Project Report

on

Intelligent Customer Helpdesk with Smart Document Understanding

in

Machine learning / Artificial Intelligence

by

RUCHI

(ruchika24majoka@gmail.com)

Intelligent Customer Help Desk with Smart Document Understanding

1. Introduction

1.1 Overview

We will build a chatbot that uses various Watson AI Services (Watson Discovery, Watson Assistant, Watson Cloud Functions and Node-Red) to deliver an effetive Web based UI through which we can chat with the assistant.

We will integrate the Watson Discovery service with Watson Assistant using webhooks.

1. Project Requirements : Node-RED, IBM Cloud, IBM Watson, Node JS
2. Functional Requirements : IBM Cloud
3. Technical Requirements : AI, ML, Watson AI, Node JS
4. Software Requirements : Watson Assistant, Watson Discovery, Watson Cloud Functions, Node-RED
5. Project Deliverables : Intelligent Chatbot with Smart Document Understanding
6. Project Team : Shane Sam A
7. Project Duration : 30 Days

1.2 Purpose

The typical customer care chatbot can answer simple ques ons, such as store loca ons and hours, direc ons, and maybe even making appointments. When a ques on falls outside of the scope of the pre-determined ques on set, the op on is typically to tell the customer the ques on isn't valid or offer to speak to a real person.

In this project, there will be another op on. If the customer ques on is about the opera on of a device, the applica on shall pass the ques on onto Watson Discovery Service, which has been pre-loaded with the device's owners manual. So now, instead of "Would you like to speak to a customer representa ve?" we can return relevant sec ons of the owners manual to help solve our customers' problems. So unless and un ll customer specifically asks for a customer representa ve 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 owners manual is important and what is not. This will improve the answers returned from the queries. Then using Watson ac ons 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.

1.2.1 Scope of Work

1. Create a customer care dialog skill in Watson Assistant
2. Use Smart Document Understanding to build an enhanced Watson Discovery collection
3. Create an IBM Cloud Functions web action that allows Watson Assistant to post queries to Watson Discovery
4. Build a web application with integration to all these services & deploy the same on IBM Cloud Platform

2. Literature Survey

2.1. Exis ng Problem

The typical customer care chatbot can answer simple ques ons, such as store loca ons and hours, direc ons, and maybe even making appointments. When a ques on falls outside of the scope of the pre-determined ques on set, the op on is typically to tell the customer the ques on isn't valid or offer to speak to a real person.

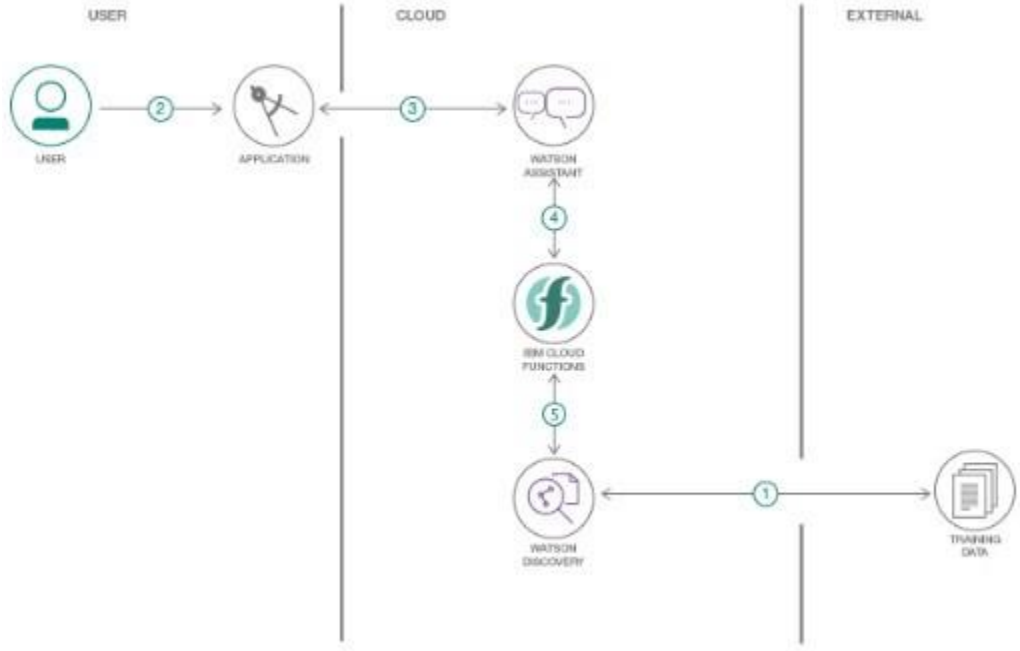
2.2. Proposed Solu on

In this project, there will be another op on. If the customer ques on is about the opera on of a device, the applica on shall pass the ques on onto Watson Discovery Service, which has been pre-loaded with the device's owners manual. So now, instead of "Would you like to speak to a customer representa ve?" we can return relevant sec ons of the owners manual to help solve our customers' problems. So unless and un l customer specifically asks for a customer representa ve 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 owners manual is important and what is not. This will improve the answers returned from the queries. Then using Watson ac ons 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.

3. Theore cal Analysis

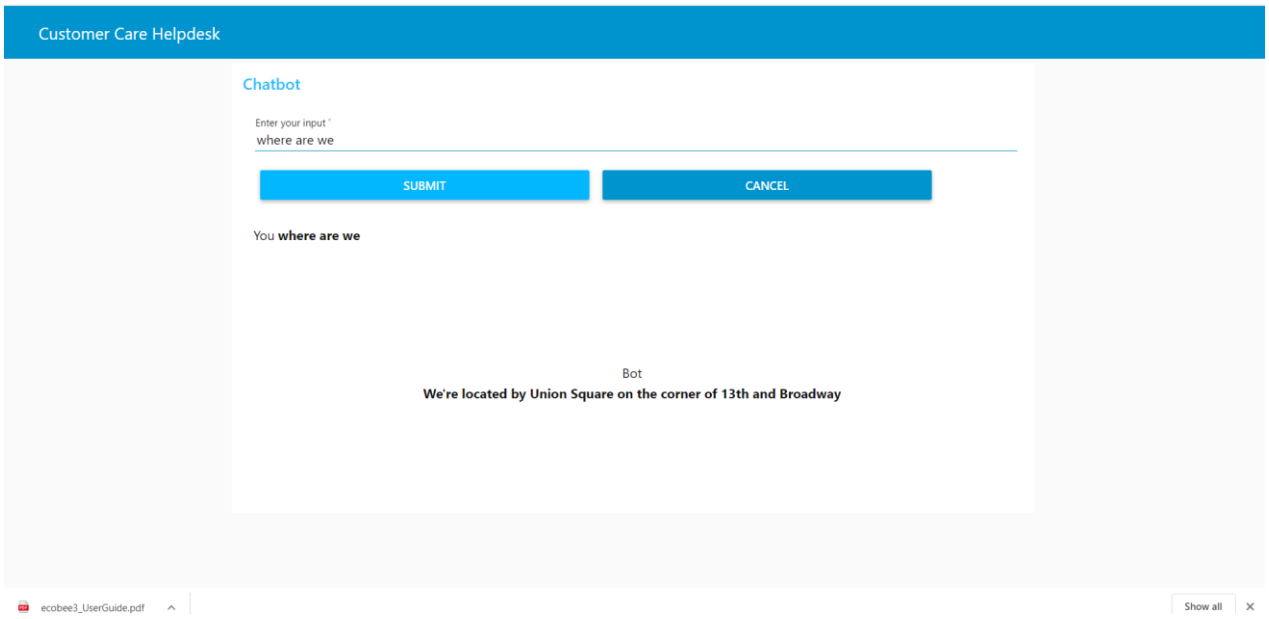
Block / Flow Diagram



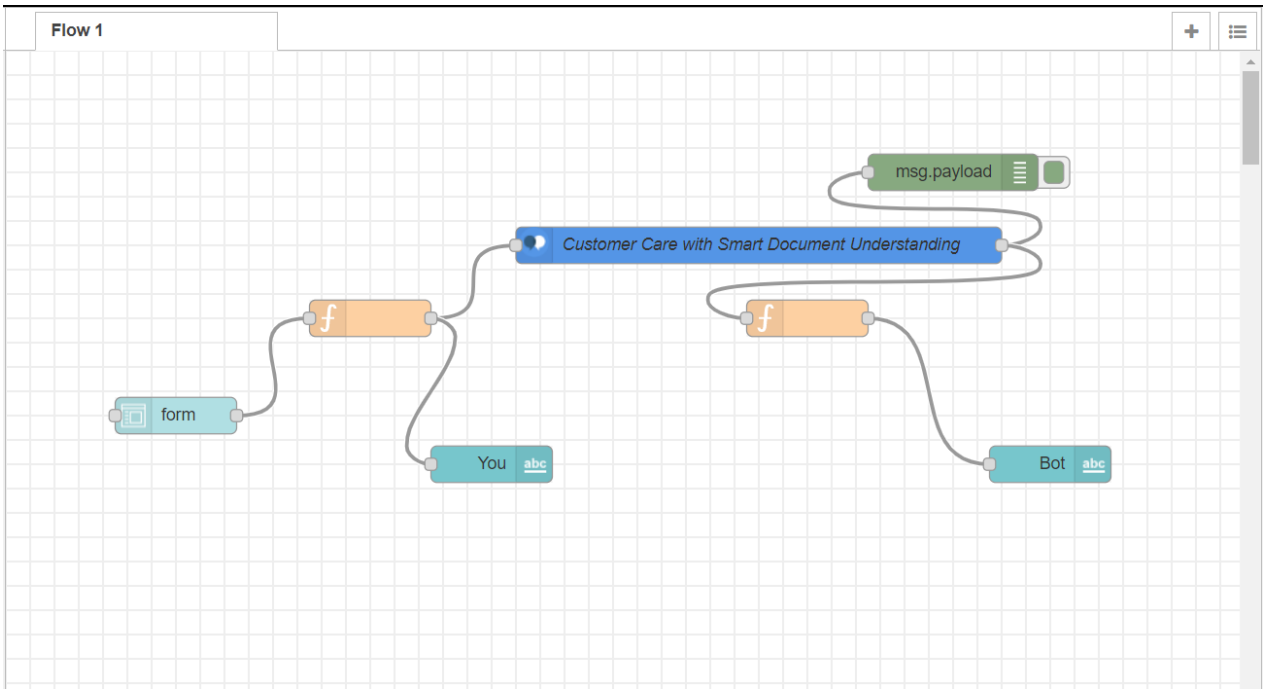
Hardware / So ware Designing

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.

5. Experimental Investigation


Customer Care Helpdesk
Chatbot
Enter your input *
where are we
SUBMIT
CANCEL
You where are we
Bot
We're located by Union Square on the corner of 13th and Broadway
ecobee3_UserGuide.pdf
Show all

5. Flowchart



Insert the following nodes into the flow in Node-RED.

1. Inject
2. Debug
3. ui_Form
4. ui_Text
5. ui_Button
6. Function
7. Switch
8. Assistant

6. Results

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

URL for UI Dashboard :
https://node-red-zmpjb.eu-gb.mybluemix.net/ui/#!/0?socketid=QGUjexVrHLCwdXpxAAAH

7. Advantages & Disadvantages

Advantages

1. Reduces Man Power.
2. Reduces cost.
3. Less calls will be diverted to Customer Representatives.
4. It saves time.
5. Customer reprsentatives will be able to focus on other aspects of their job.

Disadvantages

1. Sometimes it can mislead cutomers as it tries to search irrelevant information in the manual.
2. It may also give same answers to different queries.
3. It needs regular correction and maintenance

8. Applica ons

1. This chatbot can be deployed in various websites as it can solve a lot of basic questions.
2. It can be deployed in the customer helpdesk of small regular-use products as their manual usually has the required solutions to the user's problems.

9. Conclusion

An Intelligent Customer Helpdesk Chatbot was created successfully using various Watson services like Watson Discovery, Watson Assistant, Watson Cloud Functions and Node-RED.

10. Bibliography

1. Node-RED Starter Application : https://developer.ibm.com/tutorials/how-to-create-a-node-red-starter-applicat ion/
2. Build your oen 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://developer.ibm.com/articles/introduction-watson-discovery/

Appendix

Source Code

Node-RED Flow code

[{"id":"1933c1ac.f23f4e","type":"tab","label":"Flow 1","disabled":false,"info":""},{"id":"e399994e.151a88","type":"ui_form","z":"1933c1ac.f23f4e","name":"","label":"","group":"7def4b8a.21b4b4","order":1,"width":0,"height":0,"options":[{"label":"Enter your input","value":"text","type":"text","required":true,"rows":null}],"formValue":{"text":""},"payload":"","submit":"submit","cancel":"cancel","topic":"","x":140,"y":300,"wires":[["fdb9693d.93d9f8"]]},{"id":"fdb9693d.93d9f8","type":"function","z":"1933c1ac.f23f4e","name":"","func":"msg.payload= msg.payload.text;\nreturn msg;","outputs":1,"noerr":0,"x":300,"y":220,"wires":[["3c72bf0b.22f38","d8e0c3e8.fa342"]]},{"id":"10568c87.c891b3","type":"function","z":"1933c1ac.f23f4e","name":"","func":"msg.payload = msg.payload.output.text[0];\nreturn msg;","outputs":1,"noerr":0,"x":660,"y":220,"wires":[["ae31d58f.e95db8"]]},{"id":"3c72bf0b.22f38","type":"watson-conversation-v1","z":"1933c1ac.f23f4e","name":"Customer Care with Smart Document Understanding","workspaceid":"4b61a0d6-9c41-45fd-a70b-699ab7a3381e","multiuser":false,"context":true,"empty-payload":false,"service-endpoint":"https://api.eu-gb.assistant.watson.cloud.ibm.com/instances/a73177b3-2a67-4c3a-a3de-b2a3cc429c55","timeout":"","optout-learning":false,"x":620,"y":160,"wires":[["39609d18.b86d12","10568c87.c891b3"]]},{"id":"d8e0c3e8.fa342","type":"ui_text","z":"1933c1ac.f23f4e","group":"7def4b8a.21b4b4","order":2,"width":0,"height":0,"name":"","label":"You","format":"{{msg.payload}}","layout":"row-left","x":400,"y":340,"wires":[]},{"id":"39609d18.b86d12","type":"debug","z":"1933c1ac.f23f4e","name":"","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"false","x":780,"y":100,"wires":[]},{"id":"ae31d58f.e95db8","type":"ui_text","z":"1933c1ac.f23f4e","group":"7def4b8a.21b4b4","orde

r":3,"width":16,"height":5,"name":"","label":"Bot","format":"{{msg.payload}}","layout":"col-center","x":860,"y":340,"wires":[]},{"id":"7def4b8a.21b4b4","type":"ui_group","z":"","name":"Chatbot","tab":"ad5c2674.924b08","order":1,"disp":true,"width":16,"collapse":false},{"id":"ad5c2674.924b08","type":"ui_tab","z":"","name":"Customer Care Helpdesk","icon":"dashboard","disabled":false,"hidden":false}]

Watson Cloud Func on Ac on Code

```
/**
 *
* @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.collec on_id
* @param {string} params.configura on_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 ac on
 *
* @param Cloud Func ons ac ons accept a single parameter, which must be a JSON object.
 *
* @return The output of this ac on, which must be a JSON object.
```

```
*   */
func on main(params) {
  return new Promise(func on (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,
     'collec on_id': params.collec on_id,
     'natural_language_query': params.input,
     'passages': true,
     'count': 3,
     'passages_count': 3
}, func on(err, data) {
     if (err) {
return reject(err);
```

```
    }
    return resolve(data);
   });
  });
}
```