**PROJECT REPORT**

**Title: Intelligent Customer Help Desk With Smart Document Understanding**

**Submitted by: Manvi Kaul Artificial Intelligence Developer**

**INTRODUCTION**

The typical customer care chatbot can answer simple questions, such as store locations and hours, directions, and maybe 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 isn’t valid or offer to speak to a real person.

In this project, there will be another option. 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?” we can return relevant sections of the owners manual to help solve our customers’ problems.

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.

**SCOPE OF WORK**

* 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
* Build a web application with integration to all these services & deploy the same on IBM Cloud Platform

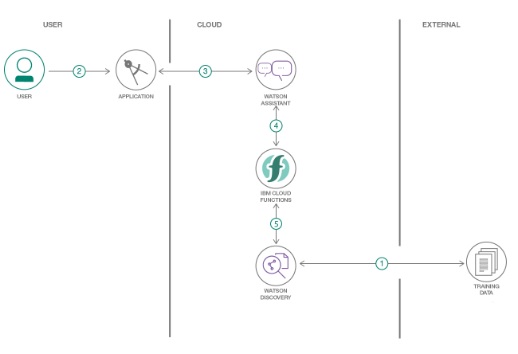
**Project Requirements**: Python, IBM Cloud, IBM Watson

**Functional Requirements**: IBM cloud

**Technical Requirements**: AI, ML, WATSON AI, PYTHON

**Software Requirements**: Watson assistant, Watson discovery.

**PROPOSED WORK FLOW**



**PROPOSED SOLUTION**

Steps involved in building the project:

1. Create IBM Cloud services
2. Configure Watson Discovery
3. Create IBM Cloud Functions action.
4. Configure Watson Assistant.
5. Create a Node red flow to connect all the services together.
6. Deploy and run the Node Red application.

# Create IBM Cloud services

Create the following services:

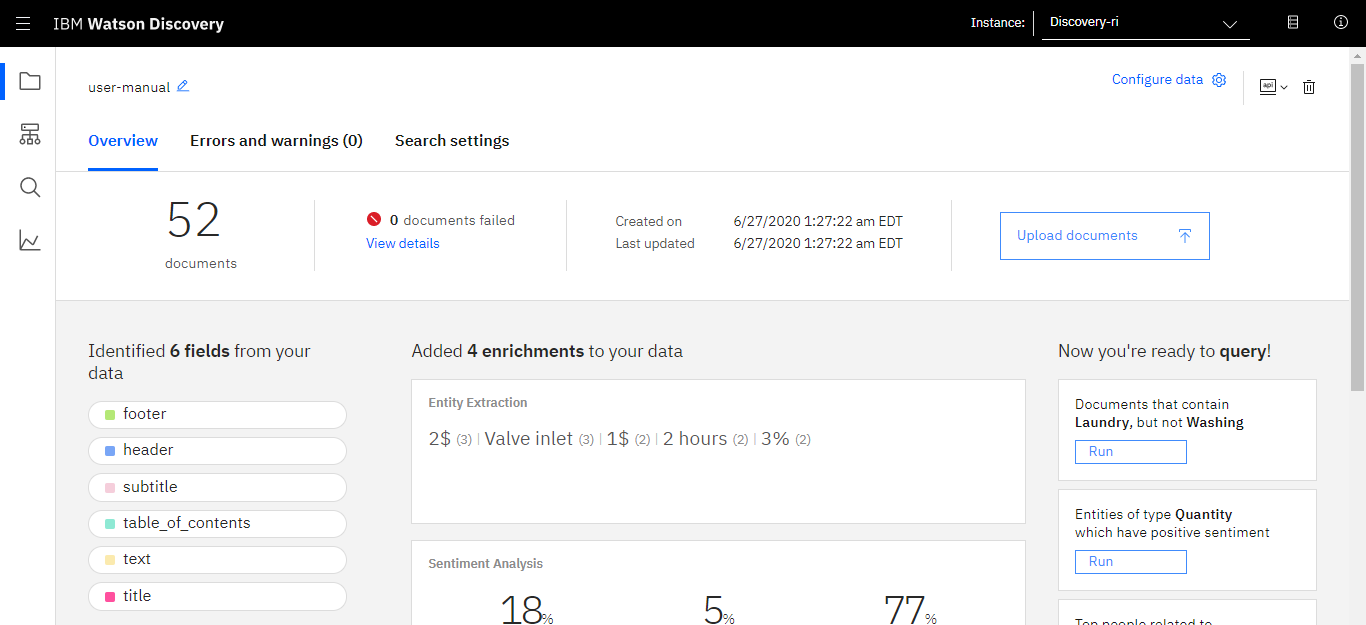
* + Watson Discovery
  + Watson Assistant
  + Node Red

1. **Configure Watson Discovery**

With IBM Watson Discovery, you can ingest, normalize, enrich, and search your unstructured data (JSON, HTML, PDF, Word, and more) with speed and accuracy. It packages core Watson APIs such as Natural Language Understanding and Document Conversion along with UI tools that enable you to easily upload, enrich, and index large collections of private or public data.

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 prompted, select and upload the file located in the data directory of your local system.

Now apply SDU to the uploaded document to improve the accuracy and generate some better query responses. From the Discovery collection panel, click the Configure data button (located in the top right corner) 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.



1. **Create IBM cloud Functions Action**

IBM Cloud Functions is a distributed compute service that executes application logic in response to requests from web or mobile apps. You can set up specific actions to occur based on HTTP-based API requests from web apps or mobile apps and from event-based requests from services like Cloudant.

Now 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, and then select the Functions card.

From the 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 Action Name. Keep the default package and select the Node.js 10 runtime. Click the Create button to create the action.

Once your action is created, click on the Code tab.

In the code editor window, type the code to connect to the Discovery service, make a query against the collection, and then returns the response.

Select the Parameters tab and add the following keys:

url

environment\_id

collection\_id

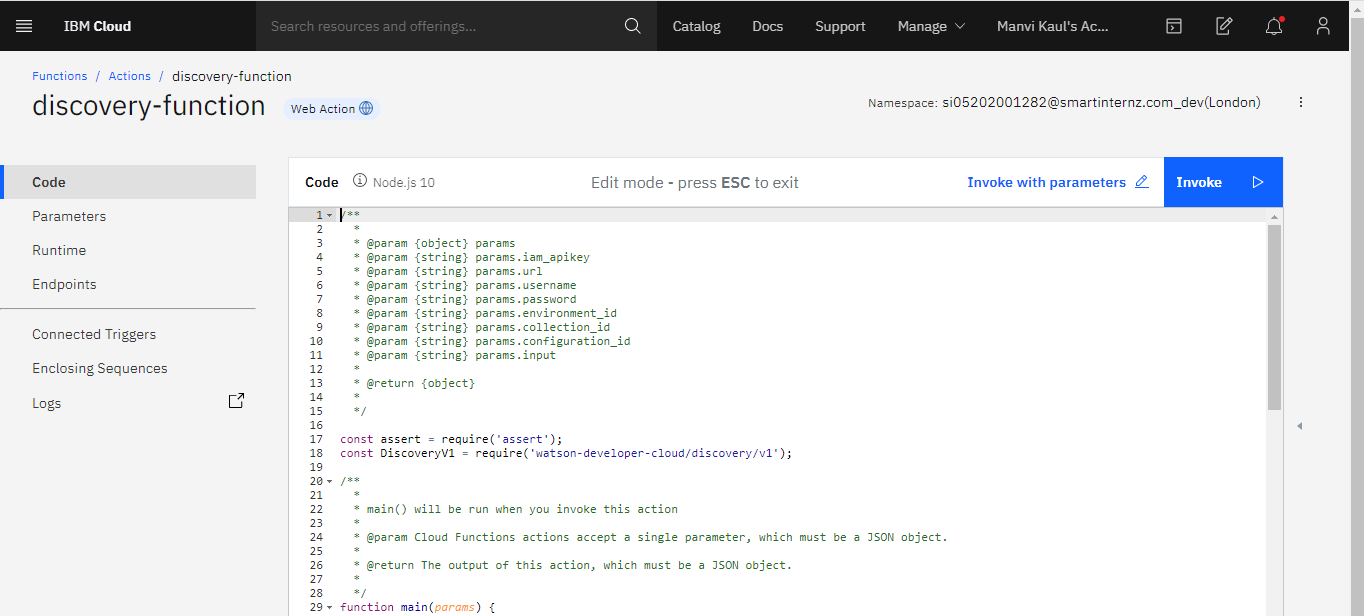
iam\_apikey

For the values, use the values associated with the Discovery service created in the previous step. Now that the credentials are set, return to the Code panel and press the Invoke button. This displays the actual results returned from the Discovery service:

Next, 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 a future step.

To verify that the correct Discovery parameters are entered, execute the provided command. If it fails, re-check the parameter values.



1. **Configure Watson Assistant**

Watson Assistant is a conversation AI platform that helps you provide customers fast, straightforward and accurate answers to their questions, across any application, device or channel. By addressing common customer inquiries, Watson Assistant reduces the cost of customer interactions, helping your agents focus on complex use cases – not repetitive responses.

Launch the Watson Assistant tool and create a new dialog skill. Select the Use sample skill option as your starting point. This dialog skill contains all of the nodes needed to have a typical call center 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 add this. Create a new intent that can detect when the user is asking about operating the Product.From the Customer Care Sample Skill panel, select the Intents tab. Click the Create intent button and name the intent #Product\_Information.

*Create new dialog node*

Now we need to add a node to handle our intent. Click on the Dialog tab, then click on the drop-down menu for the Small Talk node, and select the Add node below option.

Name the node "Ask about product" and assign it our new intent. This means that if Watson Assistant recognizes a user input such as "how do I set the time?", it will direct the conversation to this node.

*Enable webhooks from Assistant*

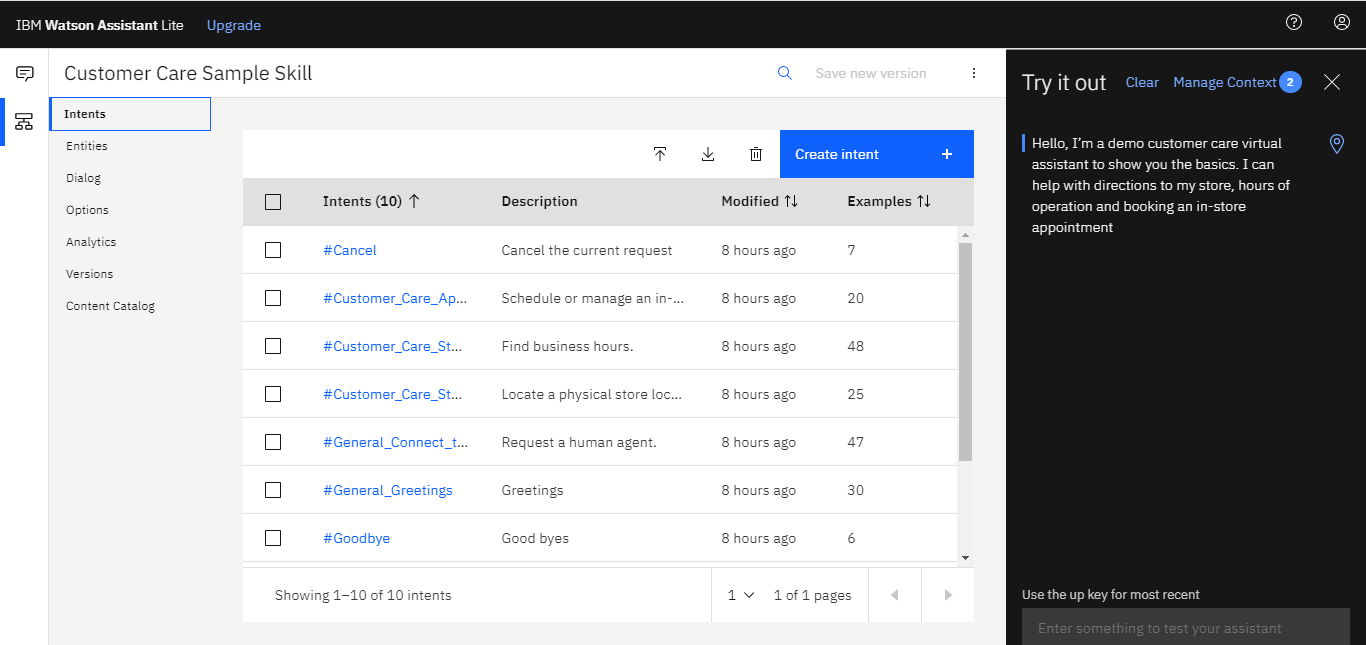
Set up access to our Webhooks for the IBM Cloud Functions action. Select the Options tab.

Enter the public URL endpoint for your action. Return to the Dialog tab, and click on the Ask about product node. From the details panel for the 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 you can use to access the result from the Discovery service query.

*Test chatbot in Assistant Tooling*

From the Dialog panel, click the Try it button located at the top right side of the panel. Enter some user input: Note that the input "how do I turn on the heater?" has triggered our Ask about product dialog node, which is indicated by the #Product\_Information response. And because we specified that $webhook\_result\_1.passages be the response, that value is displayed also. You can also verify that the call was successfully completed by clicking on the Manage Context button at the top right. The response from the Discovery query will be stored in the$webhook\_result\_1 variable.



1. **Create a Node red flow to connect all the services together**

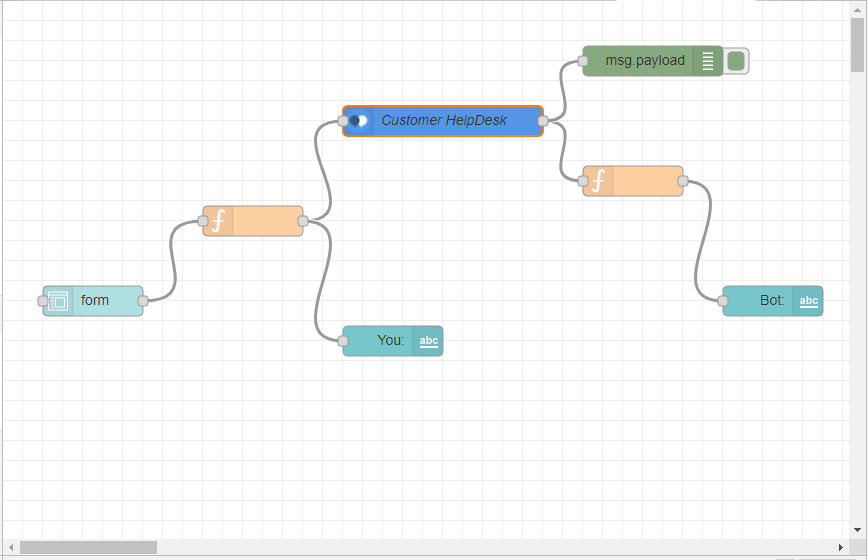
Node-red is a programming tool for wiring together of hardware devices, APIs and onile services in new and exciting ways.

Itprovides a browser-based flow editor that makes it easy to wire together flows using the wide range of nodes in the palette. Flows can be then deployed to the runtime in a single-click. JavaScript functions can be created within the editor using a rich text editor.

A built-in library allows you to save useful functions, templates or flows for re-use.

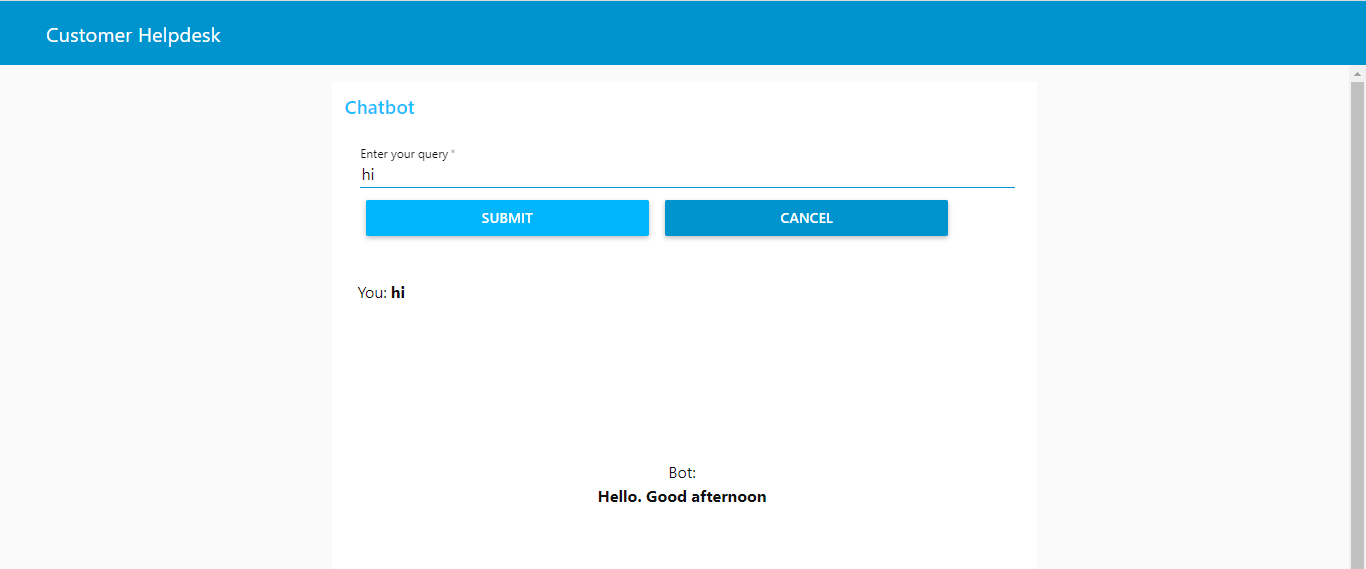
Create the flow with the help of the following nodes:

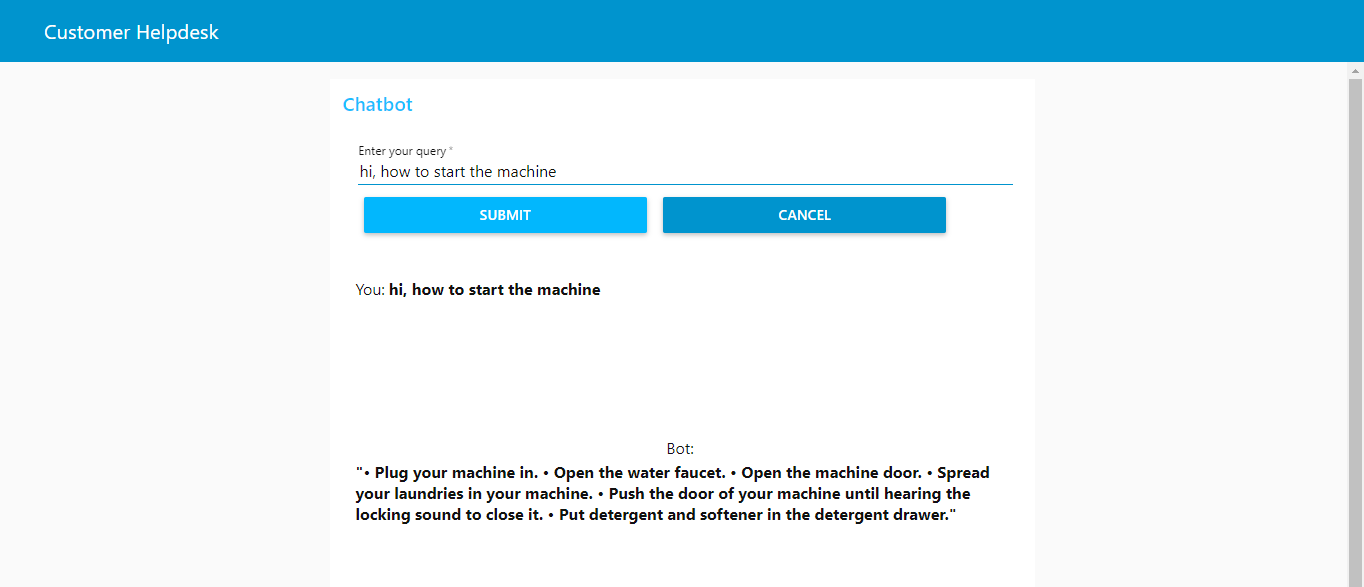
* Inject
* Assistant
* Debug
* Function
* Ui\_Form
* Ui\_Text



1. **Deploy and Run the Node-red Application**

Deploy the node red application and note down the results.





**ADVANTAGES**

* **Reduced costs:** Chatbots eliminate the need for labor during online interaction with customers. This is obviously a great advantage for companies that receive multiple queries at once. In addition to saving costs with them, companies can align the chatbot with their objectives, and use them as a means to enhance customer conversion.
* **24/7 Availability:** Unlike humans, once we install a chatbot, it can handle queries at any time of day. Thus, the customer does not have to wait for a commercial of the company to help him. This also allows companies to monitor customer « traffic » during non-working hours and contact them later.
* **Learning and updating:** AI-based chatbots are able to learn from interactions and update independently. This is one of the main advantages. When you hire a new employee, you have to train them continuously. However, chatbots « form » themselves (with certain limitations, of course).
* **Management of multiple clients:** Humans can serve a limited number of customers at the same time. This restriction does not exist for chatbots, and they can manage all the necessary queries simultaneously. This is one of the main advantages of using chatbot, as no customer is left unattended and you are solving different problems at the same time. There are chatbots companies already working on developing voice chatbot services.

**DISADVANTAGES**

* **Time-consuming:** Chatbots are installed with the aim of speeding up responses and improving customer interaction. However, due to the limited availability of data and the time needed for self-updating, this process can be slow and costly. Therefore, there are times when instead of serving several customers at once, chatbots may become confused and not serve the customer well.
* **Null decision making:** Chatbots can attack the nerves of more than one because they are not able to make decisions.
* **Bad memory:** The chatbots are not able to memorize a conversation already had, which forces the user to write the same thing over and over again. This can be cumbersome for the client and annoying for the effort required. Therefore, it is important to be careful when designing chatbots and make sure that the program is able to understand users’ queries and respond accordingly.

**FUTURE SCOPE**

Future Scope of this chatbot can be by adding the following to make it more advance: -

* Smarter Virtual Assistants: Much of what virtual assistants do now are basic skills, such as retrieving data and basic computation. As natural language processing (NLP) continues to mature, virtual assistants will improve their comprehension and response capabilities, allowing for their use to become more widespread and complex. Also, as machine learning progresses, we may see virtual assistants become smarter and begin to learn and predict customer needs.
* Integration with IoT Devices: Car speakers, smart home devices, and wearables are just a few examples where the virtual assistant is departing from its original hardware and making its way to in-context devices. These integrations ensure that virtual assistants can always be near their human and ready to support any need. It is expected that these integrations will continue at an accelerated pace throughout 2018.
* Voice-control: Voice recognition can be added with the virtual assistant. Then the customer can control application by using his voice. Soon, we could be joining meetings with a voice command, instead of dialing in the long meeting ID and password.