be formed with the help of the IBM Cloud Functions. The accuracy of this help desk chat bot can be improved by either training the Watson Assistant, and adding more details about the entities and dialogues, or training the Watson Discovery and annotating the device manual. This bot in particular, has been made to answer the queries related to the Ecobee3 thermostat device.

1.2 Purpose

Most of the service chat bots today can answer simple questions - for example a chatbot for a particular store can answer questions about the store locations or the store timings. However, when the chatbot is asked specific questions about the device or its operation, it usually says that its unable to understand a question, and asks if you want to connect to a customer representative. This however, is not always required for simple questions like how to turn on a device, or how to change a particular setting of a device. Therefore, the following smart help desk, looks for the answers within the device manual, and tries to return relevant extracts from the manual, based on the queries. This makes the chatbot more effective and responsive to the user queries.

2. Literature Survey

2.1 Existing Problem

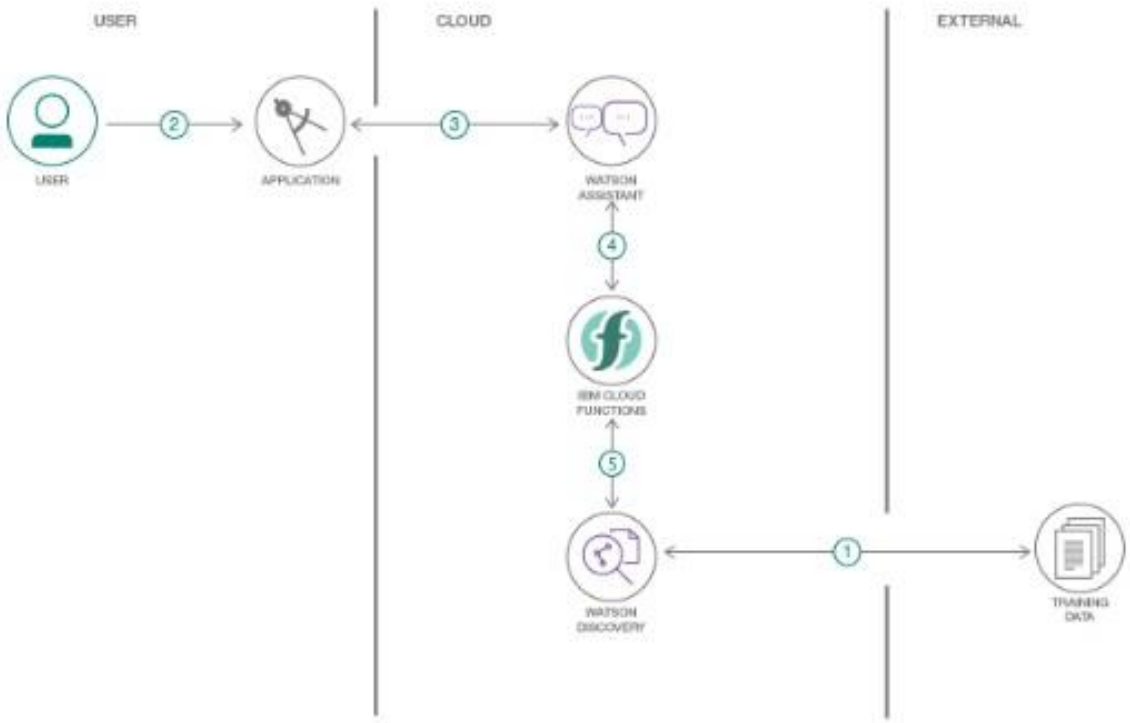
Most of the chat bots available today have limited functioning. They can provide answers for simple requests like showing the store timings or store locations, but whenever the user asks specific questions about the functioning or configuration of the problem, they do not understand the query. In such cases, they often ask the user if they would like to get connected to the an assistant. However, some simple questions like - How do I turn on the device, or how do I adjust the time can simply be answered using the device manual, and thus do not require any human assistance.

2.2 Proposed Solution

The aim of this project is therefore to create an assistant that can look up for queries related to the device in the manual. This has some challenges, as the assistant must understand the user's query and look for relevant extracts in the user manual. This will be carried out with the help of IBM Watson Assistant and Discovery. The Assistant will help with the development of the assistant, and will be able to identify queries that are related to the product, and pass them on to Discovery. The Watson Discovery will read through the user manual, and find relevant extracts based on the query. The user interface for this assistant will be created using the IBM Node Red Services.

3. Theoretical Analysis

3.1 Block Diagram



3.2 Hardware / Software designing

The smart help desk uses the following tools:

A. IBM Watson Assistant:

The Watson services by IBM use deep learning to help learn from unstructured data, in our case the device manual. The IBM Watson Assistant allows the user to build and deploy interactions in various channels. The assistant also allows the queries that are in natural language. For this project, the IBM Watson Assistant has been linked with the Watson Discovery service, that allows the assistant to search for the queries through the device manual as well.

Making the Watson Assistant starts with building a dialogue skills. For a particular skill, the intent, entities and dialogues for the skill must be added. The intent helps the assistant understand the different ways a user might input a query. The intent then helps the assistant

identify key words in the query that can be analysed to decide the result. The responses of the assistant and the reply options are then finally customized in the dialog option.

B. IBM Watson Discovery:

The IBM Watson discovery helps us enhance the results provided by the help desk. IBM Watson Discovery allows the user to ingest, normalize and search through unstructured data. Using the Watson Discovery service, the user can analyse the text in various different ways. This includes Sentiment analysis of the text, concept tagging, category classification. Finally, the service also allows queries to be made on the dataset. The user manual for ecobee3 thermostat was added to this, and then manually annotated to add the text, title, subtitle, footer and table fields.

C. IBM Cloud Functions:

The cloud functions service has been used to connect the IBM Assistant service with the IBM Discovery service. The function uses the environment id, collection id, api key and url of the Discovery service, and passes the input from the user in Assistant to Discovery. The Web Action option in Cloud Function, allows it to handle HTTP Requests. In our program, the webhook will communicate with the IBM Cloud Function web action, which is connected to the Watson Discovery Service.

D. IBM Node Red Services:

Node Red services help us build a simple user interface and dashboard for our help desk and helps us connect Watson Assistant with the same. It is a programming tool for wiring together hardware devices, APIs and other online services. Node Red helps us easily deploy our chatbot service and customize the web page. Some of the commonly used nodes in Node Red flow include the Inject Node (used to manually inject inputs/message), the Debug Node, the Function Node used to parse the message and the Assistant Node to enable Watson Assistant Services.

4. Experimental Investigation

This project involve deploying and connecting numerous the IBM Services. The first step was to set up the Discovery Service. An instance of the Discovery Service was created, and the user manual for the ecobee3 thermostat was added as a private dataset. The results for queries at this point are not very accurate, as the document has not been annotated and the fields are not identified. Therefore the configure data option is used, that allows the user to mark different fields in the text. This includes marking the title, subtitles, texts, tables, footers. Once the user marks these fields for the first few pages, IBM Watson Discovery will according mark the remaining of the document. The accuracy and the efficiency of the system can also be improved by asking Discovery to only index the document according to the text (ignore footers and titles) and split the document based on subtitles. Making these changes, split the pdf into several different documents, and the queries based on the document now return more accurate results. Finally, the collection id, configuration id, environment id, the api key and the url are saved for future connection between Watson Discovery and Watson Assistant.

The next step in the process is to build the Watson Assistant Service. This involves building a skill that can be used by the help desk bot. For this, we can either create a skill from scratch, or use the sample help desk skill provided by the IBM Assistant. In either case, we must first make an intent that can detect the customer making a query about the device. This could include questions like, "How do I access the settings", "How do I set the timer", "How do I turn on the heater". The responses for these queries are then added using the dialog option. We must enable the webhook option, that helps send a POST request from the dialog node to the Watson Discovery URL. The results of such queries are also marked as the responses from the web hook result. The API details of the Assistant are saved as they will be needed to add the Watson Assistant to the Node Red flow.

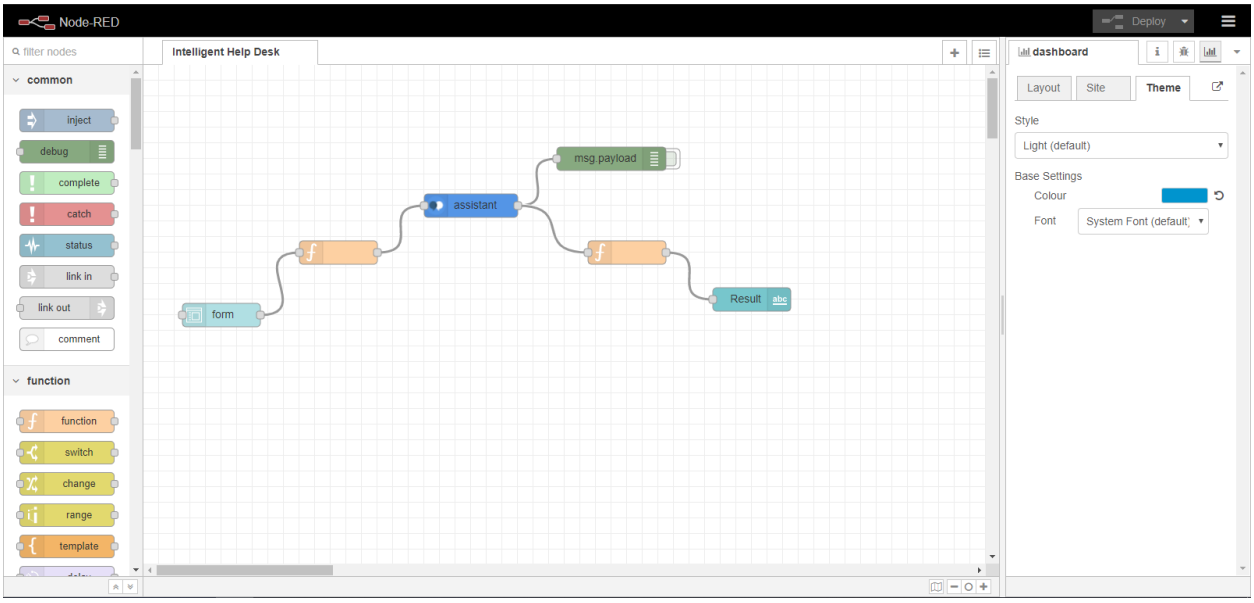
The connection between the Watson Assistant and the Watson Discovery is made with the help of Watson Cloud Function. The function uses the saved details of the Watson Discovery instance to connect with the assistant. This step is crucial, as it also allows us to enable the Webhook services used by Assistant to get results for the queries related to the device.

Finally, all the services can be put together and a user interface created using the Node Red Services. Once the service instance has been created along with the Cloudant Services for hosting, the Node Red flow must be built. The nodes that should be included in the flow are:

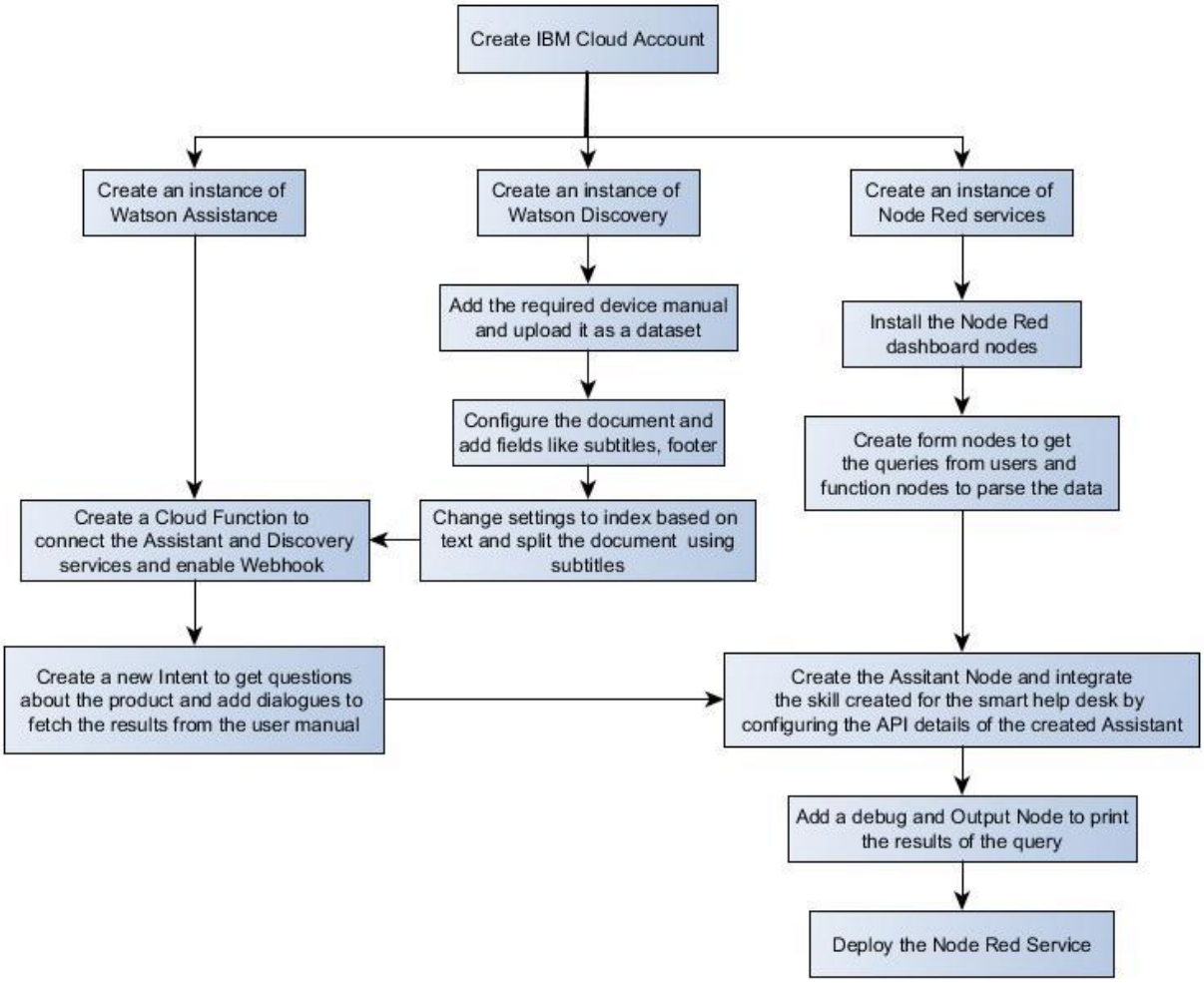
1. Form Node: This node is used to get inputs from the user. It involves customizing the form to set details like the label (place holder), name that is used to store the input and layout.

2. Function Node - One function node to get the input from the user and pass it on the assistant, and another to parse the output given by the Assistant [change it from json format to text].
3. Assistant Node - The assistant node helps us use the Watson Assistant services for out help desk bot. The API details of the assistant must be filled to connect the assistant we created for this project.
4. Debug Node - Used to debug the output.
5. Result Node - Used to print the results for the query.

These nodes must be connected as shown below. Once its done, the deploy option can be used to view the bot and use the smart help desk.



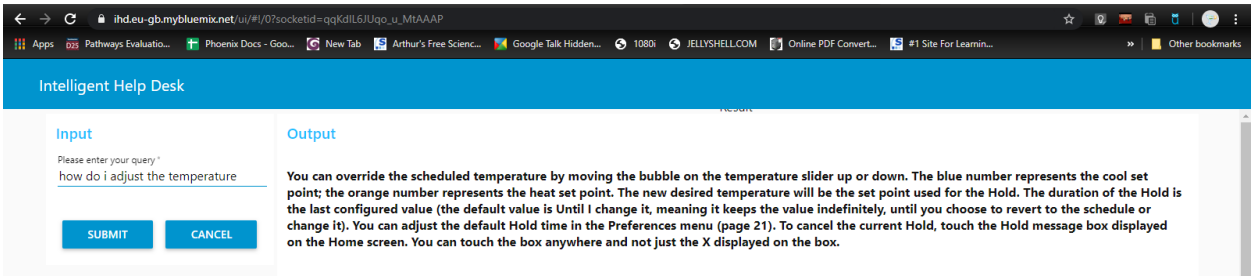
5. Flowchart


Create IBM Cloud Account
Create an instance of Watson Assistance
Create an instance of Watson Discovery
Create an instance of Node Red services
Add the required device manual and upload it as a dataset
Install the Node Red dashboard nodes
Configure the document and add fields like subtitles, footer
Create form nodes to get the queries from users and function nodes to parse the data
Create a Cloud Function to connect the Assistant and Discovery services and enable Webhook
Change settings to index based on text and split the document using subtitles
Create a new Intent to get questions about the product and add dialogues to fetch the results from the user manual
Create the Assitant Node and integrate the skill created for the smart help desk by configuring the API details of the created Assistant
Add a debug and Output Node to print the results of the query
Deploy the Node Red Service

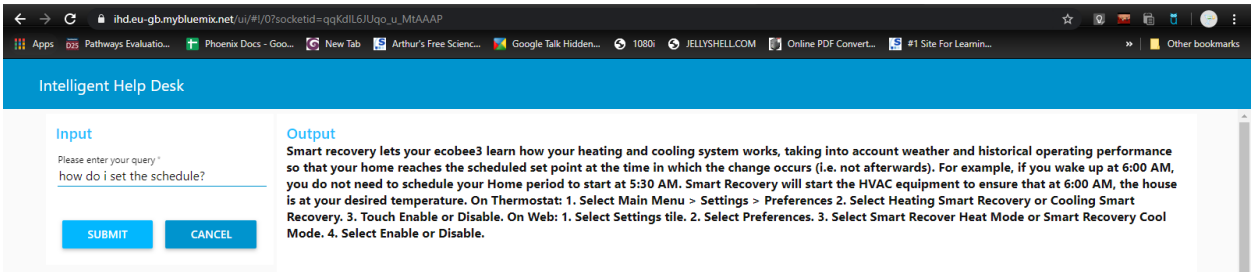
6. Result

The result of this project, is a smart help desk chat bot that is not just able to answer the normal queries about store timings and store locations, but also gives solutions for problems related to the device. The chat bot is able to extract relevant parts from the manual based on the queries of the customer.

For example, when asked about how to change the temperature settings of Ecobee, the chat bot returns:



Similarly, the chat bot can also return answers for queries related to setting a schedule:



Therefore, the intelligent help desk is an advanced version of a normal help desk chat bot that is able to answer a larger variety of questions based on the functioning of the device.

7. Advantages and Disadvantages

Advantages

Using the IBM Cloud and Watson services for the deployment of this chat bot provides us a large range of advantages. The first set of advantages are related to the Watson services. Watson discovery uses deep learning, which allows it to learn from unstructured data (in this case the user manual). This is a big advantage, as most of the data required required for this application is found in unstructured data like device manuals. Watson also uses a technique called transfer learning, which ensures that the model does not have been trained from scratch and that it can use prior knowledge to speed up the training. Watson also protects data and reduces bias to give more accurate results. Finally, the Watson Discovery services also allows the user to train the model using natural language querying.

The next set of advantages are related to the IBM Cloud Functions. The IBM Cloud functions allows us to set up actions based on HTTP based API requests from web apps and from event based requests from services like Cloudant. Additionally, the IBM cloud services also provide us with a server less computing, so that the user does not have to worry about issues like maintaining the server, or other low level infrastructures.

Disadvantages

However, the IBM services also have some disadvantages / limitations. The first limitation is that the IBM Watson and Discovery services are limited to the English language and cannot analyse texts, or reply to queries in other languages. Additionally, the functionality of the services for a free account is very limited. The user can also create 1 data source and 1 instance of the services. Therefore, using the IBM platform for large scale operations will require paid accounts which could be expensive for small scale enterprises.

8. Applications

The systems used for the intelligent help desk can be used for a variety of applications. The current application is limited to a help desk for a particular product, but other chatbots can also be created. For example, we can use the IBM Watson and Discovery services to create a public health chatbot. This chat bot can provide information like causes, symptoms, solutions for various different diseases based on the doctor's manuals / medical textbooks. Other services like booking of movie tickets, making reservations for dinner can also be carried out with the help of these services. Finally, by making use of the natural language features of IBM Watson, we can also use it for creating personal assistants (that can take both text and audio inputs) and carry out simple tasks like show news, make appointments and so on.

9. Conclusion

This project makes use of IBM Watson Discovery, IBM Watson Assistant, IBM Cloud functions and IBM Node Red services to create a smart help desk. The help desk chat bot is able to answer normal queries related to the timings / location of stores, as well as more complex queries based on the functioning of the device. For this, the chat bot makes use of the ecobee3 manual and analyses it to present extracts from the manual based on the query. The IBM Watson Discovery helps us train the model to analyse the text and return accurate results. We also annotate the manual and identify different fields and enrichments using the Watson Discovery Services. The Watson Assistant is used to create a chatbot that understands the user input and then responds to the query based on the dialog configuration set by the user. The IBM Cloud function has been used to integrate Watson Assistant and Discovery. Finally, the IBM Node Red services helps deploy the chat bot and create a simple User Interface for the same.

10. Future Scope

As stated earlier, the services used in this project can be used to build various different products like a Public Health Assistant, or a Personal Assistant. However, the scope of this project itself can also be further increased. For example, more datasets can be added for the ecobee3 device, including customer guides, that will help return more accurate results for the queries. Device manuals for other products can also be added to the database and annotated, so that the help desk can answer similar questions for a wide variety of products. Finally, more advanced features like Speech recognition for can also be added to the project, to allow the user to give speech input.

11. Bibliography

1. Cloud.ibm.com. 2020. *IBM Cloud Docs*. [online] Available at: <https://cloud.ibm.com/docs/assistant?topic=assistant-getting-started#getting-started-tutorial>

2. Cloud.ibm.com. 2020. *IBM Cloud Docs*. [online] Available at: <https://cloud.ibm.com/docs/services/discovery?topic=discovery-getting-started>

3. IBM Developer. 2020. *Create A Node-RED Starter Application*. [online] Available at: <https://developer.ibm.com/components/node-red/tutorials/how-to-create-a-node-red-starter-application/>

4. IBM Developer. 2020. *Create An App To Perform Intelligent Searches On Data*. [online] Available at: <https://developer.ibm.com/patterns/create-an-app-to-perform-intelligent-searches-on-data/>

5. IBM Developer. 2020. *Introduction To Watson Discovery*. [online] Available at: <https://developer.ibm.com/articles/introduction-watson-discovery/>

6. IBM Developer. 2020. *Learning Path: Getting Started With Watson Assistant*. [online] Available at: <https://developer.ibm.com/components/watson-assistant/series/learning-path-watson-assistant>

7. IBM Developer. 2020. *Learning Path: Getting Started With Watson Discovery*. [online] Available at: <https://developer.ibm.com/series/learning-path-watson-discovery/>

8. Ibm.com. 2020. *IBM Watson Products And Solutions*. [online] Available at: <https://www.ibm.com/watson/products-services>

9. Medium. 2020. *Node-Red Watson Discovery Chatbot Telegram*. [online] Available at: <https://medium.com/@rimaibrahim/node-red-watson-discovery-chatbot-telegram-ce616ddcd0d9>

Appendix

A. Source Code:

IBM 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');

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