**PROJECT REPORT**

**INTELLIGENT CUSTOMER HELP DESK WITH SMART DOCUMENT UNDERSTANDING**

**SUBMITTED BY**

**MOHINI BATRA**

**INTRODUCTION**

* OVERVIEW

This project focuses on a chat bot which not only serves the purpose of a typical chat bot but also when customers ask a question about the working of a particular device it gives them the answer from owner’s manual which is preloaded in the Watson discovery.

* PURPOSE

The main purpose of this project is to make a chat bot which can answer the questions of the customer when they ask about the operation of a device or a machine.

**LITERATURE SURVEY**

* EXISTING PROBLEM

Whenever the customer asks a question about the operation of a device to the chat bot it either gives an option of speaking to its customer care executive or the chat bot replies that it is not trained for such question.

* PROPOSED SOLUTION

The solution to this problem is that our chat bot takes the customers question about the operation of a device and takes it to the Watson discovery which is preloaded with the owner’s manual and returns the paragraph of the that manual which can answer the customer’s question appropriately.

**THEOTRICAL ANALYSIS**

* BLOCK DIAGRAM

CLOUD FUNCTIONS

WATSON DISCOVERY

WATSON ASSISTANT

USER

* SOFTWARE DESIGNING

In our code pattern, we will provide another option. If the customer question is about the operation of a device, we will use the webhook feature of Watson Assistant to pass the question onto our 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.

To take it a step further, we will 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.

In summary, this code pattern will:

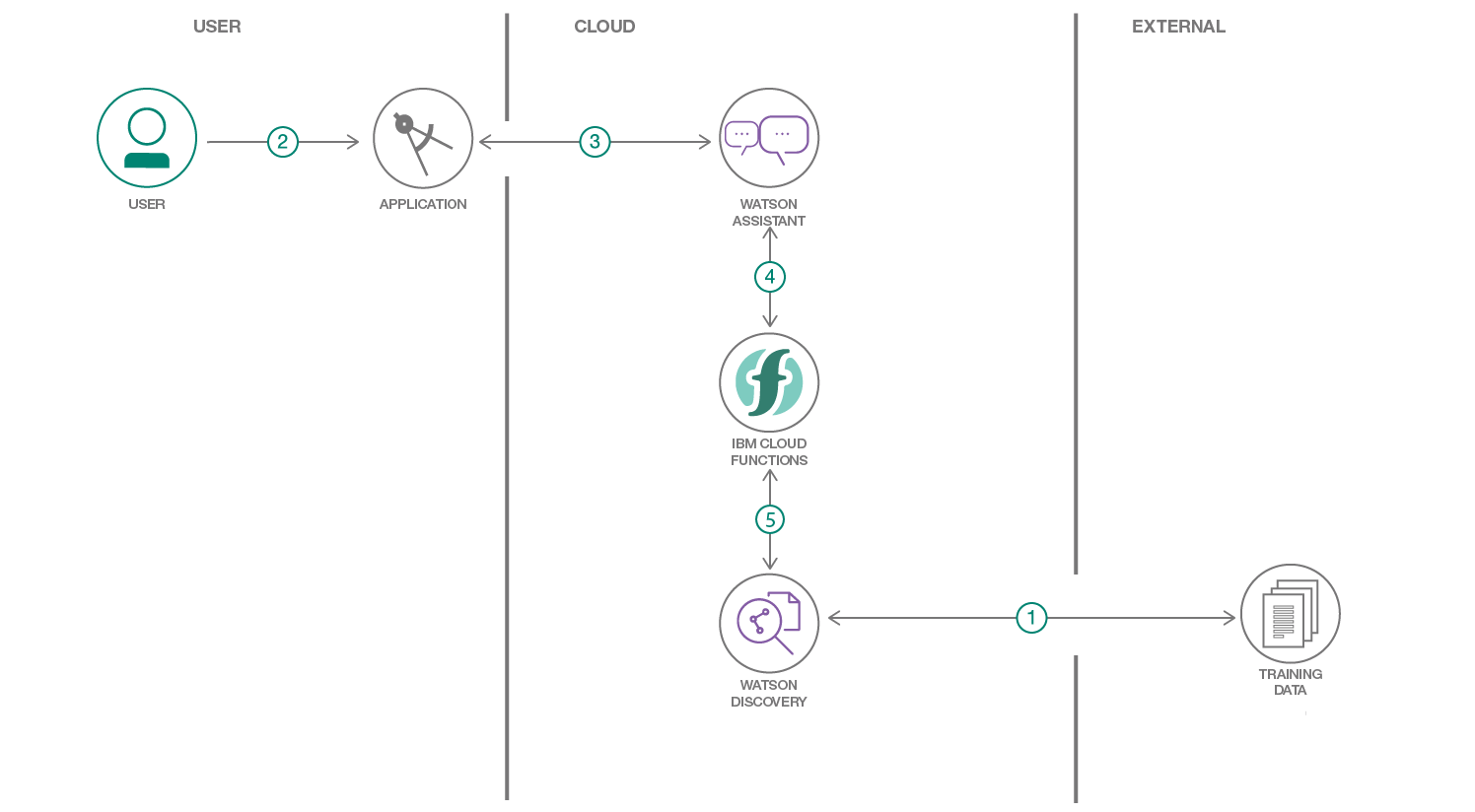
* Create a customer care dialog skill in Watson Assistant
* Use Smart Document Understanding to build an enhanced Watson Discovery collection
* Create an IBM Cloud Functions web action that allows Watson Assistant to post queries to Watson Discovery

**EXPERIMENTAL INVESTIGATIONS**

Our chat bot works in the following manner-

* The document is annotated using Watson Discovery Smart Document Understanding.
* The user interacts with the back-end server via the app UI. The front-end app UI is a chatbot that engages the user in a conversation.
* Dialog between the user and back-end 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 IBM Cloud Functions action will query the Watson Discovery Service and return the results.

**FLOWCHART**



**RESULT**

The result of this project is that whenever a customer is going to ask about a question about the operation of the device it will go the Watson discovery where with the help of smart document understanding it will return the relevant paragraph from the owner’s manual.

**ADVANTAGES**

There following advantages of this project-

* It will work as a typical chat bot and answer all the typical questions that user asks
* But the biggest advantage is that it will give quick result to the customer when they are going to ask about the operation of a device
* Even if then the user is not satisfied it will give them an option of speaking with their customer care executive or technical assistant
* The company will have profit since they don’t have to pay extra for customer service executive or technical skill assistants.

**DISADVANTAGES**

It has the following disadvantages-

* Sometimes it gets difficult for the chat bot that what user it trying to ask them
* It is more easy for the customer to explain their problem to a human technical representative instead of a chat bot

**APPLICATIONS**

This chat bot can be used in various places such as-

* For electronic device websites.
* For electrical device websites.
* For computer or laptop accessories

**CONCLUSION**

This project pattern explained how to use the Smart Document Understanding feature of Watson Discovery to train your dialog to call out to other IBM Watson services for additional sources of information and how to connect all these using node-red services of IBM. This can make the work of technical assistants very easy since they don’t have to answer every simple query and company can save a lot of money using this.

**FUTURE SCOPE**

This project can be further developed for HR of the companies and can be used for hiring process in the companies. We can use the smart document understanding feature of the Watson discovery in the chat bot and the chat bot can analyse the resume of the candidate. It can look for the particular keywords in the Resume of the candidate and if they find the skills in the Resume or the experience in the Resume they looking in a candidate. Then the chat bot can shortlist that candidate and can give him the time for the interview.

**BIBILOGRAPHY**

This project was assigned to me by smart bridge and I referred to IBM boot camp videos and smart bridge mentor’s session for making this project.

**APPENDIX**

SOURCE CODE

CLOUD FUNCTION 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.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);

});

});

}