

# PROJECT REPORT

NAME :	Preeti Nair [preetinair369@gmail.com]
TITLE :	Intelligent Customer Help Desk With Smart Document Understanding.
CATEGORY :	Artificial Intelligence

Internship at [smartinternz.com](https://smartinternz.com)@2020

# CONTENT

1. **INTRODUCTION**
  - 1.1 Overview
  - 1.2 Purpose
2. **LITERATURE SURVEY**
  - 2.1 Existing problem
  - 2.2 Proposed solution
3. **THEORETICAL ANALYSIS**
  - 3.1 Block Diagram
  - 3.2 Hardware/Software designing
4. **EXPERIMENTAL INVESTIGATION**
5. **FLOWCHART**
6. **RESULT**
7. **ADVANTAGES & DISADVANTAGES**
8. **APPLICATIONS**
9. **CONCLUSION**
10. **FUTURE SCOPE**
11. **BIBLIOGRAPHY**
12. **APPENDIX**
  - A. Source Code

# INTRODUCTION

## 1.1 Overview

The project is focused on creating a customer care chatbot whose dialog will provide a hook that can call out to other Watson services for additional source of information unlike the typical chatbot which rely on predefined responses. In this case, an users manual has been uploaded in Watson Discovery service. This project utilizes different IBM services like Watson Discovery, Cloud Functions, Watson Assistant and Node-Red.

**1. Project Requirements:** Intelligent customer care chatbot with smart document understanding feature given by Watson Discovery which will link to appropriate help context from the user manual. Watson Assistant to store all the dialogs, Node-Red tool to connect all the above mentioned services.

Functional Requirements: Each IBM services utilized have purpose in completing the project. Watson Discovery uses data analysis to take unstructured data and enrich it so you can query it for the information needed. Watson Assistant acts as the Chatbot which gives response to the customer's query. Cloud Functions is used to integrate cloud services. Node-Red is a programming tool which will wire together all the services mentioned above.

**2. Technical Requirements:** This includes Artificial Intelligence, Machine Learning, Python, IBM Watson Services.

**3. Software Requirements:** IBM Watson services like Watson Discovery, Watson Assistant.

**4. Project Deliverables:** This chatbot is created to improve customer experience and to show how various services are involved to get an accurate and desired result.

**5. Project Team:** Individual project work [Preeti Nair].

**6. Project Duration:** 29 days.

## **1.2 Purpose**

Chatbots aren't just for customer. They can provide useful support throughout a business, including service desk. From organisation to simplification to satisfaction, there are many purposes of creating chatbot with smarter service desk.

With feature like Smart Document Understanding (SDU), the customer won't have to undergo the tedious procedure of connecting with customer care representative. Instead the answer of his query will be displayed through the chatbot. The solution provided will be accurate and handy as it is straight from a owner's manual which has been pre-loaded in Watson Discovery. It can direct the user to relevant available content. By taking in

keywords and understanding syntax, bots are then able to suggest useful help resources.

The response will be lightening fast and the users can implement on the advice provided by the bot and get on with their day.

To summarize, the purpose of this chatbot is to enhance customer experience by integrating services provided by IBM like Watson Discovery, Watson Assistant, Node-Red and Cloud Functions.

# LITERATURE SURVEY

## 2.1 Existing Problem

A standard chatbot will provide solution for basic queries like "location of store" or "when does the store open". But when the query is something related to the operation of a device, instead of displaying the solution it will direct the customer to a customer representative.

Due to this, the customer has to wait until the representative gets back to them. Sometimes they don't provide useful solution to the query.

Another problem is these chatbots have limited responses for customers.

This overall deteriorates the customer experience.

## 2.1 Proposed Solution

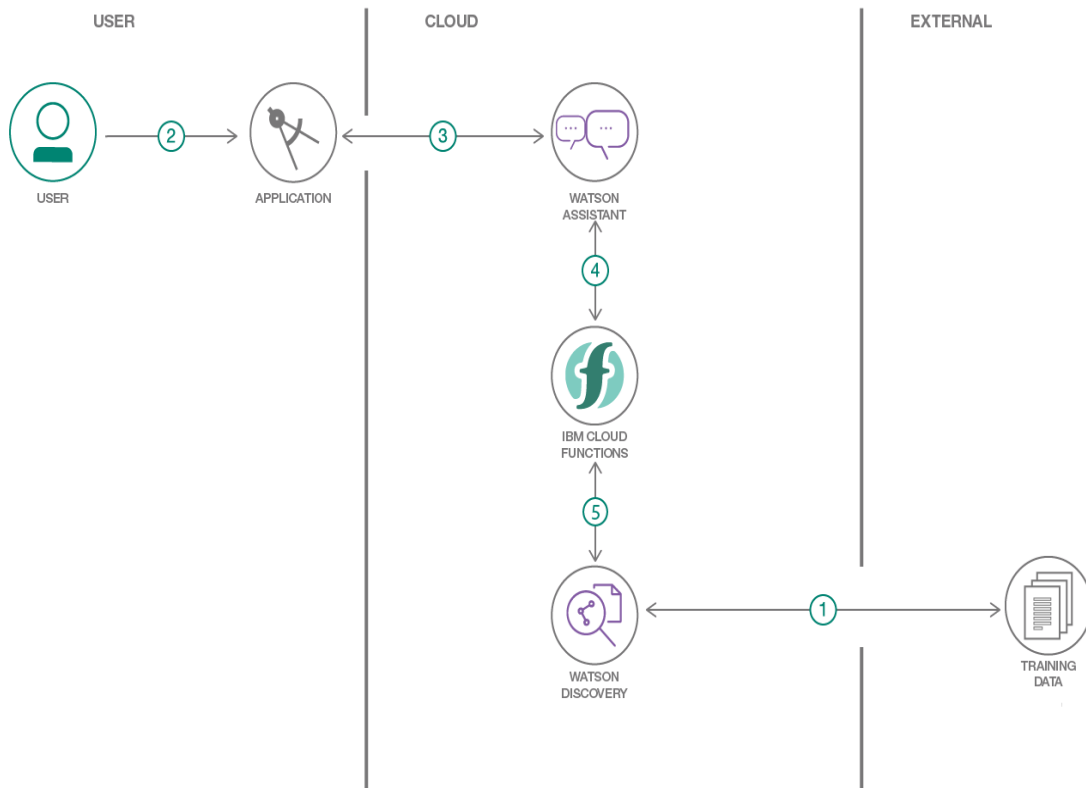
The proposed solution is to create a chatbot that will display the appropriate solution instead of involving any human interaction.

If the customer question 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 owners manual. So now, instead of "Would you like to speak to a customer representative?" it will return relevant sections of the owners manual to help solve our customers' problems.

In this way the customer won't have to undergo the tedious process of connecting with the representative and the customer will get lightning fast response. This will thereby enhance customer service experience.

# THEORETICAL ANALYSIS

## 3.1 BLOCK DIAGRAM



- **Step 1** : The document in this case the owner's manual is indexed and annotated by the Smart Document Understanding feature of Watson Discovery.
- **Step 2** : The user interacts with the app UI. The frontend server is the chatbot which engages with the customer with small talk.
- **Step 3** : The interaction is coordinated by Watson Assistant service.
- **Step 4** : When a technical query comes into picture, the assistant invokes the Cloud Functions action.
- **Step 5** : The action then queries the Discovery and returns with the result relevant to the query.

### **3.2 Hardware/Software Designing**

The softwares utilized are Watson Discovery, Watson Assistant and Node-Red.

**1] Watson Discovery** : Discovery makes it possible to rapidly build cognitive, cloud-based exploration applications that unlock actionable insights hidden in unstructured data. It applies latest breakthroughs in machine learning, including natural language processing capabilities, and is easily trained on the language of your domain. It breaks open the data silos and retrieves specific answers to user's questions while analysing trends and relationships buried in the uploaded data. With Smart Document Understanding (SDU) feature you can train IBM Watson Discovery to extract custom fields in your documents.

**2] Watson Assistant** : It is conversational AI platform that provides customer fast, straightforward and accurate answers to their question, across any application, device or channel. By addressing common customer inquiries, Watson Assistant reduces the cost of customer interactions.

It uses Watson AI machine learning (ML) and natural language understanding (NLU). The user input is received by the assistant and routes it to a dialog skill. The dialog skill interprets the input further, then directs the flow of the conversation. The dialog gathers any information needed to respond on user's behalf.

**3] Node-Red** : It is a prototyping tool that builds applications easily. Applications are developed by building data flows through a series of connected nodes. Intricate applications can be built that allows Watson services to interact with range of capabilities and services exposed as Node-Red nodes.

It helps in wiring together hardware devices, APIs and online services without writing any code. It provides web based flow editor, which can be used to connect these services and also create UI.

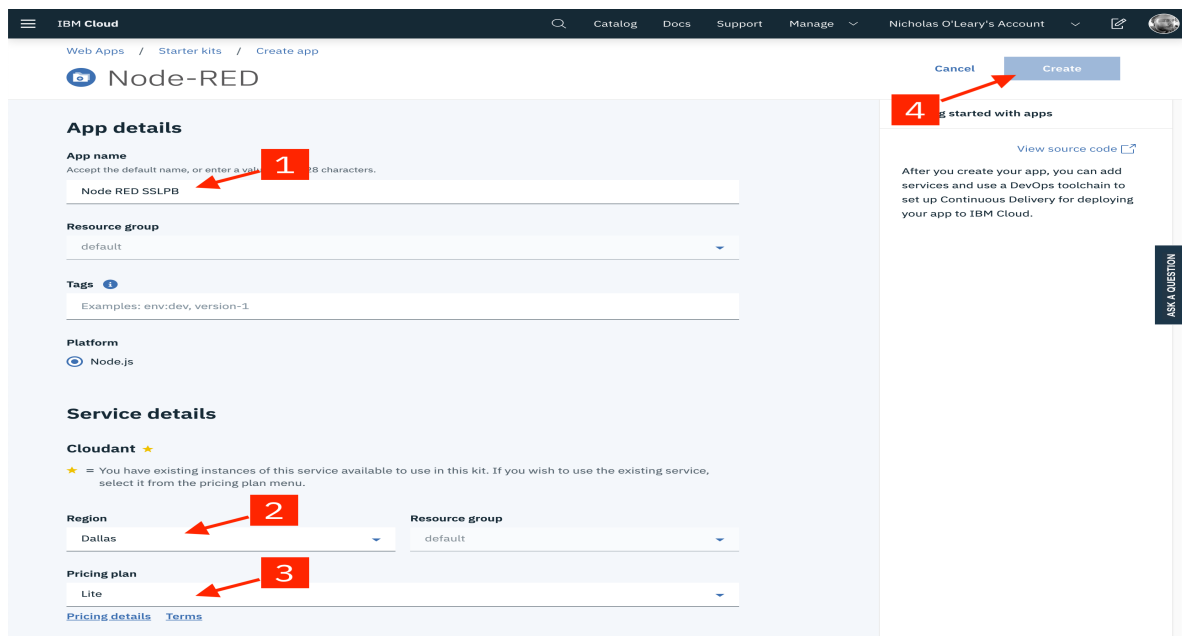
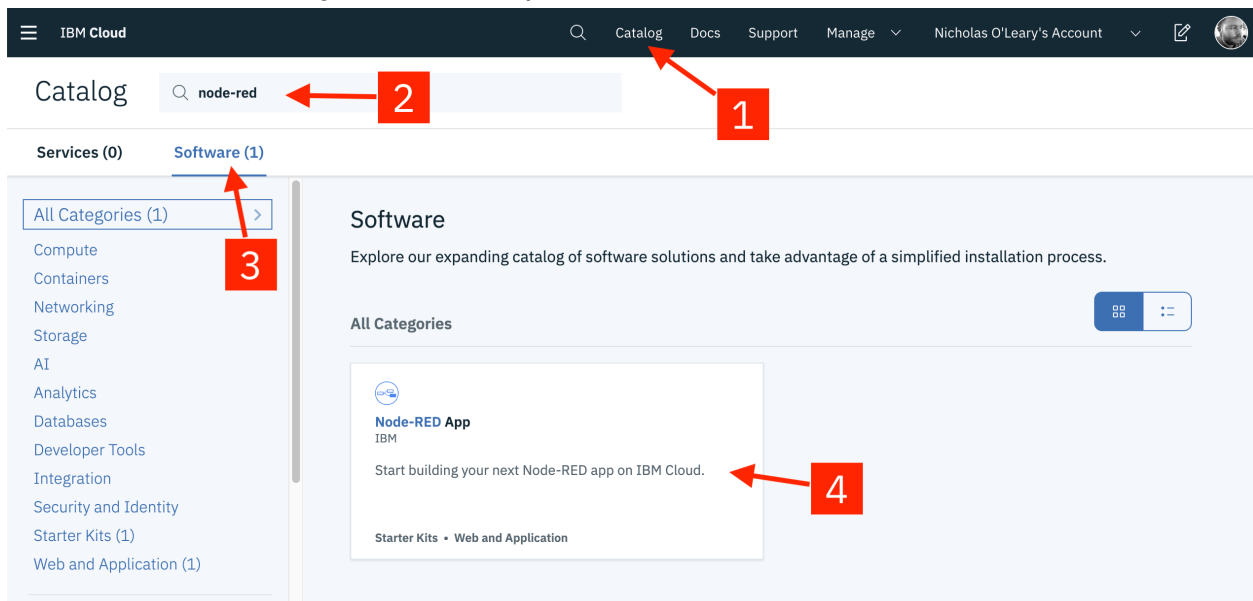


# EXPERIMENTAL INVESTIGATIONS

Following services are to be created :

## 4.1 Node-Red Application

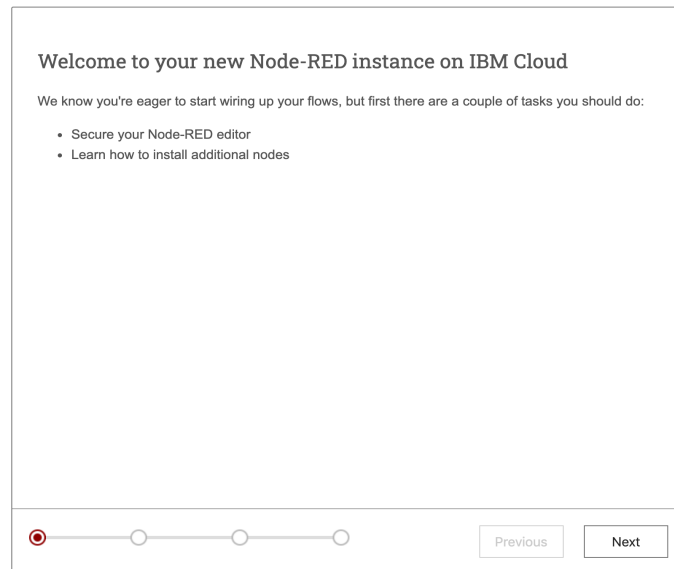
1. Search for Node-Red in catalog and create an application by filling in details like the cloudant region and a unique name.



2. Setup Continuous Delivery feature to deploy the application created into the Cloud foundry apps in IBM Cloud.

After deploying stage is completed, the application is now running.

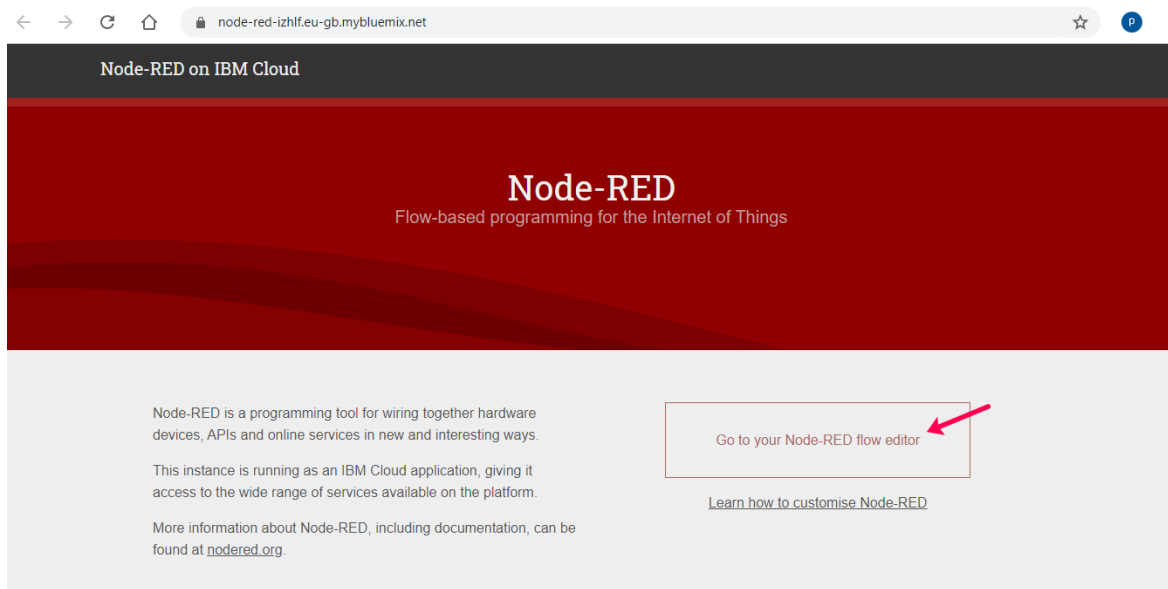
3. In the details page, click on [Visit App URL](#) to configure it and set up security.



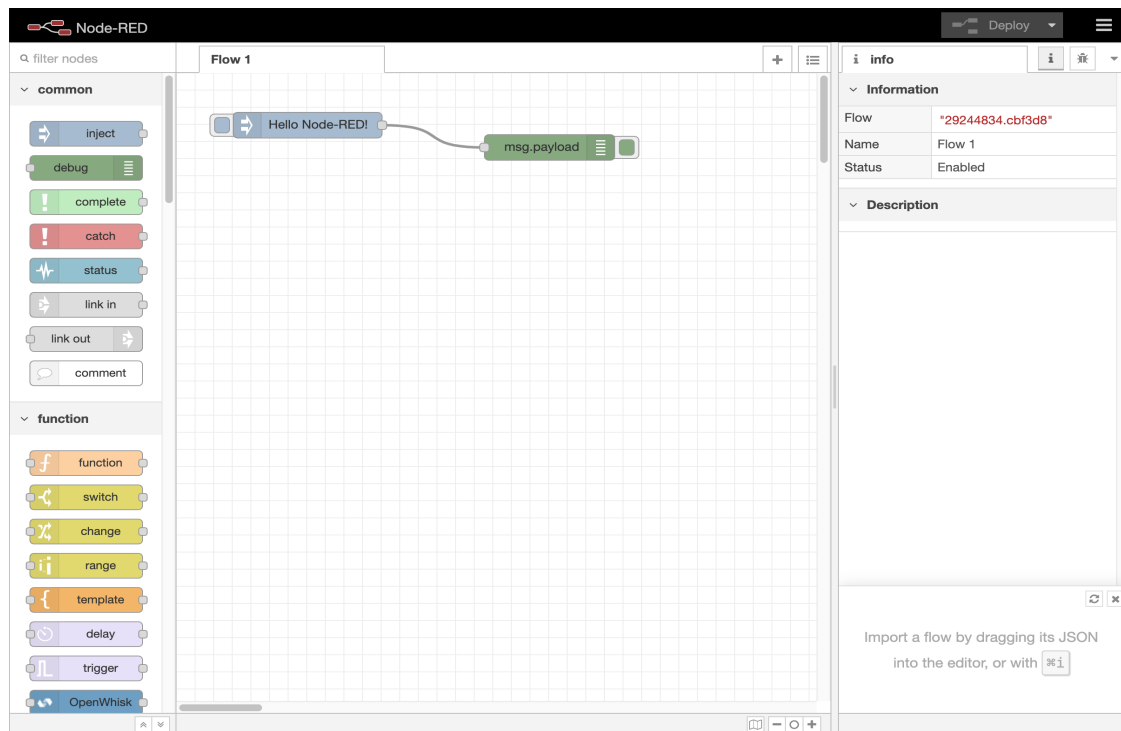
4. Click on Next. Now it will ask for username and password for security reasons. Enter them.

5. The final screen summarizes the options you've made and highlights the environment variables you can use to change the options in the future. Click Finish to proceed.

6. Node-RED will save your changes and then load the main application. From here you can click the Go to your Node-RED flow editor button to open the editor

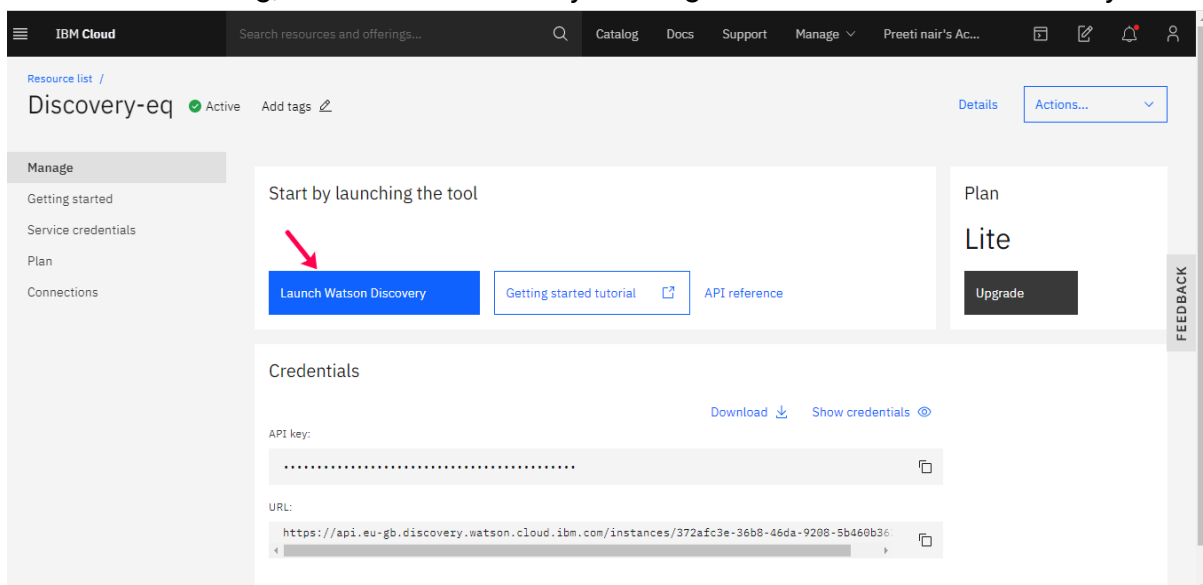


7. This is the place where you will create flow to wire all the services utilized to function the chatbot.

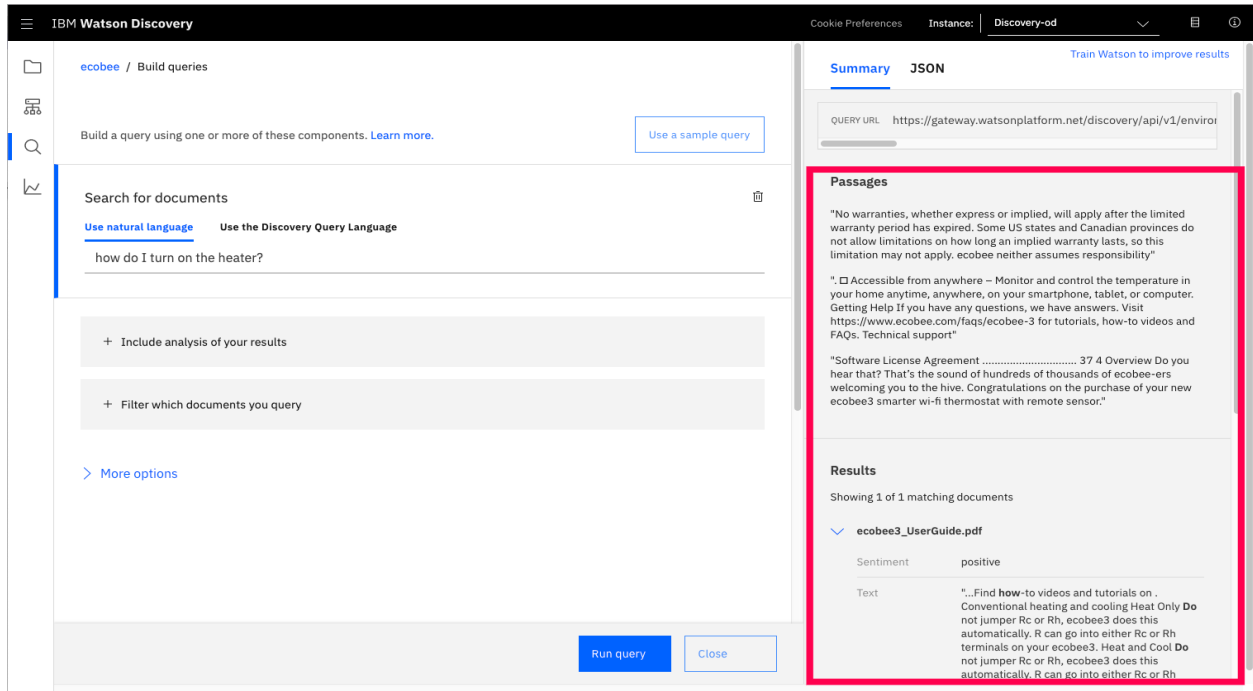


## 4.2 Watson Discovery

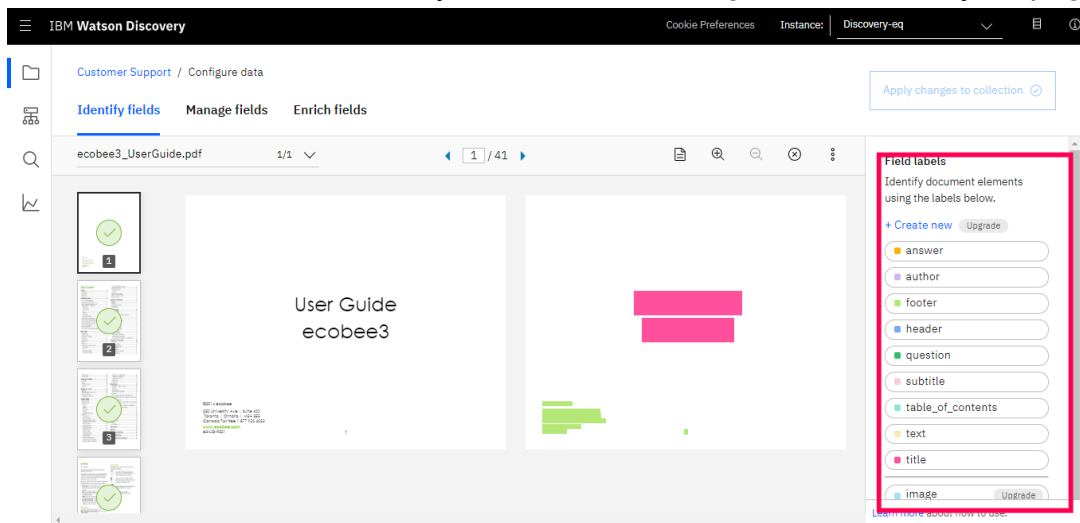
1. Search for Discovery from Catalog and create an instance of the service.
2. After creating, launch the service by clicking on Launch Watson Discovery.



3. Upload the ecobee3 owner's manual by clicking on Upload your own data.
4. Now before setting up the SDU feature, try few queries. Click on the Build your own query.
5. When ask a query, it will display you many sections from the manual. To improve query result, setup SDU feature.

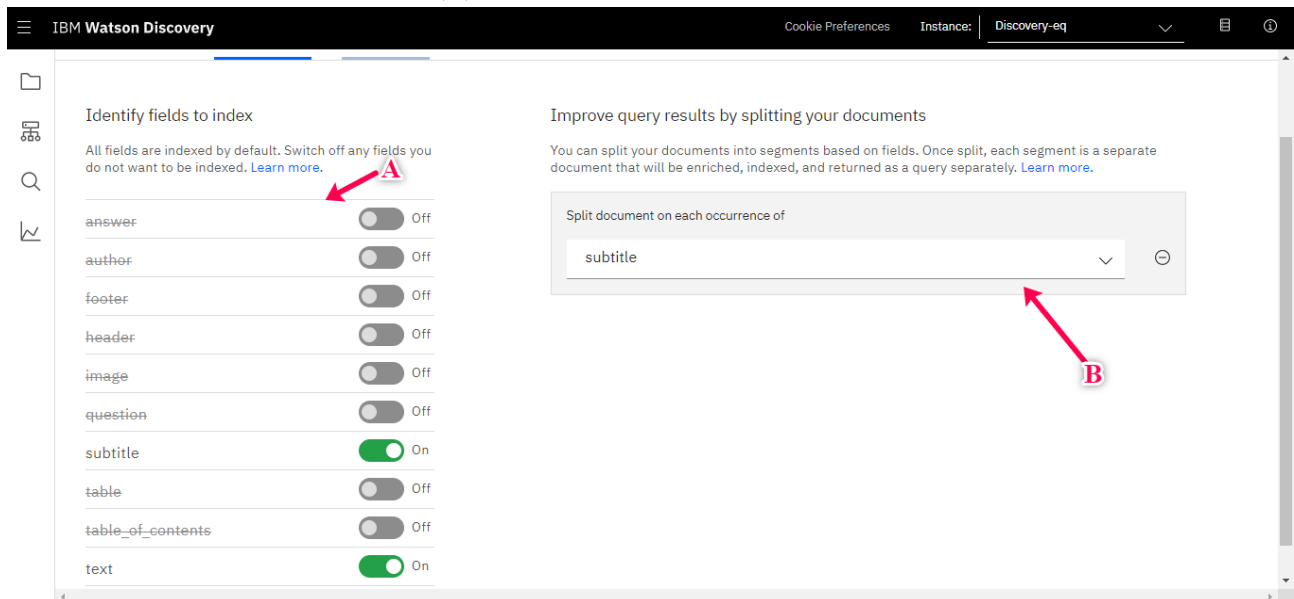


6. Go to Configure data. You will see SDU layout. There are three fields : **Identify fields**, **Manage fields** and **Enrich fields**.
7. In Identify field, here you will identify the elements in the document by labelling them as **title** , **subtitle** or **text** which you will see on the right side of the layout page.



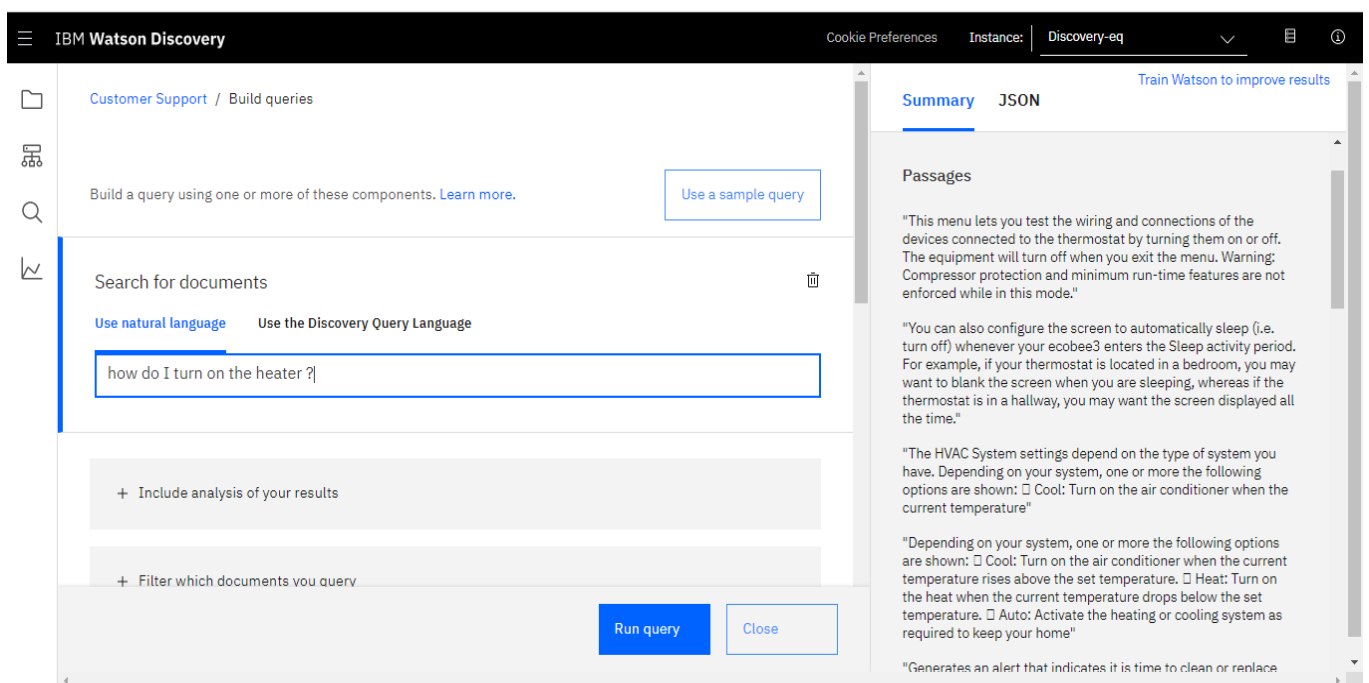
8. Assign relevant labels for 10-11 pages and click on Submit Page accordingly. After few pages, Discovery starts to recognize the element automatically. This means the service is trained.

9. Now go to Manage Fields tab, select the items you want to be indexed (A). After that, split the document by subtitle (B). Click the **Apply to changes button**.



9. It will ask you to upload the document, upload it. You will see all the annotations applied to your index.

10. Again click on **Build your own query**. Ask the same query. This time you will see the most relevant answer from the manual being displayed.



## 4.3 Cloud Functions

1. Search for Functions in catalog and click on Functions.
2. Opening Functions, in the left tab click on **Actions** (A) and select create (B).

The screenshot shows the IBM Cloud Functions console. On the left sidebar, the 'Actions' tab is highlighted with a red arrow labeled 'A'. In the main content area, the 'Create' button is highlighted with a red arrow labeled 'B'. The console displays a table of actions under the 'Default Package'.

Name	Runtime	Web Action	Memory	Timeout
disco-action	Node.js 10	Enabled	256 MB	60 s

Items per page: 10 | 1-1 of 1 items | 1 of 1 pages

3. Once the action is being created, go to code tab. Paste the code which will revoke the Discovery service.

The screenshot shows the IBM Cloud Functions console with the 'disco-action' code tab selected. The code is a JavaScript function that uses the 'watson-developer-cloud/discovery/v1' API to revoke the Discovery service. The code is as follows:

```
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- const assert = require('assert');
17- const DiscoveryV1 = require('watson-developer-cloud/discovery/v1');
18- /**
19-  *
20-  * main() will be run when you invoke this action
21-  *
22-  * @param Cloud Functions actions accept a single parameter, which must be a JSON object.
23-  *
24-  * @return The output of this action, which must be a JSON object.
25-  */
26-
27- function main(params) {
28-   return new Promise(function (resolve, reject) {
29-     let discovery;
30-     if (params.iam_apikey) {
```

4. Before clicking on **Invoke** button, click on Parameters tab and add all necessary

parameters of Discovery service which you will find on following pages.

IBM Cloud Search resources and offerings... Catalog Docs Support Manage Preeti nair's Ac...

Functions / Actions / disco-action

disco-action Web Action Namespace: nairpreeti15@gmail.com\_dev(London)

Code Parameters Runtime Endpoints Connected Triggers Enclosing Sequences Logs

Parameters

Parameter Name	Parameter Value
url	"https://api.eu-gb.discovery.watson.cloud.ibm.com/instances/372afc3"
environment_id	"9db796ff-6b909-2820a40da900-e2082a2f0014"
collection_id	"be24edf2-6b909-2820a40da900-abe473d73af7"
iam_apikey	"vkSyEriKl_Uk...TXNrCdU6I4CC4Gnnlg"

Add Parameter

IBM Watson Discovery Cookie Preferences Instance: Discovery-eq

Customer Support Overview Errors and warnings (127) Search settings

127 documents 0 documents failed View details Created on 5/14/2020 1:32:09 pm E Last updated 5/14/2020 1:32:09 pm E

Identified 3 fields from your data: subtitle, text

Added 4 enrichments to your data: Entity Extraction (0.3°C (4) | 0.5°F (4) | 10 °F (4) | 900 seconds (4) | 20 min (3))

Top people related to /technology and computing/operating systems

Collection ID: be24edf2-6b909-2820a40da900-abe473d73af7

Configuration ID: bbc17804-6b909-2820a40da900-ff1d040049ac

Environment ID: 9db796ff-6b909-2820a40da900-e2082a2f0014

Resource list / Discovery-eq Active Add tags Details Actions...

Manage Getting started Service credentials Plan Connections

You can generate a new set of credentials for cases where you want to manually connect an app or external consumer to an IBM Cloud™ service. Learn more

Search credentials... New credential +

Key name	Date created
Auto-generated service credentials	MAY 14, 2020 - 11:01:12 PM

```
{
  "apikey": "vkSyEriKl_Uk...TXNrCdU6I4CC4Gnnlg",
  "iam_apikey_description": "Auto-generated for key bb0502b6-b42b-48eb-b85d-ed1685e76e28",
  "iam_apikey_name": "Auto-generated service credentials",
  "iam_role_crn": "crn:v1:bluemix:public:iam:::serviceRole:Manager",
  "iam_serviceid_crn": "crn:v1:bluemix:public:iam-identity::a/f2554465f6c8423e818f60065e452137::serviceid:ServiceId-1a2bdc48-98bf-4f97-b909-2820a40da900",
  "url": "https://api.eu-gb.discovery.watson.cloud.ibm.com/instances/372afc3"
}
```

FEEDBACK

5. After adding parameters, go back to code page and click on **Invoke**. You will see the actual results returned from the Discovery service.

The screenshot shows the Cloud Functions console for the namespace 'nairpreeti15@gmail.com\_dev(London)'. The 'Code' tab is active, displaying a Node.js function named 'main' that uses the 'watson-developer-cloud/discovery/v1' API. The function is annotated with parameters like 'params', 'params.iam\_apikey', 'params.url', 'params.username', 'params.password', 'params.environment\_id', 'params.collection\_id', 'params.configuration\_id', and 'params.input'. The 'Invoke' button is highlighted. The 'Activations' tab shows a successful invocation of 'disco-action' with a duration of 1099 ms, timestamped 5/21/2020, 23:54:12. The activation ID is 'e8a38f7502424616a38f7502426616eb'. The results are a JSON object containing 'matching\_results' (127), 'passages' (empty array), and 'results' (array of enriched text objects with categories and labels like 'business and industrial/energy/electricity', 'technology and computing/operating systems', and 'technology and computing/hardware/computer components').

6. In Cloud Functions, click on Endpoints tab, and select **Enable as Web Action**. This will create a url which will be used to setup Webhook in Watson Assistant.

The screenshot shows the 'disco-action' function in the Cloud Functions console, with the 'Endpoints' tab selected. The function is configured as a 'Web Action'. The 'Enable as Web Action' checkbox is checked. The 'Raw HTTP handling' checkbox is unchecked. The 'HTTP Method' is set to 'ANY', the 'Auth' is 'Public', and the 'URL' is 'https://eu-gb.functions.cloud.ibm.com/api/v1/web/nairpreeti15@gmail.com\_dev/default/disco-action'. The 'Connected Triggers' section is empty.

## 4.4 Watson Assistant

1. Create Watson Assistant service in similar fashion as Watson Discovery. Launch the service.
2. When you open the service, you will see a dialog skill named Customer Care skill automatically provided by the service. Either choose that and remodify it or



create new skill.

The screenshot shows the 'Skills' management page in the IBM Watson Assistant console. On the left, there's a sidebar with icons for 'Skills' and 'Channels'. The main area is titled 'Skills' and contains the text: 'Skills contain the training to respond to your customer queries. Add skills to your assistant and then deploy to your channels.' Below this is a blue 'Create skill' button. A card for 'My first skill' is displayed, showing its type as 'Dialog - English (US)', creation and update timestamps, and a link to 'My first assistant' under 'LINKED ASSISTANTS (1)'.

3. Firstly create intent which we will detect if the customer is asking about the operation of the product. Given down below is an intent about Product\_details. Provide few sample examples of queries for the assistant to detect.

The screenshot shows the 'User example' section for the '#Product\_details' intent. At the top, there's a header with a back arrow, the intent name '#Product\_details', and a 'Try it' button. Below the header, there's a text input field with the placeholder 'Type a user example here, e.g. I want to pay my credit card bill'. Below the input field are two buttons: 'Add example' and 'Show recommendations'. At the bottom, there's a table of user examples.

<input type="checkbox"/>	User examples (10) ↑	Added ↑↓
<input type="checkbox"/>	customize thermostat	5 days ago
<input type="checkbox"/>	how to adjust screen brightness	5 days ago
<input type="checkbox"/>	How to adjust the temperature ?	7 days ago

Similarly you can give intents for location, landmarks or timings.

				Create intent +
<input type="checkbox"/>	Intents (7) ↑	Description	Modified ↑↓	Examples ↑↓
<input type="checkbox"/>	#greetings		3 days ago	6
<input type="checkbox"/>	#Holidays		7 days ago	3
<input type="checkbox"/>	#Landmark		7 days ago	1
<input type="checkbox"/>	#location		5 days ago	4
<input type="checkbox"/>	#Product_details		2 days ago	10
<input type="checkbox"/>	#store_timings		5 days ago	5
<input type="checkbox"/>	#Thanks		5 days ago	4
Showing 1–7 of 7 intents				1 ▾ 1 of 1 pages

4. Click on Options. Here you will setup Webhook. Use the url from the Web action of Cloud functions in Endpoints tab.

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/nairpreeti15%40gmail.c

### Headers

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

5. Move onto Dialog tab. Here, add nodes to handle intents.
6. Create a node for Product\_details (1) and accordingly assign its intent (2).
7. Enter syntax (3) which will pass on the query via **input** parameter to discovery.

Product\_details

Node name will be shown to customers for disambiguation so use something descriptive.

Settings

If assistant recognizes

#Product\_details

Then callout to my webhook

Learn more

Parameters

Key	Value
input	"<?input.text?>"

8. Click on Customize and enable the Webhook.

Customize "Product\_details"

☐ Prompt for everything

Enable this to ask for multiple pieces of information in a single prompt, so your user can provide them all at once and not be prompted for them one at a time.

Webhooks

On





Enable this setting to send a POST request from this dialog node to the webhook URL. The URL and headers are defined in the Webhooks settings of the Options tab. After you enable this setting, the Multiple conditional responses setting is enabled automatically to support adding a response to show when the request is successful and another response to show if the request fails. [Learn more](#)

Webhook URL Your webhook URL is configured. Options

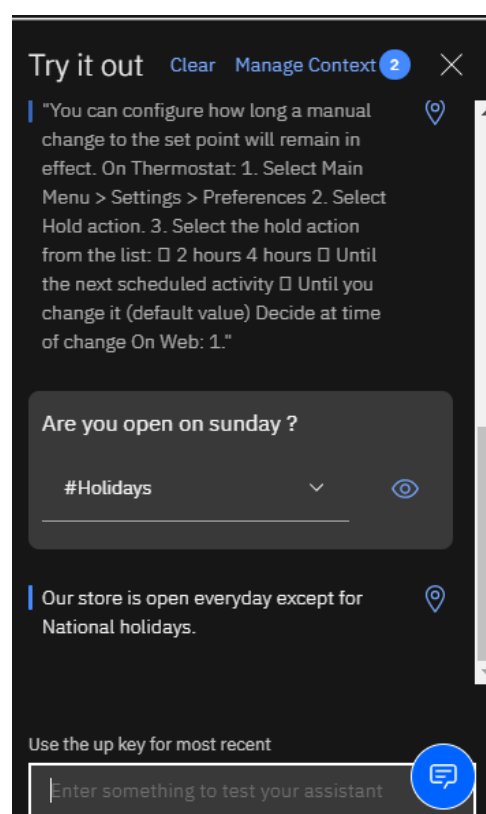
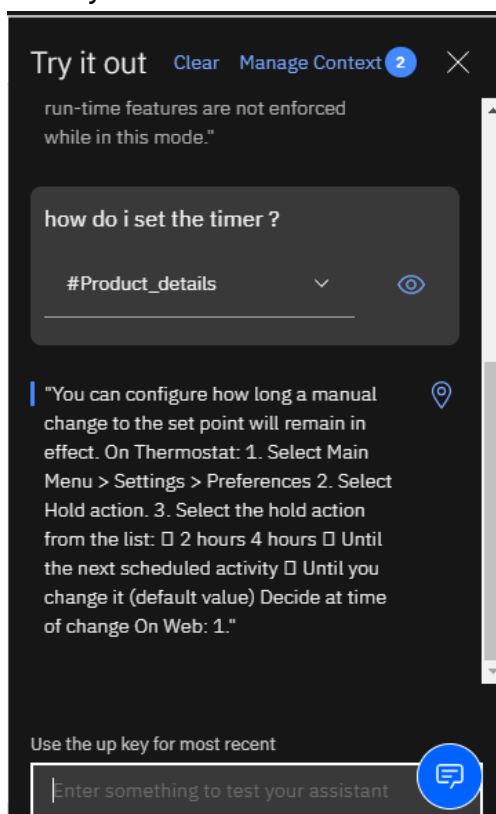
Cancel Apply

9. Now add the responses to aid in debugging.

Assistant responds

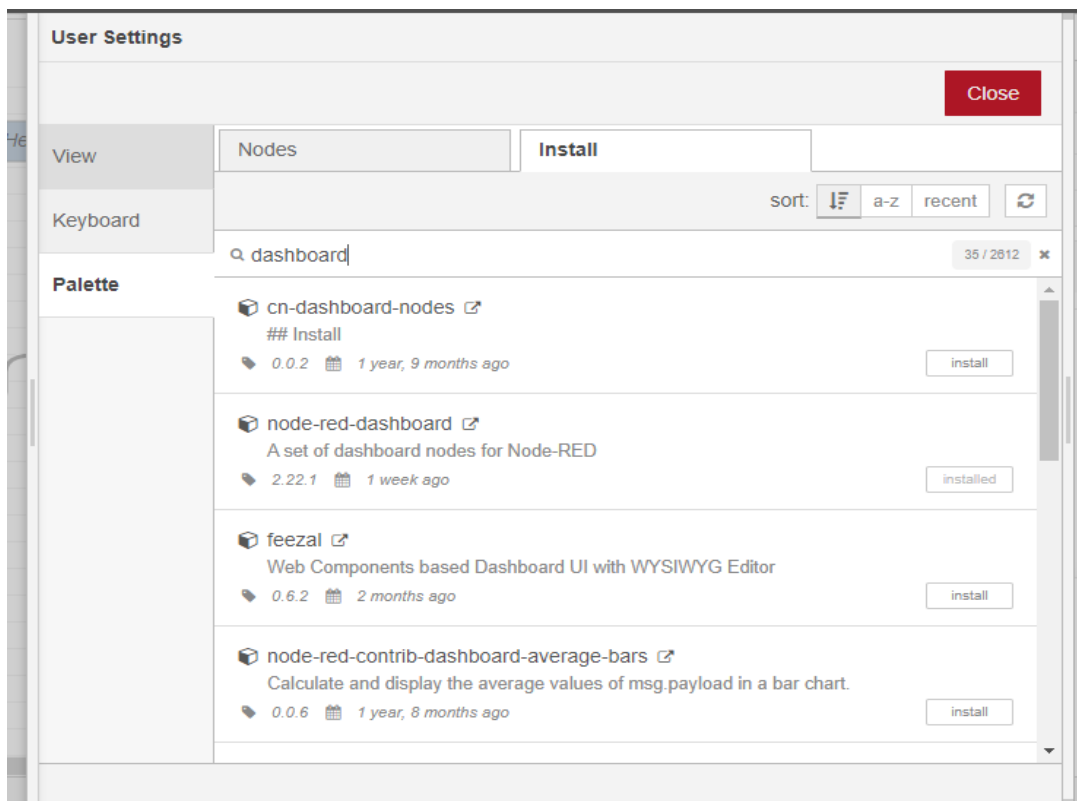
	If assistant recognizes	Respond with		
1	<code>\$webhook_result_1</code>	<code>"&lt;?\$webhook_result_1.passaj</code>		
2	<code>anything_else</code>	Try again later.		

10. Try out the assistant.



## Integration of all the above services in NODE-RED

1. Open the Node-Red Flow editor. Before beginning, download dashboard nodes used for creating chatbot UI. Click on **Manage Palette**. Go to install tab and search for node-red-dashboard. Then click on install.



2. Add **Form** node. Double click on it and add the Group name and Tab name. Also select the size of form.
3. Add **Inject** node. Double click it and name it as Hello.
4. Add two **Text** nodes. Double click the first one and name it as Query and the second one as the Response.
5. Drag two **Debug** nodes.
6. Drag two **Functions** nodes. Add in the codes for both the nodes.

**Edit function node 1**

Delete Cancel Done

**Properties**

Name

**Function**

```

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

```

**Edit function node 2**

Delete Cancel Done

**Properties**

Name

**Function**

```

1 msg.payload.text="";
2 if(msg.payload.context.webhook_result_1){
3   for(var i in msg.payload.context.webhook_result_1
4     msg.payload.text=msg.payload.text+"\n"+msg.payload
5   }
6   msg.payload=msg.payload.text;
7 }
8 else
9   msg.payload = msg.payload.output.text[0];
10 return msg;

```

7. Drag **Assistant** node. Double click on it and it will ask credentials like API key, Workspace ID and Service Endpoint which you will get from Watson Assistant service page. Get the details and enter them and click on Done.

**Edit assistant node**

Delete Cancel Done

**Properties**

Name

Username

Password

API Key

Service Endpoint

Workspace ID

Timeout Period

☐ Save context

Resource list /

## Watson Assistant-vw Active Add tags

[Details](#) [Actions...](#)

Manage

**Service credentials**

Plan

Connections

You can generate a new set of credentials for cases where you want to manually connect an app or external consumer to an IBM Cloud™ service. [Learn more](#)

Search credentials...

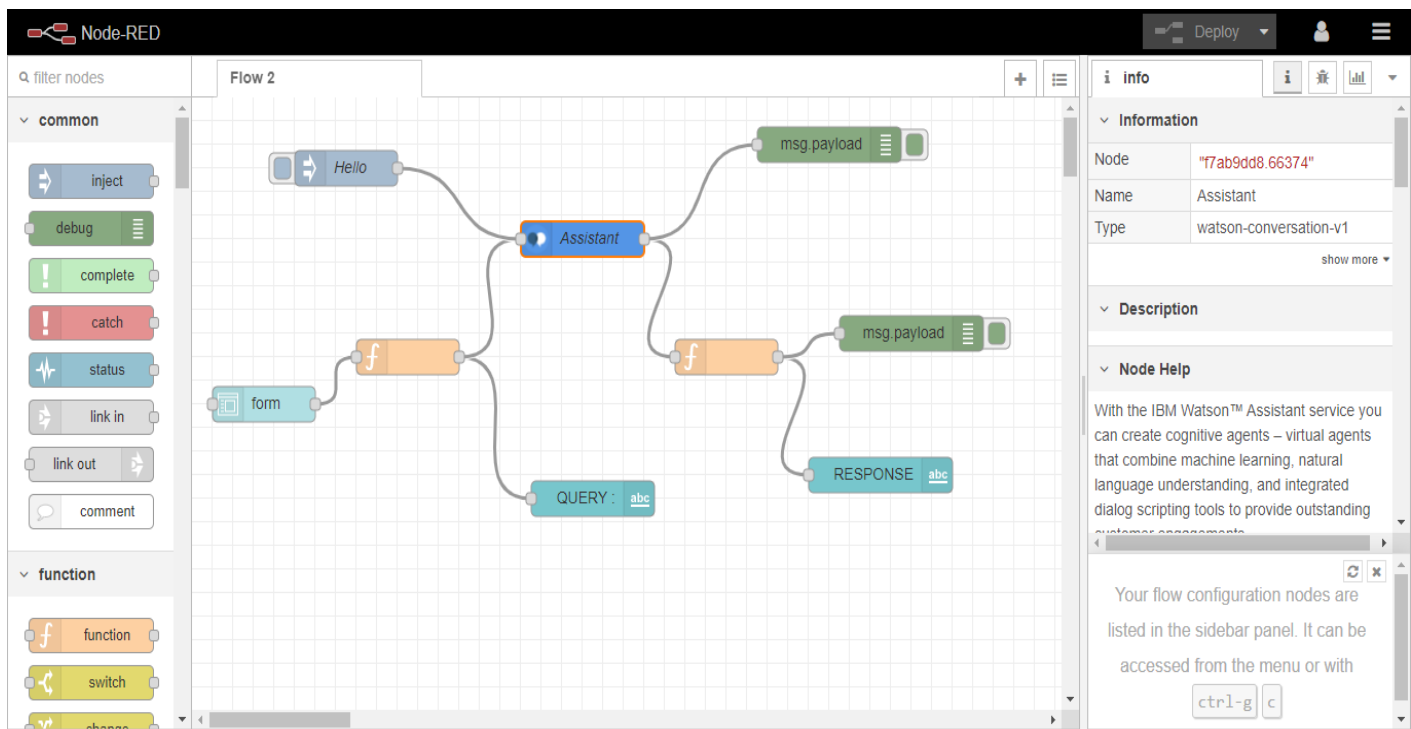
[New credential](#) +

Key name	Date created
Auto-generated service credentials	MAY 14, 2020 - 11:03:22 PM

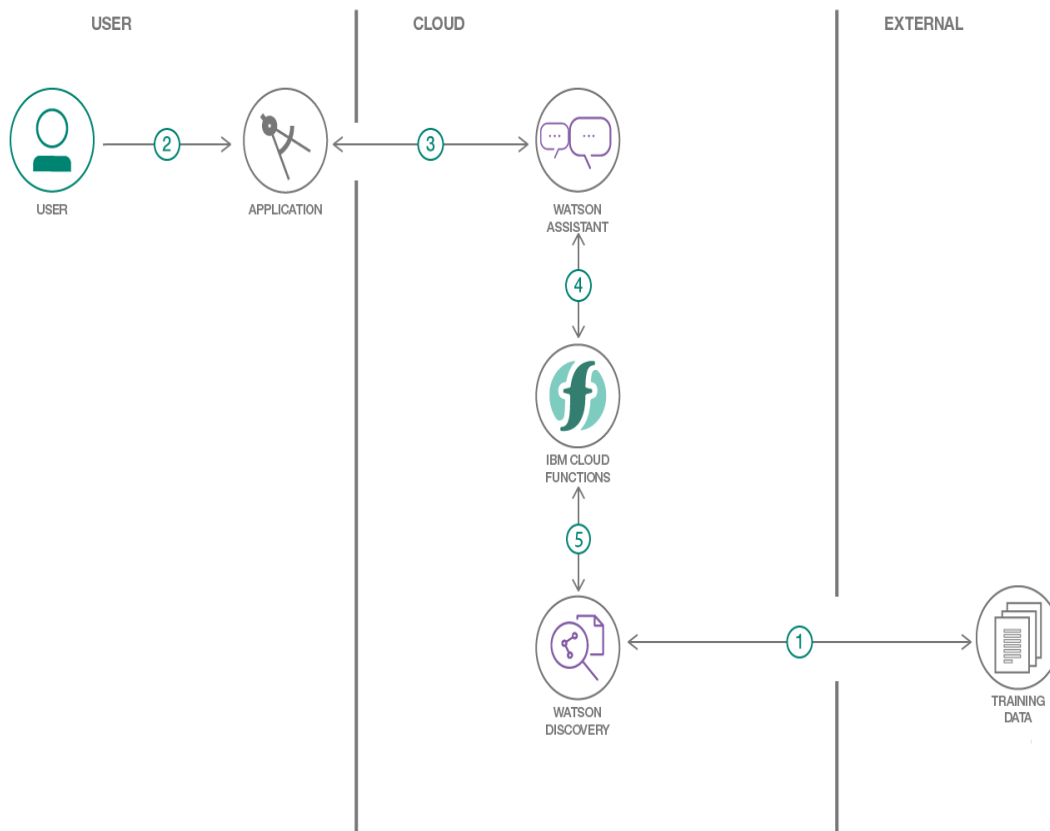
```
{
  "apikey": "YwtkrjfnxLiXQjw!3pGi0o136BGv4de2oiY",
  "iam_apikey_description": "Auto-generated for key f89dbb2f-b18a-4b15-b332-34e2130c9c17",
  "iam_apikey_name": "Auto-generated service credentials",
  "iam_role_crn": "crn:v1:bluemix:public:iam:::serviceRole:Manager",
  "iam_serviceid_crn": "crn:v1:bluemix:public:iam-identity::a/f2554465f6c8423e818f60065e452137::serviceid:ServiceId-2b87efd3-46e0-41b8-966d-4e21356083bc",
  "url": "https://api.eu-gb.assistant.watson.cloud.ibm.com/instances/573d7901-ceed-4dab-8d12-f37af61cb0f7"
}
```

8. Connect the form node to input of function node 1 and its output to input of Assistant node. Connect output of function node 1 to "Query" text node. Connect the output of Assistant node to input of function node 2 and its output to "Response" text node.

9. Click on Deploy.



# FLOWCHART



The flow summarizes the functionality of the Chatbot.

The document preloaded in Discovery is being annotated by Smart Document Understanding feature. This trains the Discovery to improve the query result. The user interacts with frontend ui of the chatbot and it keeps them engaged in conversation with small talk. This interaction between the user and the UI is being coordinated by Watson Assistant.

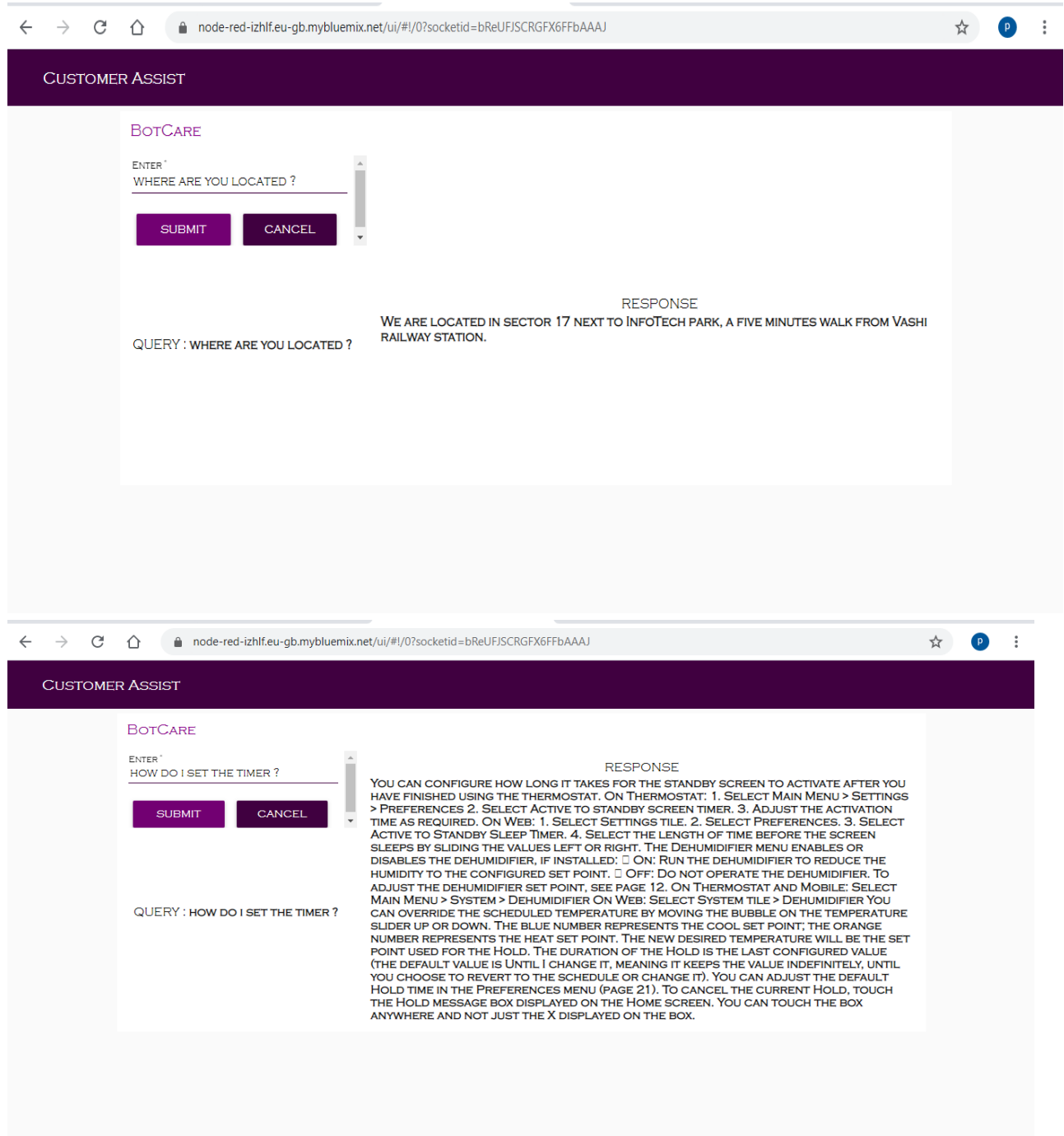
When user asks a technical query, the Assistant invokes the Cloud Function action. This action then queries Watson Discovery and returns with relevant result from the pre-loaded owner's manual.

Watson Assistant, Discovery and Cloud Functions are IBM services. The Training Data is the owner's manual which is external section. The user and applications comes under User section because the application is accessed by the user.



# RESULT

After deploying the flow in Node-Red, open the UI by typing <https://node-red-izhlf.eu-gb.mybluemix.net/ui/> in the browser. You will see the Help Desk created. The layout of the page can also be customized.



## **ADVANTAGES / DISADVANTAGES**

### **Advantages**

- They are available 24/7 which maintain continuous communication between the customer and the seller.
- An operator can concentrate on one customer at a time. A chatbot can however, answer thousands of questions at the same time.
- This reduces the cost of manpower.
- Actions like changing or querying records are almost instantaneous for bots which can significantly improve customer experience.
- Since bots are on digital platforms where people spend majority of their time, bots can be used to automate common tasks such as providing advanced search functionality.

### **Disadvantages**

- Chatbots are installed in the aim of getting fast response and improving customer interaction. However, due to limited availability of data and the time needed for self updation, the process can be slow and costly.
- For better outcome, more complex designing is required and need of skilled developers.
- At times they can mislead customer if the query doesn't match with the pre loaded dialog. Bots get confused and respond with same answer for different questions. This leads to frustrated customers.
- Unlike humans, each bot needs to be programmed differently for each business, which increases the initial installation cost.

## APPLICATIONS

Integration of chatbots with social media platforms like Facebook Messenger, LINE, WeChat, and WhatsApp make it easier for businesses to provide 24/7 customer service to the user.

- For travel industry, such customer service chatbot proved to be a boon. Apart from benefit of booking and scheduling flights, they help to integrate additional services via social media platforms. Services like Airbnb, Uber can be integrated for enhancement of customer experience.
- One of the most useful application of these bots is in healthcare department. An average patient spends 30 minutes trying to get to the right service in a local hospital. But deploying conversational chatbots in the healthcare sector can significantly reduce long waits. From registration to coverage and claims, chatbots are in popular demand in this industry.
- The use of chatbots in any appliance website is a very common thing today. Earlier, chatbots would transfer any function related query onto an agent. But with growing technology, that is also eliminated. The bot will give accurate result without any human interaction.

## **CONCLUSION**

In conclusion, we have created a chatbot which not only engage the customer with small talk but will also provide user with accurate and smart solution. The response will be fast and customer will be satisfied with the service.

By integrating all the above mentioned services, a smarter and efficiently working Customer Help Desk has been developed.

## **FUTURE SCOPE**

While most businesses will have chatbot for customer services, many will develop bots for their internal processing.

One such thing will be Bots handling first stage interviews to find promising candidates. The bots can process resumes to find highly qualified candidates and fasten the otherwise slow recruiting process and managing cost.

Customer service is increasingly an automated one. Many companies are adopting AI tools to build a better way of handling and expanding the service.

Chatbots have a long way to go to realize their full potential. Still, the chatbots will ultimately generate significant future value in both corporate and customer settings.

## **BIBLIOGRAPHY**

1. <https://developer.ibm.com/tutorials/how-to-create-a-node-red-starter-application/>
2. <https://cloud.ibm.com/docs/services/discovery?topic=discovery-getting-started>
3. <https://eu-gb.discovery.watson.cloud.ibm.com/regions/eu-gb/services/crn%3Av1%3Abluemix%3Apublic%3Adiscovery%3Aeu-gb%3Aa%2Ff2554465f6c8423e818f60065e452137%3A372afc3e-36b8-46da-9208-5b460b3627a8%3A%3A>
4. <https://eu-gb.assistant.watson.cloud.ibm.com/eu-gb/crn:v1:bluemix:public:conversation:eu-gb:a~2Ff2554465f6c8423e818f60065e452137:573d7901-ceeda-4dab-8d12-f37af61cb0f7:/home>
5. <https://node-red-izhlf.eu-gb.mybluemix.net/red/#flow/20dda74.f6e756>

# APPENDIX

## A. Source Code

Code to invoke action :

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

```

Code for Watson Assistant :

<https://github.com/SmartPracticeschool/IISPS-INT-619-Intelligent-Customer-Help-Desk-with-Smart-Documents-Understanding/blob/master/WATSON-ASSISTANT/skill-My-first-skill.json>