

Intelligent Customer Help Desk with Smart Document Understanding

Category : Artificial Intelligence

Skills Required : Node.js, IBM Cloud, IBM Watson

Name : Eshwar Nagaraj

Reg.No : 17BF1A04D1(B.Tech 3rd Year)

College : Sri Venkateswara College Of Engineering

Table of Contents :

S.No	Contents	Page N.O
1.	Introduction 1.1 Purpose 1.2 Overview	1
2.	Literature Survey 2.1 Existing problem 2.2 Proposed solution	2
3.	Theoretical Analysis 3.1 Block Diagram 3.2 Hardware / Software designing	3
4.	Flowchart	4
5.	Experimental Investigations	4
6.	Result	20
7.	Advantages and Disadvantages	21
8.	Applications	22
9.	Conclusion	23
10.	Future Scope	23
11.	Bibilography	23
12.	Appendix (Source Code)	23

1. INTRODUCTION

1.1 Purpose

In this project, we walk through a working example of a web app that utilizes multiple 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**.

1.2 Overview

In summary, this project 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
- Create a flow in **Node-RED** and test the bot to check whether it is giving accurate results or not.

In this project, we use the typical customer care chatbot experience but instead of relying on predefined responses, our dialog will provide a hook that can call out to other IBM Watson services for additional sources of information. In our case, it will be an owner's manual that has been uploaded into Watson Discovery.

2. LITERATURE SURVEY

2.1 Existing Problem

The typical customer care chatbot can answer simple questions, such as locations and hours, directions, and may be 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 is not valid or offer to speak to a real person.

2.2 Proposed Solution

In this project, 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 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 the customer's problems.

To take it a step further, we will use the **Smart Document Understanding (SDU)** 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.

What is SDU?

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

from queries. With SDU, we annotate the fields within the documents to train custom conversion models. As we annotate, Watson is learning and will start predicting annotations. SDU models can also be exported and used on other collections.

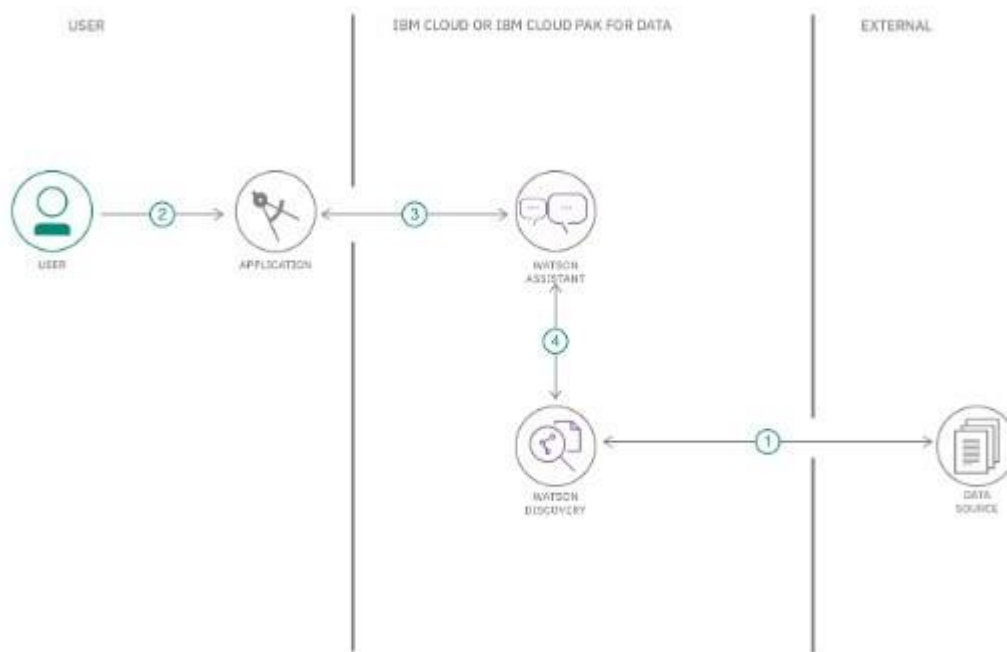
What is a Webhook?

A webhook is a mechanism that allows to call out an external program based on something happening in our 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 we specify or collect from the user during conversation and save in context variables and sends the data to the Webhook request URL. 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 our example, the webhook will communicate with an IBM Cloud Functions web action, which is connected to the Watson Discovery Service.

3. THEORITICAL ANALYSIS

3.1 Block Diagram



3.2 Hardware / Software Designing

Software Required : IBM Cloud, Node-RED

- IBM Cloud offers the most open and secure public cloud for business, a nextgeneration hybrid multicloud platform, advanced data and AI capabilities, and deep enterprise expertise across 20 industries.

- Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways. It provides a browser-based editor that makes it easy to wire together flows using the wide range of nodes in the palette that can be deployed to its runtime in a single-click. Javascript functions can be created within the editor using a rich text editor. The flows created in Node-RED are stored using JSON which can be easily imported and exported for sharing with others. Node-RED is built on Node.js

4. FLOWCHART

1. The document is annotated using Watson Discovery Smart Document Understanding.
2. 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.
3. Dialog between the user and back-end server is coordinated using a Watson Assistant dialog skill.
4. If the user asks a product operation question, a search query is passed to a predefined IBM Cloud Functions action.
5. The IBM Cloud Functions action will query the Watson Discovery Service and return the results.

5. EXPERIMENTAL INVESTIGATIONS

Step – 1 : Create IBM Cloud Services

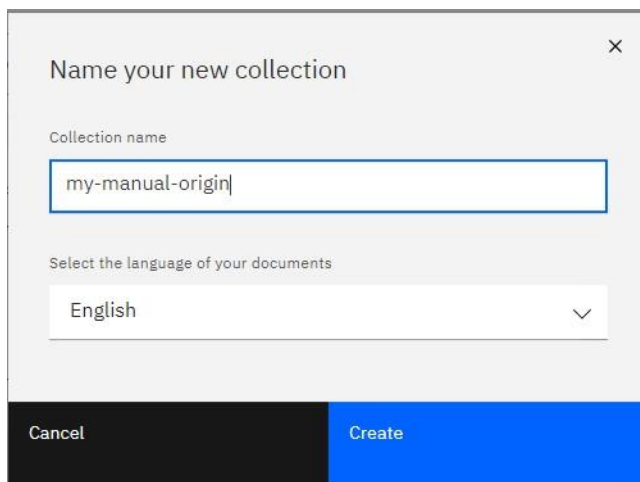
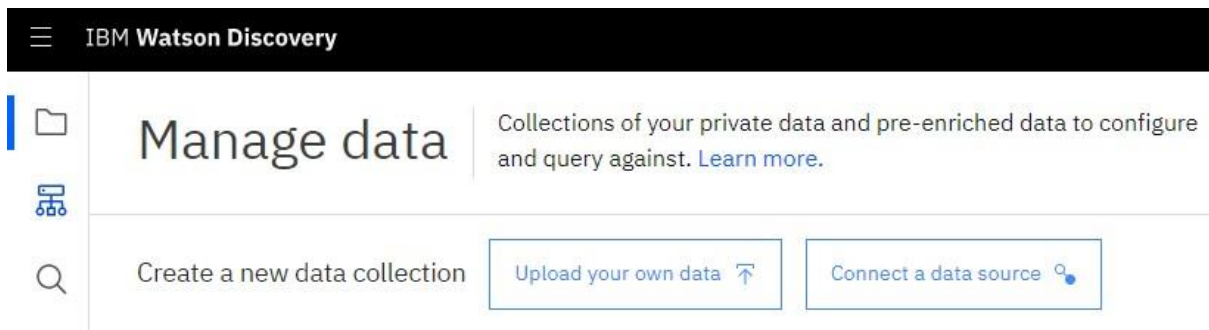
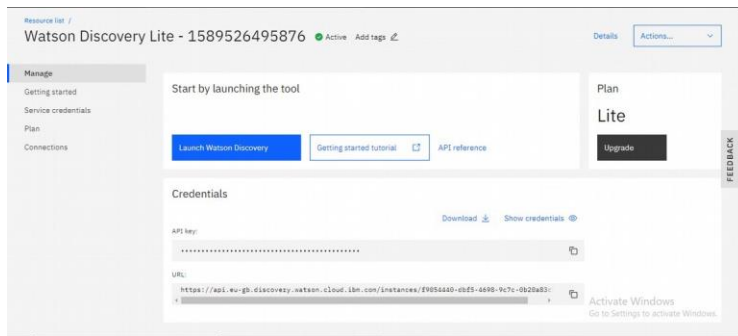
Create the following services : i) Watson Discovery

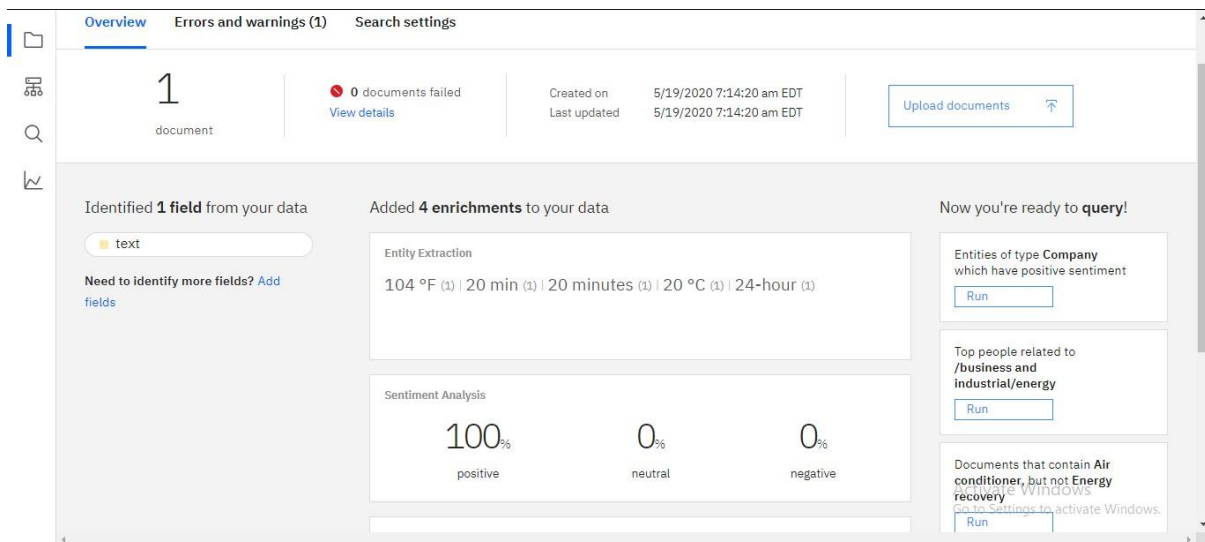
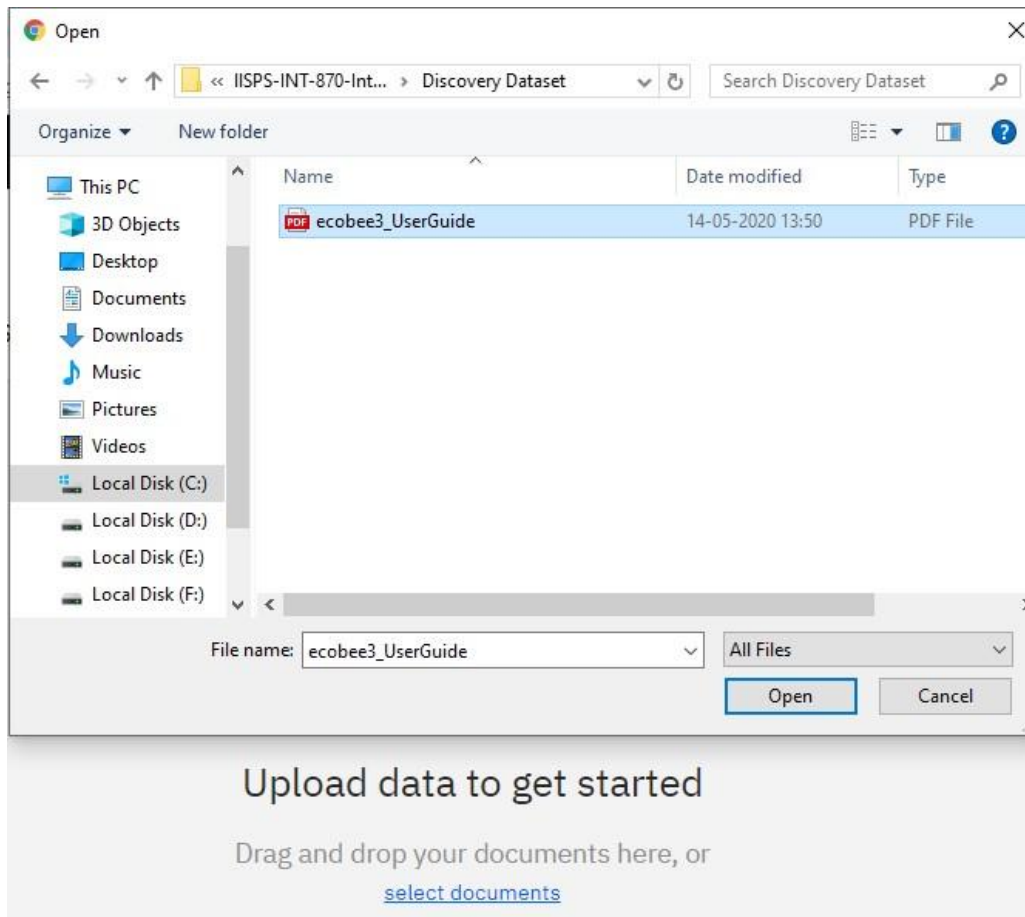
ii) Watson Assistant

Step – 2 : Configure Watson Discovery

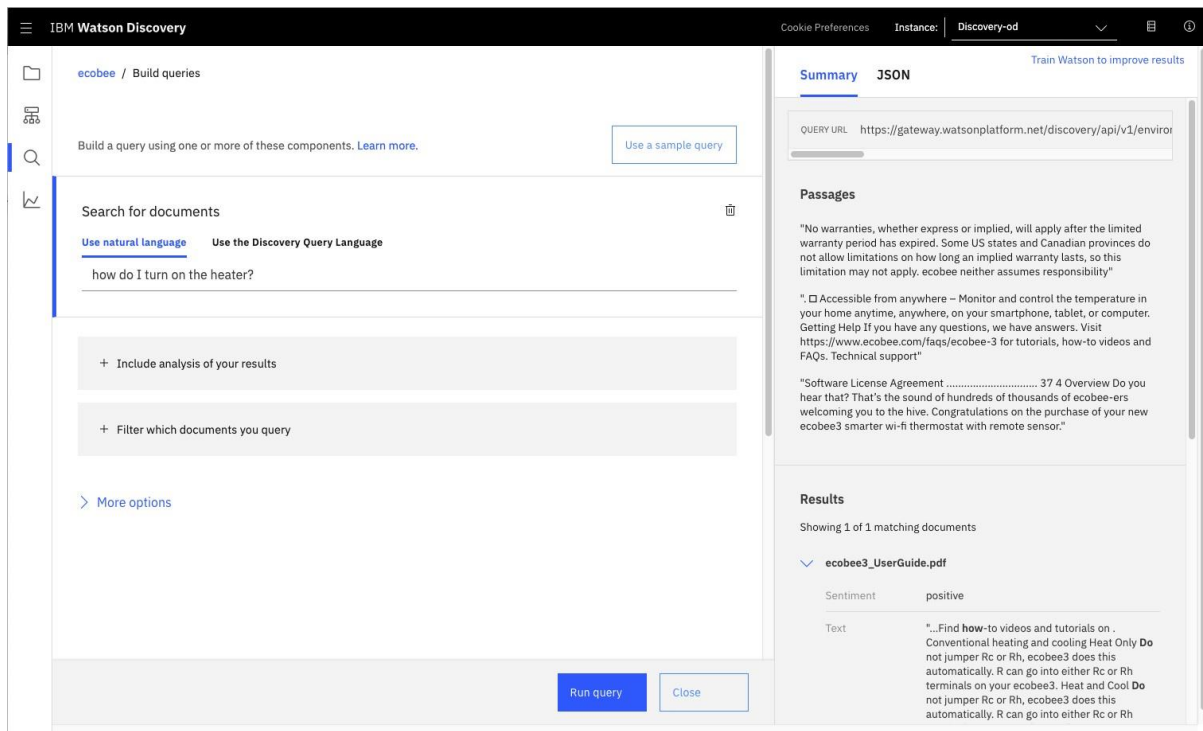
- Import the document

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 promoted, select and upload the **ecobee3_UserGuide.pdf** file located in the local storage. The Ecobee is a popular residential thermostat that has a wifi interface and multiple configuration options.





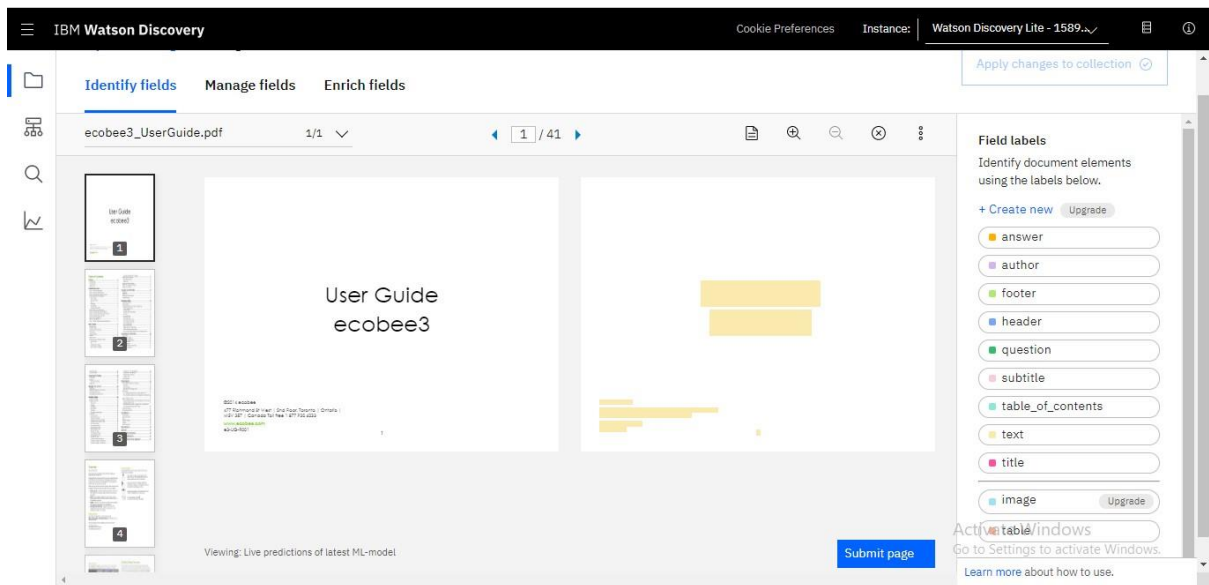
Before applying SDU to our document, let's do some simple queries on the data so that we can compare it to results found after applying SDU. Click the **Build your own query button**. Enter queries related to the operation of the thermostat and view the results. The results are not very useful, and in some cases, not even related to the question.



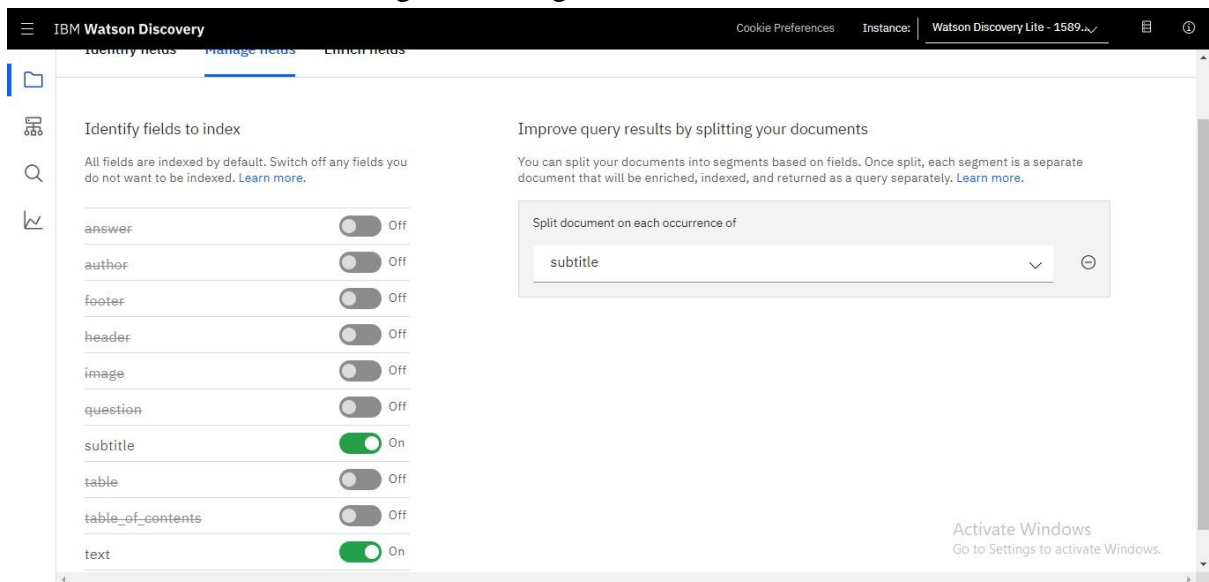
- Annotate with SDU

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 to start the SDU process. The goal is to annotate all of the pages in the document so Discovery can learn what text is important, and what text can be ignored.

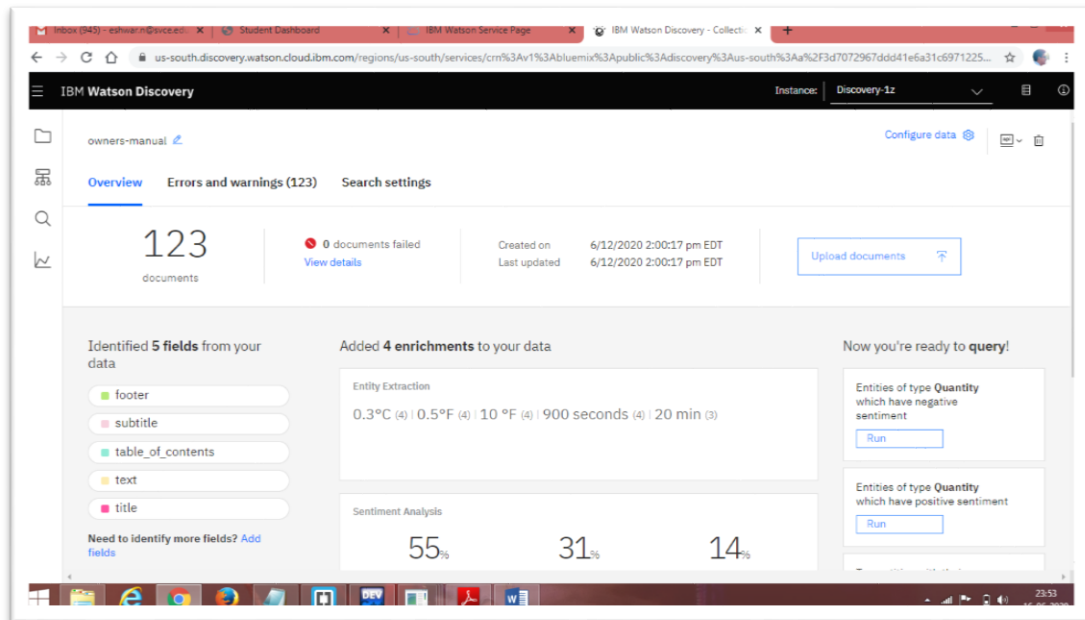
Select the text and assign it a label. For example subtitle, text, footer, table of contents etc., Submit the page to Discovery. As each page is processed, a green check mark will appear on the page. As you go through the annotations one page at a time, Discovery is learning and should start automatically updating the upcoming pages. Once we get to a page that is correctly annotated, we can stop or simply click Submit to acknowledge it is correct. The more pages we annotate, the better the model will be trained. Once we click on the **Apply changes to collection** button, we will be asked to reload the document. Choose the same owner's manual .pdf document as before.



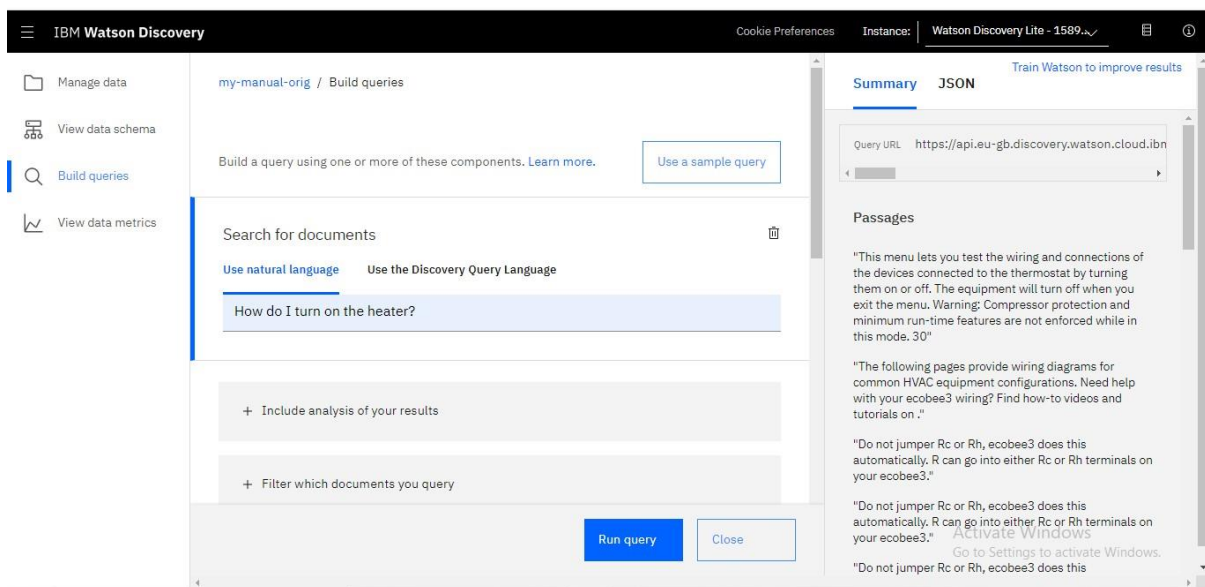
Now, click on the **Manage fields** tab. Tell Discovery which fields to ignore using the on/off buttons, turn off all labels except **subtitles** and **text**. Split the document based on **subtitle**. Submit the changes. Once again, we will be asked to reload the document.



Now as a result of splitting the document apart, our collection will look very different.



Return to the query panel and see how much better the results are :



In upcoming steps, we will need to provide the credentials to access Discovery collection. We will be needing **Collection Id, Environment Id**.

Configure data

Collection ID

62541ad0-dfac-4341-ad0a-f1ad987ad1ba

Configuration ID

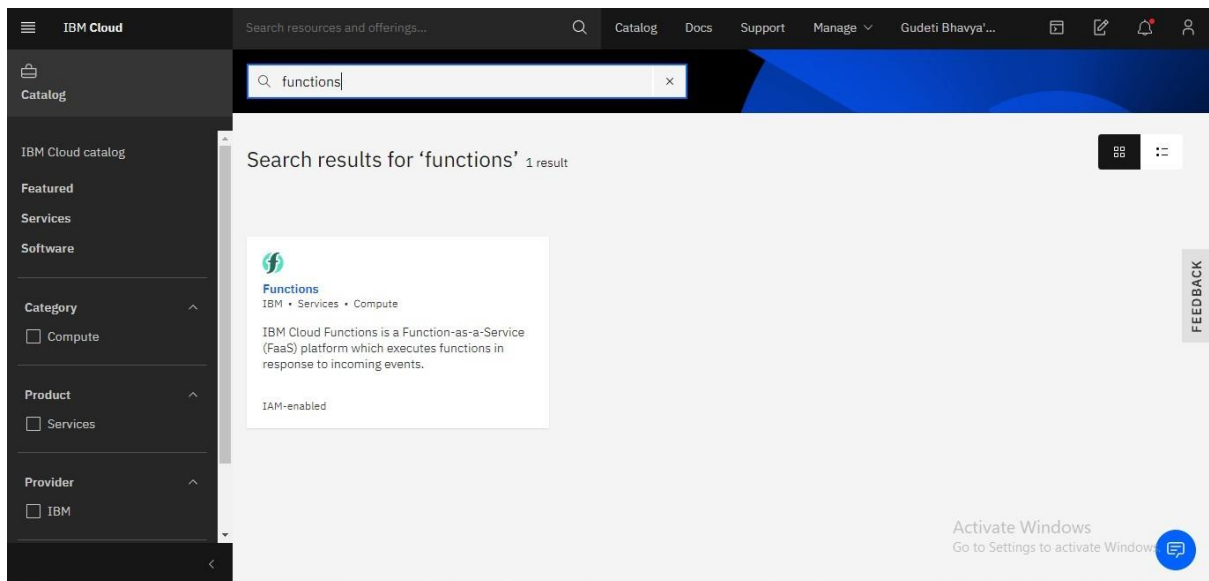
cad33b42-6cdf-4bd7-9cdd-8976148c0bfb

Environment ID

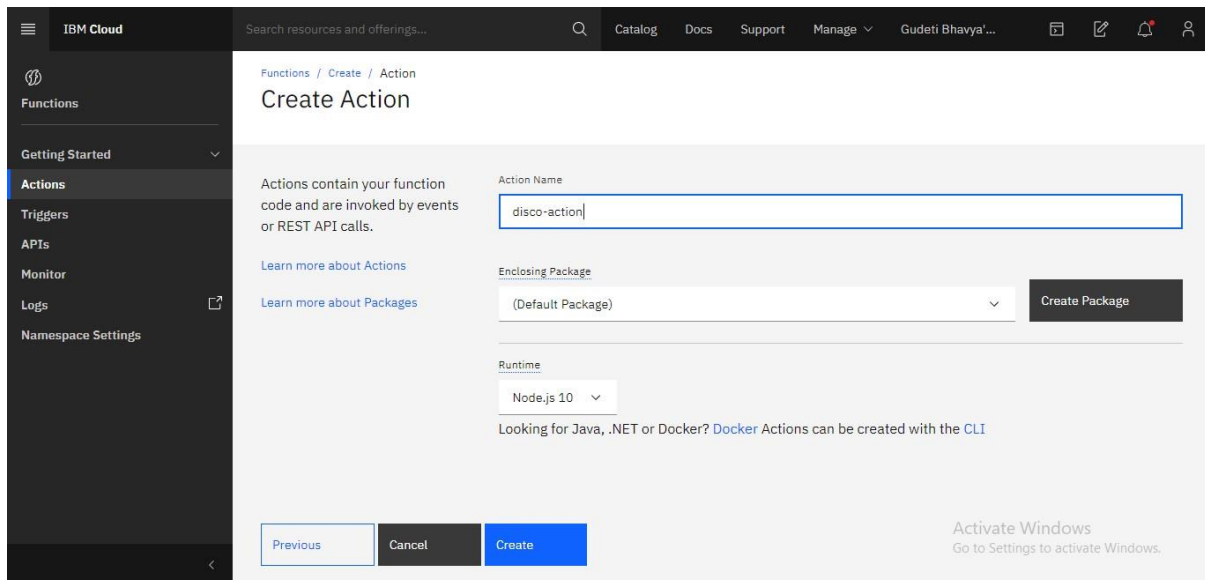
211759ec-3d6c-4d7d-8f72-9e36cc9181f8

Step – 3 : Create IBM Cloud Functions action

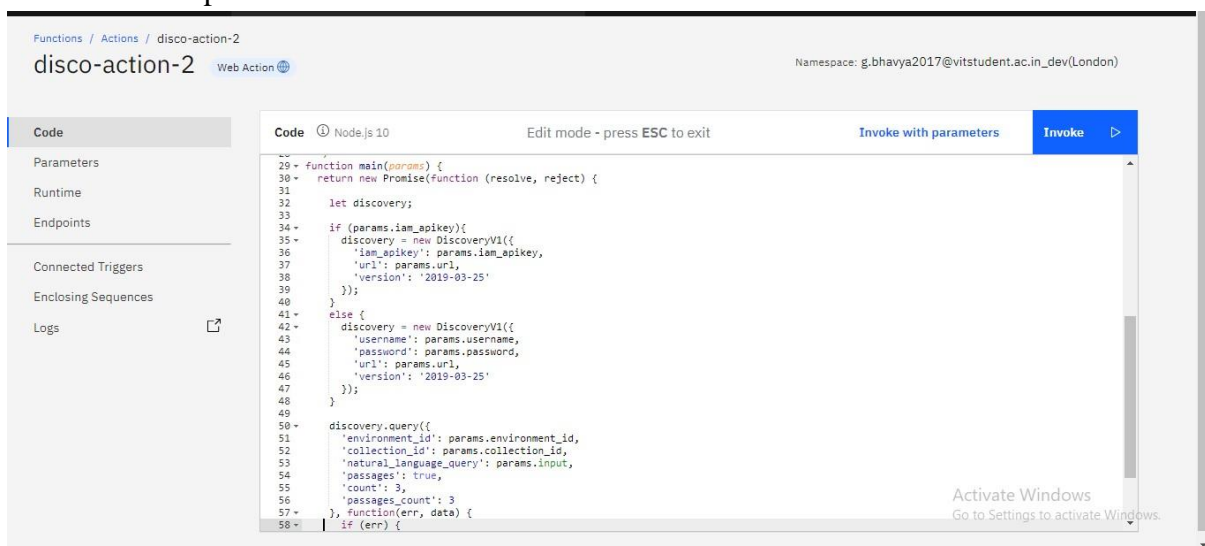
Now let's create the web action that will make queries against our Discovery Collection. Start the IBM Cloud Functions service by selecting **Create Resource** from the IBM Cloud dashboard. Enter functions as the filter then select the **Functions** card.



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 name, keep the default package and select the **Node.js 10** runtime. Click the **Create** button to create the action.



Once the action is created, click on the **Code** tab. Modify the code accordingly to connect to the Discovery service, and to make a query against the collection, then returns the response.



Now select the Parameters tab. Add the following keys : url, environment_id, collection_id, iam_apikey.

Functions / Actions / disco-action-2

disco-action-2

Web Action

Namespace: g.bhavya2017@vitstudent.ac.in_dev(London)

Code

Parameters

Runtime

Endpoints

Connected Triggers

Enclosing Sequences

Logs

Parameter Name	Parameter Value
url	"https://api.eu-gb.discovery.watson.cloud.ibm.com/instances/f905444"
environment_id	"211759ec-3d6c-4d7d-8f72-9e36cc9181f8"
collection_id	"62541ad0-dfac-4341-ad0a-f1ad987ad1ba"
iam_apikey	"yUAgEBBznRqxaaENR-Rvo7r4Ycl1EvRnUgIGLCQz26jI"

Activate Windows
Go to Settings to activate Windows.

Now that the credentials are set, return to the **Code** panel and press the **Invoke** button. Now we should see actual results returned from the Discovery service.

IBM Cloud

Search resources and offerings...

Catalog Docs Support Manage Gudetl Bhavya...

Functions / Actions / disco-action-2

disco-action-2

Web Action

Namespace: g.bhavya2017@vitstudent.ac.in_dev(London)

Code

Parameters

Runtime

Endpoints

Connected Triggers

Enclosing Sequences

Logs

```

1  /**
2   *
3   * @param (object) params
4   * @param (string) params.iam_apikey
5   * @param (string) params.url
6   * @param (string) params.username
7   * @param (string) params.password
8   * @param (string) params.environment_id
9   * @param (string) params.collection_id
10  * @param (string) params.configuration_id
11  * @param (string) params.input
12  *
13  * @return (object)
14  */
15
16
17 const assert = require('assert');
18 const DiscoveryV1 = require('watson-developer-cloud/discovery/v1');
19
20 /**
21 *
22 * main() will be run when you invoke this action
23 *
24 * @param Cloud Functions actions accept a single parameter, which must be
25 *
26 * @return The output of this action, which must be a JSON object.
27 *
28 */
29 function main(params) {
30   return new Promise(function (resolve, reject) {

```

Invoke with parameters

Invoke

Activations

disco-action-2 491 ms 5/19/2020, 21:11:42

Activation ID: ff67d5a98e241b997d5a968e2e1b984

Results:

```

{
  "matching_results": 112,
  "pages": [
    {
      "enriched_text": {
        "categories": [
          {
            "label": "/business and industrial/energy",
            "score": 0.69935
          },
          {
            "label": "/business and industrial/green solutions",
            "score": 0.643933
          },
          {
            "label": "/business and industrial/business operations",
            "score": 0.639224
          }
        ],
        "concepts": [
          {
            "label": "/business and industrial/business operations",
            "score": 0.639224
          }
        ]
      }
    }
  ]
}

```

Activate Windows
Go to Settings to activate Windows.

Now, 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 in future.

IBM Cloud

Search resources and offerings...

Catalog Docs Support Manage Gudetl Bhavya...

Functions / Actions / disco-action-2

disco-action-2

Web Action

☒ Enable as Web Action

Allow your Cloud Functions actions to handle HTTP events. Web Actions allow to control the response data and type by using a set of URL extensions, such as .json or .html. Learn more about [Web Actions](#).
Note: The Web Action URL below requires to return a dict object that contains a body property.

☐ Raw HTTP handling

When enabled your Action receives requests in plain text instead of a JSON body

HTTP Method	Auth	URL
ANY	Public	https://eu-gb.functions.cloud.ibm.com/api/v1/web/g.bhavya2017%40vitstudent.ac.in_dev/default/disco-action-2

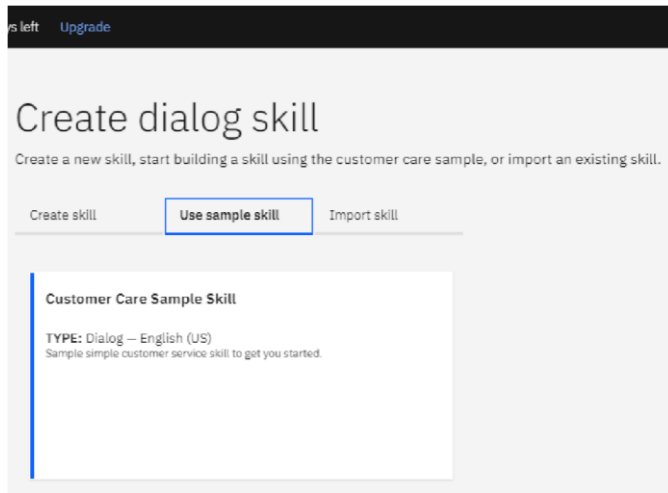
REST API

HTTP Method	Auth	URL
POST	API-KEY	https://eu-gb.functions.cloud.ibm.com/api/v1/namespaces/g.bhavya2017%40vitstudent.ac.in_dev/actions/disco-action-2

Activate Windows
Go to Settings to activate Windows.

Step – 4 : Configure Watson Assistant

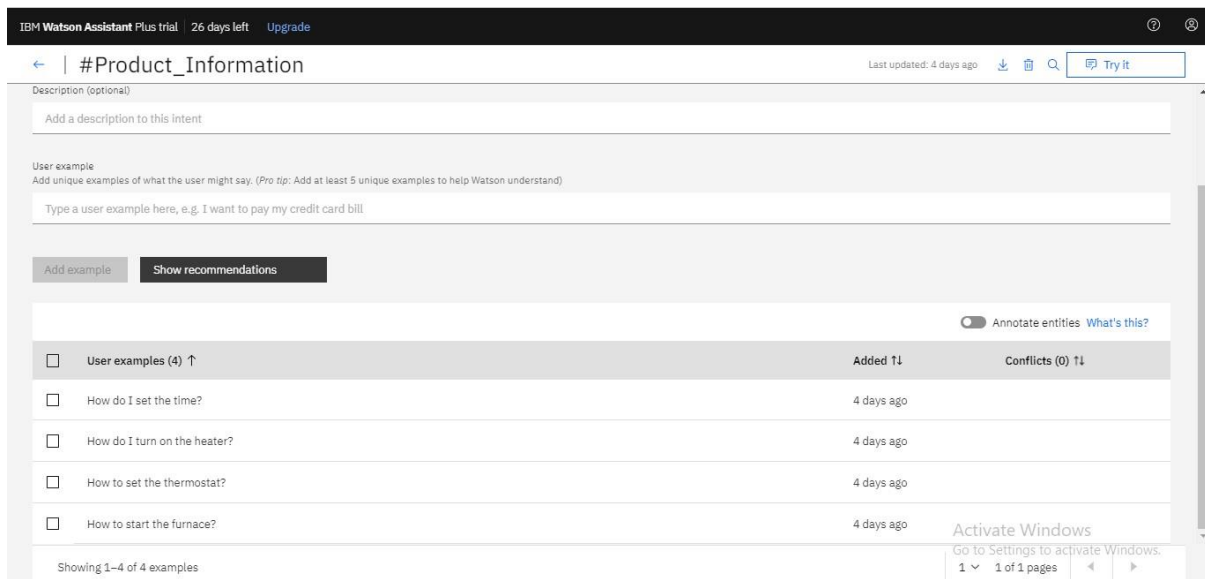
Launch the **Watson Assistant** tool and create a new dialog skill. Select the **Use sample skill** option as the starting point. This dialog skill contains all of the nodes needed to have a typical call centre conversation with a user.



- Add new intent

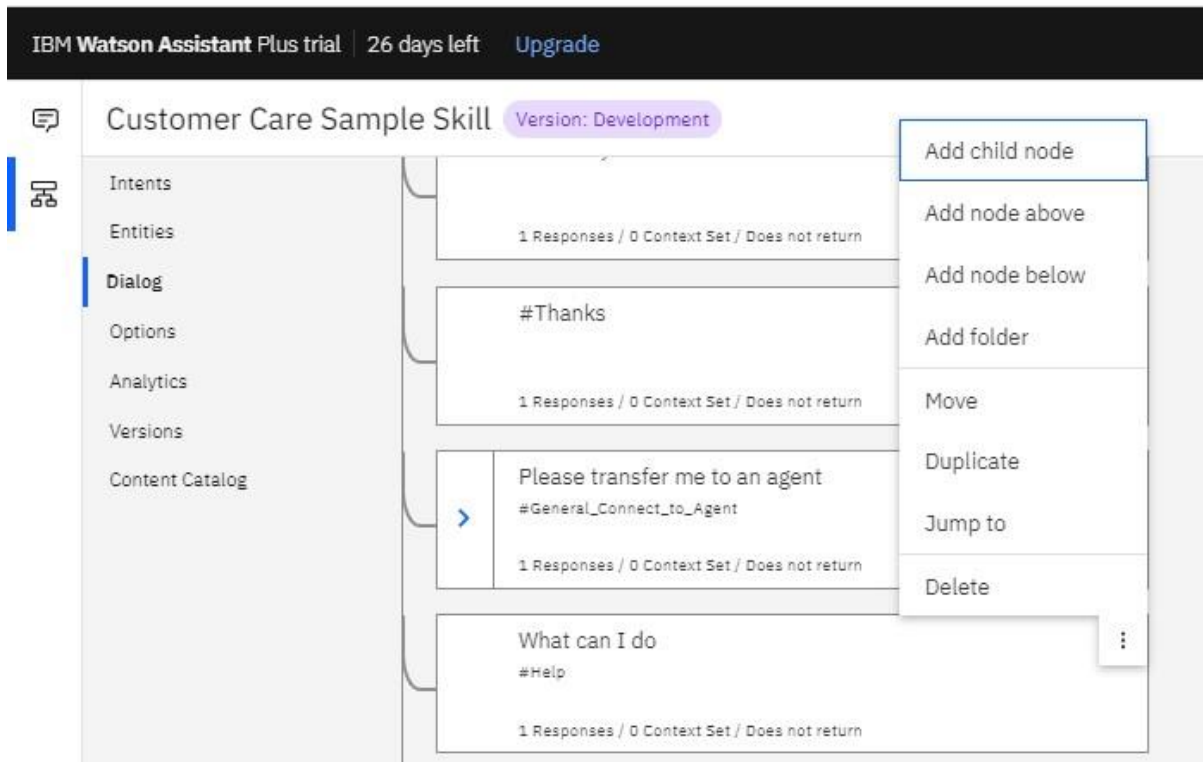
The default customer care dialog does not have a way to deal with any questions involving outside resources, so we will need to create a new intent that can detect when the user is asking about operating the Ecobee thermostat.

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.

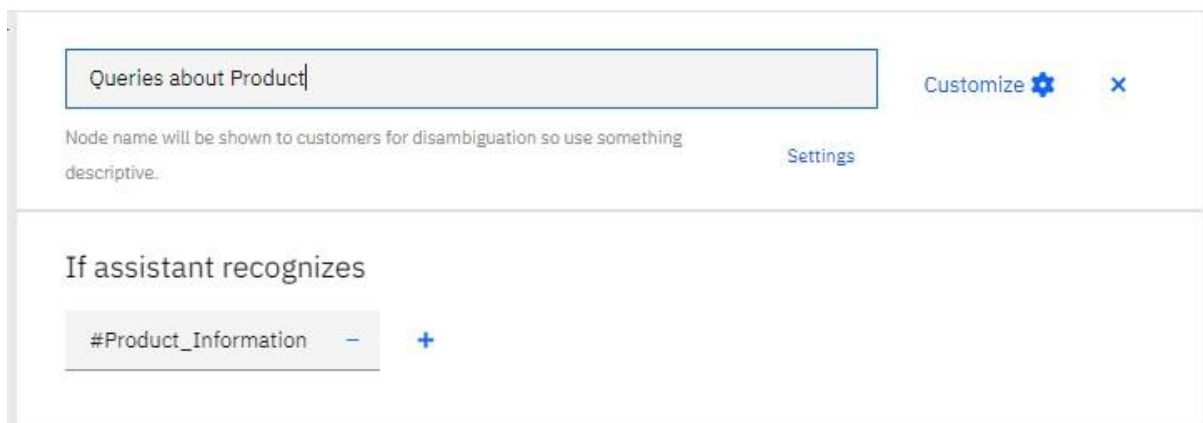


- Create new dialog node

Now we need to add a node. Click on the **Dialog** tab, then search for “**What can I do**” node, and select the **Add node below** option.



Name the node **“Queries about Product”** and assign it our newly created intent. This means that if Watson Assistant recognizes a user input such as **“How do I set the timer?”**, it will direct the conversation to this node.



- Enable webhook from Assistant

Set up access to our WebHook for the IBM Cloud Functions action that we created earlier. Select the **Options** tab. Enter the public URL endpoint for the action. Add **.json** to the end of the URL to specify that the result should be in JSON format.

The screenshot shows the IBM Watson Assistant console interface. On the left is a sidebar with navigation links: Intents, Entities, Dialog, Options, Webhooks (highlighted), Disambiguation, Autocorrection, Irrelevance Detection, System Entities, Analytics, Versions, and Content Catalog. The main area is titled 'Webhooks' and includes a description: 'A webhook is a mechanism that allows you to call out to an external program based on events in your dialog.' Below this is the 'Webhook setup' section, which contains a 'URL' field with the value 'https://eu-gb.functions.cloud.ibm.com/api/v1/web/g.bhavya2017%40vitstu'. There is also a 'Headers' section with a table for 'Header name' and 'Header value', and buttons for 'Add header' and 'Add authorization'. At the bottom, a 'Next step' section provides instructions on how to trigger the webhook from a dialog node.

Customer Care Sample Skill Version: Development

Intents

Entities

Dialog

Options

Webhooks

Disambiguation

Autocorrection

Irrelevance Detection

System Entities

Analytics

Versions

Content Catalog

Webhooks

A webhook is a mechanism that allows you to call out to an external program based on events in your dialog.

Webhook setup

Specify the request URL for an external API you want to be able to invoke from dialog nodes. Watson will call this URL when configured to do so from a dialog node. [Learn more](#)

URL

`https://eu-gb.functions.cloud.ibm.com/api/v1/web/g.bhavya2017%40vitstu`

Headers

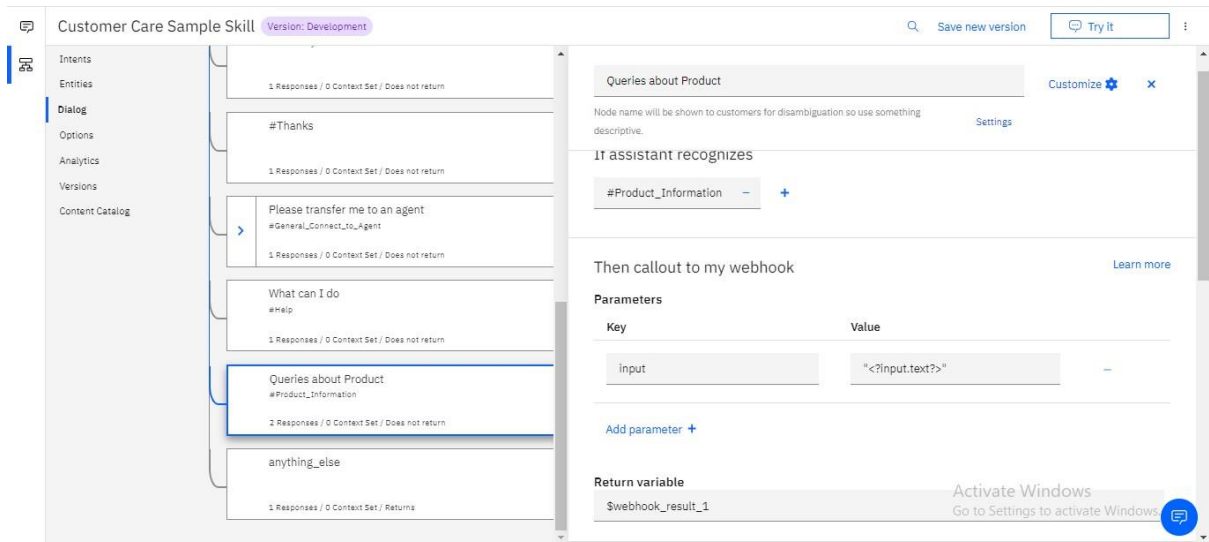
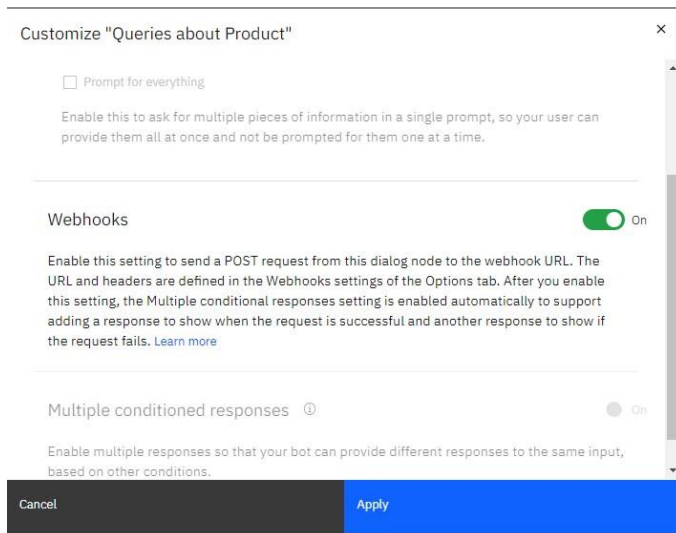
Add HTTP headers for authorization or any other parameters required for invoking the webhook.

Header name	Header value
Add header +	Add authorization +





Next step

To trigger this webhook from an individual dialog node, enable webhooks from the Customize page of the node. [Go to dialog](#)

Now return to the **Dialog** tab and click on the **Queries about Product** 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 we can use to access the result from the Discovery service query. We will also need to pass in the users question via the parameter input. The key needs to be set to the value “<?input.text?>”.

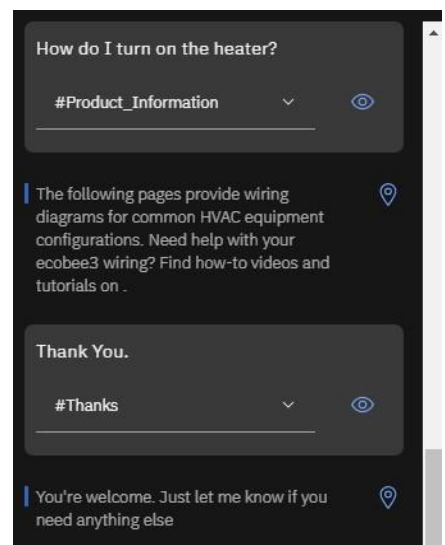
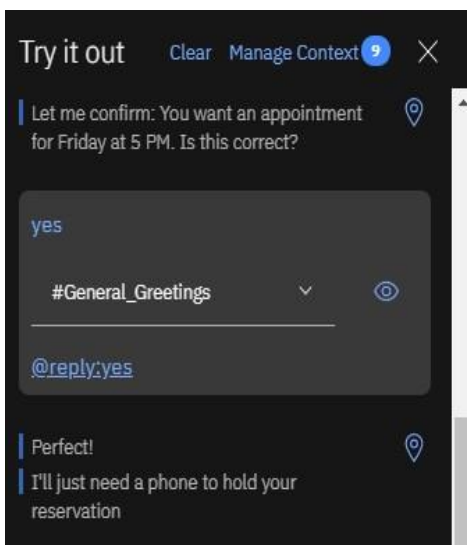
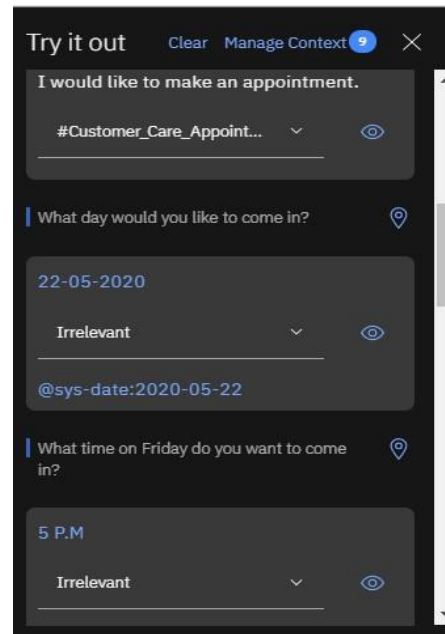
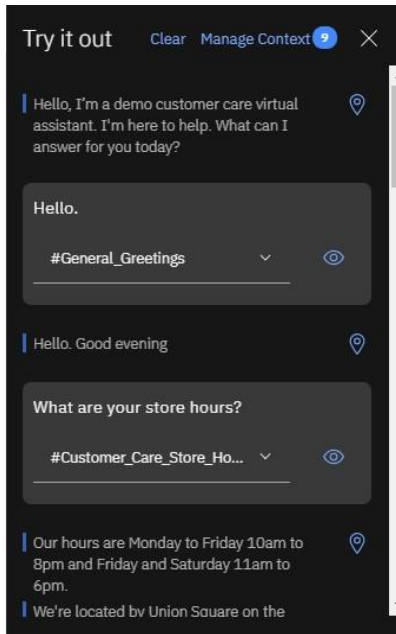


Optionally we can add the following responses to aid in debugging.

Assistant responds	
If assistant recognizes	Respond with
1 \$webhook_result_1	<?\$webhook_result_1.passag  
2 anything_else	Try again later  
Add response +	

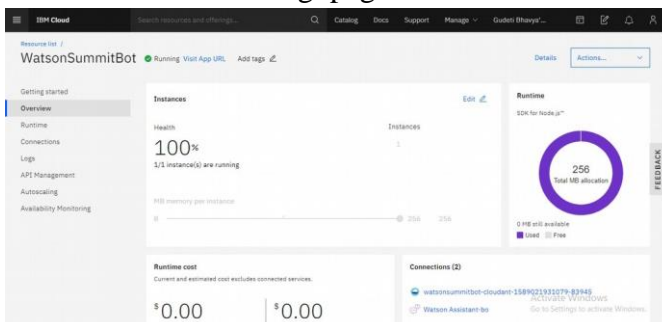
- Test in Assistant Tooling

From the **Dialog** panel, click the **Try it** button. Enter some user input:



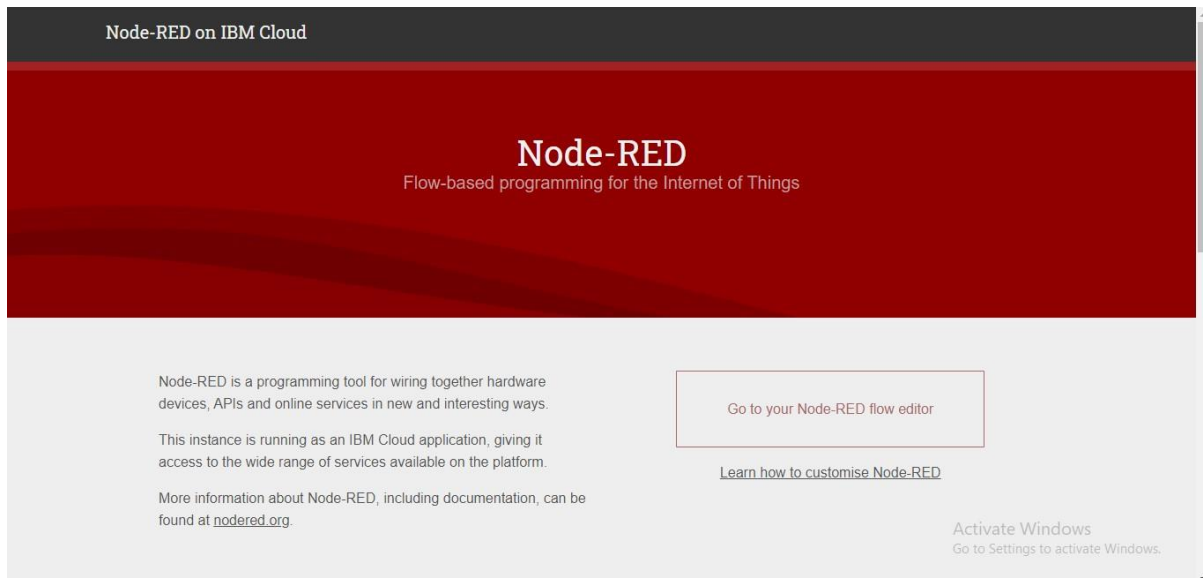
Step – 5 : Create the Node-RED flow

First go to the **IBM Cloud** dashboard and Click on the **Resources** list, Go to **Cloud Foundry apps** and click on the app that we have created earlier. This will direct to the following page. There click on the **Visit App URL**.

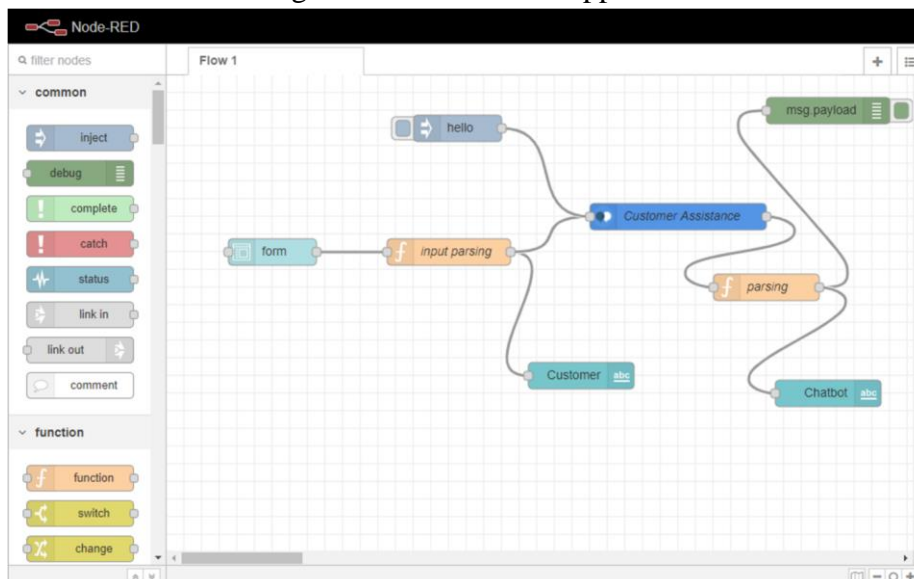


The first time we open the **Node-RED** app, we'll need to configure it and set up security.

1. On the initial screen, click **Next** to continue.
2. Secure the Node-RED editor by providing a **username** and **password**. Click **Next** to continue.
3. The final screen summarizes the options we have made and highlights the environment variables. Click **Finish** to proceed.
4. From here, we can click **Go to your Node-RED flow editor** button to open the editor.



5. Now start creating the flow for this application as shown:



Edit form node

Delete Cancel Done

Properties

Group [customercare] chatbot

Size auto

Label optional label

Label	Name	Type	Required	Rows	Remove
Enter your input	text	Text	<input checked="" type="checkbox"/>		

Edit function node

Delete Cancel Done

Properties

Name input parsing

Function

```

1 msg.payload = msg.payload.text;
2 return msg;

```

Edit inject node

Delete Cancel Done

Properties

Payload hello

Topic

Inject once after 0.1 seconds, then

Repeat none

Name Name

Edit text node

Delete Cancel Done

Properties

Group [customercare] chatbot

Size 6 x 2

Label Customer

Value format {{msg.payload}}

Layout

label value

label value

label value

label value

label value

Name

Edit text node

Delete Cancel Done

Properties

Group [customercare] Group

Size auto

Label Chatbot

Value format {{msg.payload}}

Layout

label value

label value

label value

label value

label value

Name

For **parsing** function write the code as follows :

```
msg.payload.text="";
```

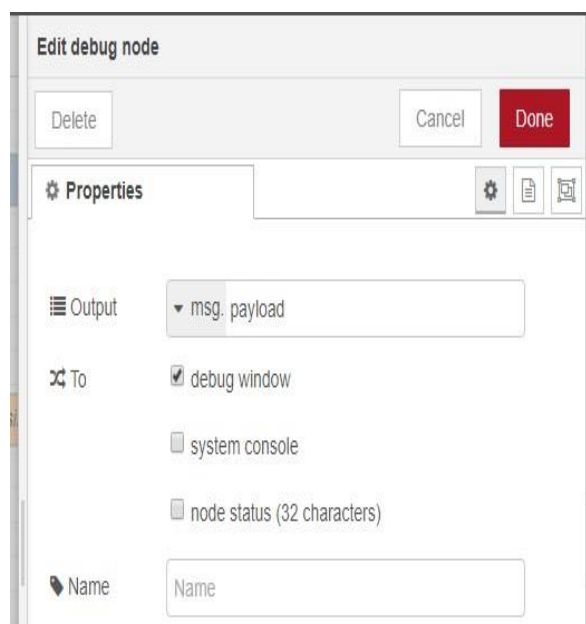
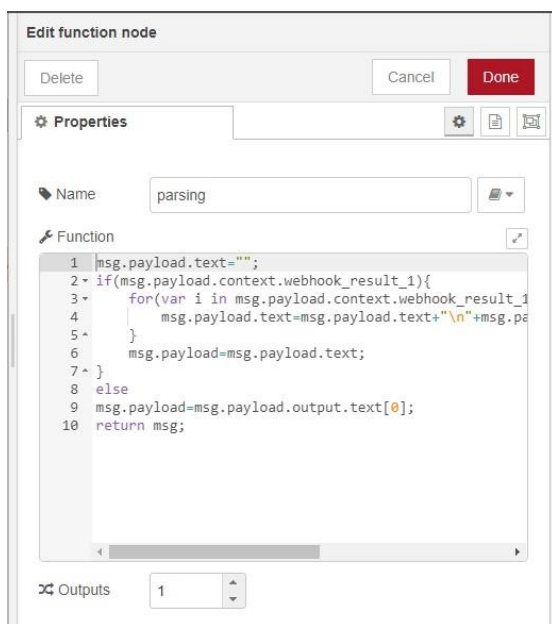
```
if(msg.payload.context.webhook_result_1){      for(var i in
```

```

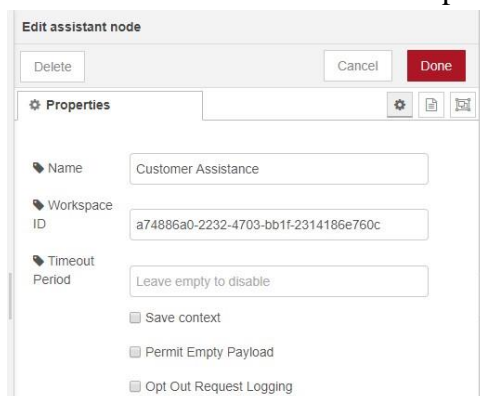
msg.payload.context.webhook_result_1.results){
msg.payload.text=msg.payload.text+"\n"+msg.payload.context.webhook_result_1.results[i].text;
}

msg.payload=msg.payload.text;
}
else
msg.payload=msg.payload.output.text[0];
return msg;

```

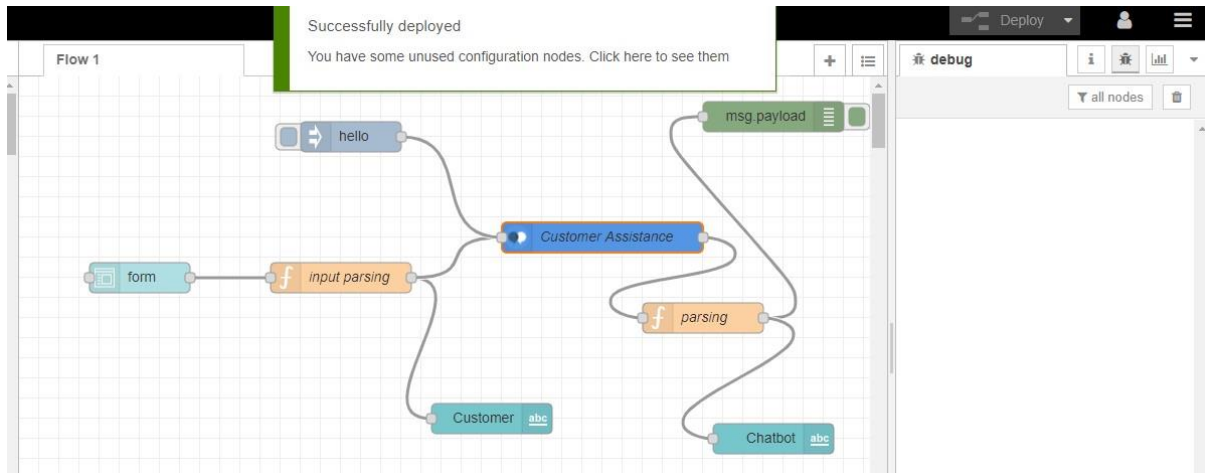


Configure the **Assistant** node with the **Workspace Id** (We get it from **View API details** from Watson Assistant)(**Skill Id**). Make sure that you give the correct credentials. Otherwise it shows an error that “**Resource is not found**”. And make sure to **uncheck the save context** option in Assistance node.

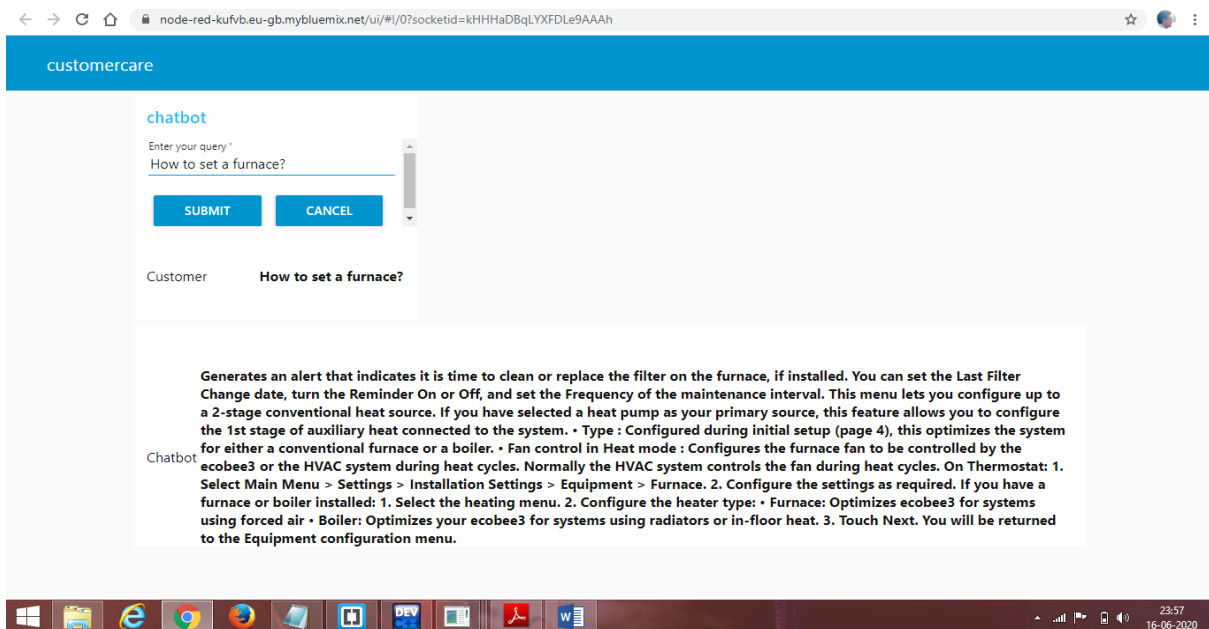


6. RESULT

Now Click on the **Deploy** button. If you configure everything correctly, then you should see the following image saying that “Successfully deployed”.



Now copy the URL till **.net/** and go to the new tab in browser and paste the URL and add **ui** at the last of the URL. You should see the following page. Enter some user inputs and test the bot and check whether it is giving accurate results or not.



7. ADVANTAGES & DISADVANTAGES

Advantages:

Intelligent document understanding solutions combine search and analytics with Machine Learning(ML) to automatically extract relevant information from unstructured data. Businesses can then access critical insights in real-time, saving significant time and resources that would've been required to manually search through massive data. The benefits include:

- Improve compliance and risk management.
- Increase operational efficiencies.
- Enhance business processes through automation.

Introducing smart document understanding (SDU), a powerful feature now available in Watson Discovery, that can give accurate answers faster. SDU distinguishes text elements and extracts the most valuable content such as key paragraphs, while excluding noise like footers and headers and even identifying text in images. SDU allows to tell AI which data in the document matters, immediately making large data set smaller.

Disadvantages:

The problem isn't with search so much as it is with what information is being searched. That is, traditional search engines are great for crawling, indexing and querying the relatively homogenous information that constitutes web sites and online data. However they're less effective at dealing with the masses of heterogeneous structured and unstructured information that businesses store in various on premises and cloud locations. The more diverse, and dispersed an organization's data resources are, the less likely they can be fully managed or exploited with existing search tools.

8. APPLICATIONS

Consider the time savings – not to mention potential revenue and risk reduction opportunities for a financial institution like U.S. Bank. For businesses today, accepting credit and debit card payments has never been easier. Pricing, on the other hand, remains complex involving multiple parties with varying fee structures based on risk, fraud potential and card-specific rewards. Additionally, pricing can be customised based on the type of customers a merchant sells to. As a result, monthly billing statements breaking down costs and fees is confusing, time consuming and often times frustrating for both merchants and incoming sales reps hoping to win new business.

To solve this problem, U.S. Bank's Innovation team partnered with Elavon, piloted and tested a statement analysis solution capable of analysing a prospect billing statement in real-time and generating an optimized pricing proposal. Using IBM Watson Discovery, with the enhanced smart document understanding capability, what once took 10 days now takes 2 minutes. The solution can analyse dense documents, often dozens of pages along with thousands of relevant pricing codes and other details, and parse relevant information for merchants and sales reps quickly and clearly. As a result, both merchants and sales reps can now spend time doing what they do best, serving their customers and building meaningful, valuable relationships.

9. CONCLUSION

This project explained how to use the Smart Document Understanding feature of Watson Discovery to train the dialog to call out to other IBM Watson Services for additional sources of information. We now should have a fundamental understanding of Watson Discovery and some of its advanced features.

10. FUTURE SCOPE

Within IT, few things are ever really finished or settled. That squares with the fact that technologies are tools that, with evolutionary refinement, can be successfully applied to increasing numbers and other types of problems. A notable point to consider about Watson Discovery with SDU is how it can demonstrably benefit both old school processes like sales proposal creation for U.S. Bank and emerging efforts, including document-based machine learning for AI and searching for valuable “small data” assets. It is surprising to see that the organizations find new ways to use IBM’s Watson Discovery with SDU in the months and years ahead.

11. BIBLIOGRAPHY

1. <https://developer.ibm.com/patterns/enhance-customer-helpdesk-with-smart-document-understanding-using-search-skill/>
2. <https://developer.ibm.com/patterns/enhance-customer-help-desk-with-smart-document-understanding/>
3. <https://developer.ibm.com/recipes/tutorials/how-to-create-a-watson-chatbot-on-nodered/>
4. <https://developer.ibm.com/components/node-red/tutorials/how-to-create-a-node-red-starter-application/>
5. <https://www.ibm.com/cloud/get-started>
6. <https://www.ibm.com/watson/products-services>
7. <https://developer.ibm.com/components/watson-assistant/series/learning-path-watson-assistant>
8. <https://cloud.ibm.com/docs/openwhisk?topic=cloud-functions-getting-started>

APPENDIX

Source Code :

NODE-RED.json :

```
[{"id":"898085f0.ac7548","type":"tab","label":"Customer Helpdesk","disabled":false,"info":""},{ "id":"c8012c38.cc23","type":"ui_form","z":"898085f0.ac7548","name":"","label":"","group":"f9306c2c.f51bb","order":1,"width":"0","height":"0","options":[{"label":"Enter your query","value":"text","type":"text","required":true,"rows":null}], "formValue":{"text":""},"payload":"","submit":"submit","cancel":"cancel","topic":"","x":120,"y":200,"wires":[["d9a31711.e6b0a8"]]}, {"id":"d9a31711.e6b0a8","type":"function","z":"898085f0.ac7548","name":"in
```

```

put parsing", "func": "msg.payload = msg.payload.text;\nreturn
msg;", "outputs": 1, "noerr": 0, "x": 320, "y": 200, "wires": [[{"id": "a6d4f157.6381d", "type": "ui_text", "z": "898085f0.ac7548", "group": "f9306c2c.f51bb", "order": 2, "width": "6", "height": "2", "name": "", "label": "Customer", "format": "{ {msg.payload } }", "layout": "row-spread", "x": 470, "y": 340, "wires": []}, {"id": "e121e134.ac52b", "type": "ui_text", "z": "898085f0.ac7548", "group": "4d096e7c.26761", "order": 3, "width": "20", "height": "4", "name": "", "label": "Chatbot", "format": "{ {msg.payload } }", "layout": "row-spread", "x": 750, "y": 360, "wires": []}, {"id": "3aa9f6fb.7ee0ba", "type": "watson-conversation-v1", "z": "898085f0.ac7548", "name": "Customer Assistance", "workspaceid": "f3fda8f4-f0a1-4929-8352-7c13990ca353", "multiuser": false, "context": false, "empty-payload": false, "service-endpoint": "https://api.eu-gb.assistant.watson.cloud.ibm.com/instances/500b8f3d-46e6-49feb8d8-88a4904b643b", "timeout": "", "optout-learning": false, "x": 580, "y": 160, "wires": [{"id": "a8b57ae6.d14c48"}]}, {"id": "a8b57ae6.d14c48", "type": "function", "z": "898085f0.ac7548", "name": "parsing", "func": "msg.payload.text=\\\"\\\";\nif(msg.payload.context.webhook_result_1){\n  for(var i in msg.payload.context.webhook_result_1.results){\n    msg.payload.text=msg.payload.text+\\\"\\n\\\"+msg.payload.context.webhook_result_1.results[i].text;\n  }\n  msg.payload=msg.payload.text;\n}\nelse\nmsg.payload=msg.payload.output.text[0];\nreturn msg;", "outputs": 1, "noerr": 0, "x": 680, "y": 240, "wires": [{"id": "6d2f21fe.fa9a6", "type": "debug", "z": "898085f0.ac7548", "name": "", "active": true, "tosidebar": true, "console": false, "tostatus": false, "complete": "payload", "targetType": "msg", "x": 750, "y": 40, "wires": []}, {"id": "e61cca47.27ca48", "type": "inject", "z": "898085f0.ac7548", "name": "", "topic": "", "payload": "hello", "payloadType": "str", "repeat": "", "crontab": "", "once": false, "onceDelay": 0.1, "x": 330, "y": 60, "wires": [{"id": "3aa9f6fb.7ee0ba"}]}, {"id": "f9306c2c.f51bb", "type": "ui_group", "z": "", "name": "chatbot", "tab": "99423a5b.5d81c8", "order": 1, "disp": true, "width": "6", "collapse": false}, {"id": "4d096e7c.26761", "type": "ui_group", "z": "", "name": "", "tab": "99423a5b.5d81c8", "order": 2, "disp": true, "width": "20", "collapse": false}, {"id": "99423a5b.5d81c8", "type": "ui_tab", "z": "", "name": "customercare", "icon": "dashboard", "disabled": false, "hidden": false}]]

```

Actions.js :

```

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

Assistant Skill :

<https://github.com/SmartPracticeschool/IISPS-INT-2359-Intelligent-Customer-Help-Desk-with-Smart-Documents-Understanding/blob/master/Assistant%20Skill.json>

Project Demonstration Video Link:

<https://www.youtube.com/watch?v=URB5x1WV2C4&t=13s>

Feedback Video Link:

https://www.youtube.com/watch?v=sT_HNR_4vsM&t=11s