

PROJECT REPORT

Intelligent Customer Help Desk with Smart Document
Understanding

Project By :-
Ishika Garg
gargishika1998@gmail.com

INDEX

1 INTRODUCTION

1.1 Overview

1.2 Purpose

2 LITERATURE SURVEY

2.1 Existing problem

2.2 Proposed solution

3 THEORITICAL ANALYSIS

3.1 Block diagram

3.2 Hardware / Software designing

4 EXPERIMENTAL INVESTIGATIONS

5 FLOWCHART

6 RESULT

7 ADVANTAGES & DISADVANTAGES

8 APPLICATIONS

9 CONCLUSION

10 FUTURE SCOPE

11 BIBILOGRAPHY

APPENDIX

A. Source Code

B. Reference

1. INTRODUCTION

1.1. OVERVIEW:

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.

1.2. PURPOSE:

To create a chat-bot which can answer simple questions as well as return more and more accurate answers.

2. LITERATURE SURVEY

2.1. EXISTING PROBLEM:

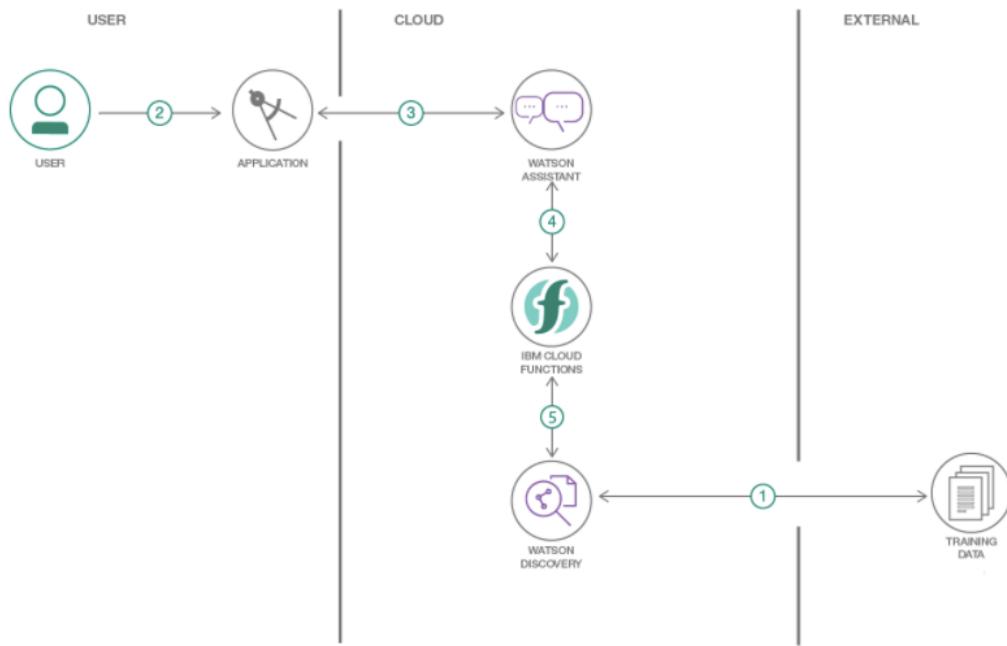
The typical customer care chat-bot can answer simple questions, such as store locations and hours, directions, and maybe even making appointments. When a question falls outside of the scope of the pre-determined question set, the option is typically to tell the customer the question isn't valid or offer to speak to a real person.

2.2. PROPOSED SOLUTION:

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 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, the project shall 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.

3. THEORITICAL ANALYSIS

3.1. BLOCK DIAGRAM:



1. The document is annotated using Watson Discovery SDU
2. The user interacts with the backend server via the app UI. The frontend app UI is a chatbot that engages the user in a conversation.
3. Dialog between the user and backend 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 Cloud Functions action will query the Watson Discovery service and return the results.

3.2. HARDWARE/SOFTWARE DESIGNING:

1. Create IBM Cloud services
2. Configure Watson Discovery
3. Create IBM Cloud Functions action
4. Configure Watson Assistant
5. Create flow and configure node

6. Deploy and run Node Red app.

4. EXPERIMENTAL INVESTIGATIONS

1.Create IBM Cloud Services -

To Create IBM Cloud, go to <https://www.ibm.com/cloud>

The screenshot shows the IBM Cloud homepage. At the top, there's a dark blue banner with white text: "Accelerate agility and efficiency with cloud". Below the banner, there's a sub-section titled "IBM Cloud Paks" with a blue button labeled "Learn more". To the right of this, there's a call-to-action "Talk to an expert →" and a "Let's talk" button. The URL in the address bar is visible as "https://cloud.ibm.com/explore/".

Click on sign up or login.

The screenshot shows the IBM Cloud sign-up/login page. On the left, there's a "Create an account" form with three steps: "1. Account information", "2. Verify email", and "3. Personal information". The first step has fields for "Email" and "Password", and a "Next" button. On the right, there's a large promotional section with a blue background and white text: "Build for free on IBM Cloud", "Develop for free, no credit card required", and "Access the full catalog at your fingertips". There's also a "Catalog" and "Docs" link at the top right of the main content area.

For Cloud Sign-Up: Follow the steps on the screen and fill in all the required details to create a new cloud account.

For Cloud Log In: Click on [Already have an IBM Cloud account? Log in](#) and fill your credentials to Log in to your cloud account.



After Logging in, you can see the IBM Cloud Dashboard.

A screenshot of the IBM Cloud Dashboard. The top navigation bar includes 'IBM Cloud', a search bar, and links for 'Catalog', 'Docs', 'Support', 'Manage', and 'Hashim Irfan Ali...'. Below the navigation is a 'Dashboard' section with a 'Resource summary' card showing 13 resources, including Cloud Foundry apps, services, storage, and functions namespaces. There are also cards for 'Planned maintenance', 'For you' (with a video thumbnail and a 'Learn Infrastructure as Code' link), 'News' (with articles about Airtel and Vodafone), 'Recent support cases', 'User access' (with a field to enter email addresses), and 'IBM Cloud status' (with a world map). At the bottom right is a prominent blue 'Create resource +' button.

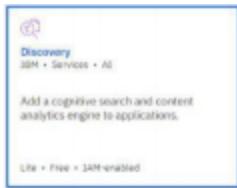
To Create any Resource (Services/Apps/etc), click on **Create resource +**

The screenshot shows the IBM Cloud Catalog homepage. On the left, there's a sidebar with categories like Catalog, Featured, Services, and Software. The main area has a search bar at the top with the placeholder "Search resources and offerings...". Below it, a section titled "IBM Cloud products" says "Over 190+ products available for you to customize and build the solutions that you need for your business". There's a search bar labeled "Search the catalog...". A "Recommended for you" section lists services like App ID, Object Storage, Visual Recognition, Streaming Analytics, Continuous Delivery, and Speech to Text. Each service has a small icon, a name, a category (e.g., IBM + Services + Security and Identity), and a status (e.g., Lite + Free + IBM-enabled).

Using the search box, we can find the service we want. For this project, we need to Create the following services: 1. Watson Discovery 2. Watson Assistant 1. To create a Watson Discovery Service, search for Discovery in the search box.

The screenshot shows the IBM Cloud Catalog search results for "discovery". The search bar at the top contains "discovery". Below it, a heading says "Search results for 'discovery'" with "2 results". The first result is "Discovery" (IBM + Services + AI), which is highlighted with a blue border. It has a brief description: "Add a cognitive search and content analytics engine to applications." and a status: "Lite + Free + IBM-enabled". The second result is "HashiCorp Consul" (Third party + Software + Developer Tools), with a description: "Highly available and distributed service discovery and key-value store designed with support for the modern data center..." and a status: "Helm charts + IBM Kubernetes Service + Free".

Click on



The screenshot shows the IBM Cloud Catalog interface. In the top navigation bar, there is a search bar labeled "Search resources and offerings..." and several menu items: Catalog, Docs, Support, Manage, and Hashim Irfan Al... On the left, a sidebar lists categories like Catalog, IBM Cloud catalog, Featured, Services, Software, Category (with AI selected), Product (with Services selected), Provider (with IBM selected), and Pricing plan (with Lite and Free selected). The main content area is titled "Discovery" and shows a summary for a service named "Discovery-qk" located in London, Plan: Lite, Service name: Discovery-qk, and Resource group: Default. Below the summary, there are tabs for "Create" and "About", with "Create" being active. A dropdown menu for "Select a region" shows "London" is selected. A pricing plan table is displayed, showing the "Lite" plan which includes 0 - 1,000 documents per month, 200 news queries per month, and 1 custom model. The price is listed as "Free". A note states: "The Lite plan is a great starter plan to trial features at no cost. With 1000 documents, 200 news queries, and 1 custom model, test-drive the service's foundational capabilities. When you upgrade to a paid plan, you'll keep any content you have ingested. See documentation for details." Buttons for "Create", "Add to estimate", and "View terms" are present.

Select a region, select a plan, configure your service (Service name, etc) and click Create. Your Watson Discovery service is created successfully. (If you are on Lite Plan, you can have only one instance per service).

2. To create a Watson Assistant Service, search for Assistant in the search box

The screenshot shows the IBM Cloud Catalog search results for "assistant". The search bar at the top contains the query "assistant". The results page is titled "Search results for 'assistant'" and shows one result: "Watson Assistant" by IBM, Services, AI. The description states: "Watson Assistant lets you build conversational interfaces into any application, device, or channel." It is categorized under AI, Services, and AI. The price is listed as "Lite • Free • IAM-enabled". The URL shown is <https://cloud.ibm.com/catalog/services/watson-assistant?context=category>.



click on

The screenshot shows the IBM Cloud Catalog interface for creating a Watson Assistant service. The 'Create' tab is active. A dropdown menu for 'Select a region' has 'London' selected. Below it, a 'Select a pricing plan' section shows the 'Lite' plan is free, listing features like AI-based Intent and Entity Recognition, Entity Synonym Recommendations, Visual Dialog DM with Simple Response Types (Text, Options, Images, etc...), Prebuilt Content Available, Analytics Dashboard with 7 Days of Storage, 5 Dialog Skills, each with 100 Dialog Nodes, and Shared Public Cloud. To the right, a summary panel shows the service name is 'Watson Assistant', region is 'London', plan is 'Lite', and resource group is 'Default'. Buttons for 'Create', 'Add to estimate', and 'View terms' are visible.

Select a region, select a plan, configure your service (Service name, etc) and click Create. Your Watson Assistant service is created successfully. (If you are on Lite Plan, you can have only one instance per service). To check whether you have correctly configured the services, go back to the IBM Dashboard and click on View All from the Resource Summary Tab.

The screenshot shows the IBM Cloud Dashboard's 'Resource summary' section. It indicates there are 13 total resources. Under the 'Services' category, there are 6 resources listed. Other categories shown include Cloud Foundry apps (1), Cloud Foundry services (2), Storage (1), Functions namespaces (1), and Apps (1). A 'View all' button is at the top right, and an 'Add resources +' button is at the bottom right.

All of your existing Resource list will be shown here, click on Services to unveil the list of services you have.

The screenshot shows the IBM Cloud Resource list interface. On the left, there's a sidebar with categories like Devices, VPC Infrastructure, Clusters, Cloud Foundry apps, Cloud Foundry services, Services, and Storage. The Services section is expanded, showing several entries. One entry is 'Discovery-qk' which is listed as Active. Other entries include Watson Assistant, Watson Studio, and Watson Vision. The main table has columns for Name, Group, Location, Offering, Status, and Tags.

Name	Group	Location	Offering	Status	Tags
Discovery-qk	Default	London	Discovery	Active	
Watson Assistant-61	Default	London	Watson Assistant	Active	
Watson Studio-4h	Default	London	Watson Studio	Active	
node-red-qeyad-cloudant-1589268036...	Default	Chennai 01	Cloudant	Active	
watson-vision-combined-eq	Default	Dallas	Visual Recognition	Active	cont...

Here we can find that the status of Watson Discovery and Watson Assistant as Active which means we have configured the services correctly.

3. Configure Watson Discovery from the resource list screen, click to open Watson Discovery service

This screenshot shows the configuration page for the 'Discovery-qk' service. On the left, there's a sidebar with 'Manage' sections: Getting started, Service credentials, Plan, and Connections. The 'Plan' section shows a 'Lite' plan with an 'Upgrade' button. The main area has tabs for 'Start by launching the tool' (selected), 'Getting started tutorial', and 'API reference'. Below these tabs, there's a 'Credentials' section with fields for 'API key' and 'URL'. The URL field contains the value 'https://api.eu-gb.discovery.watson.cloud.ibm.com/instances/486633d2-d1aa-4747-9ff0-78446f4...'.

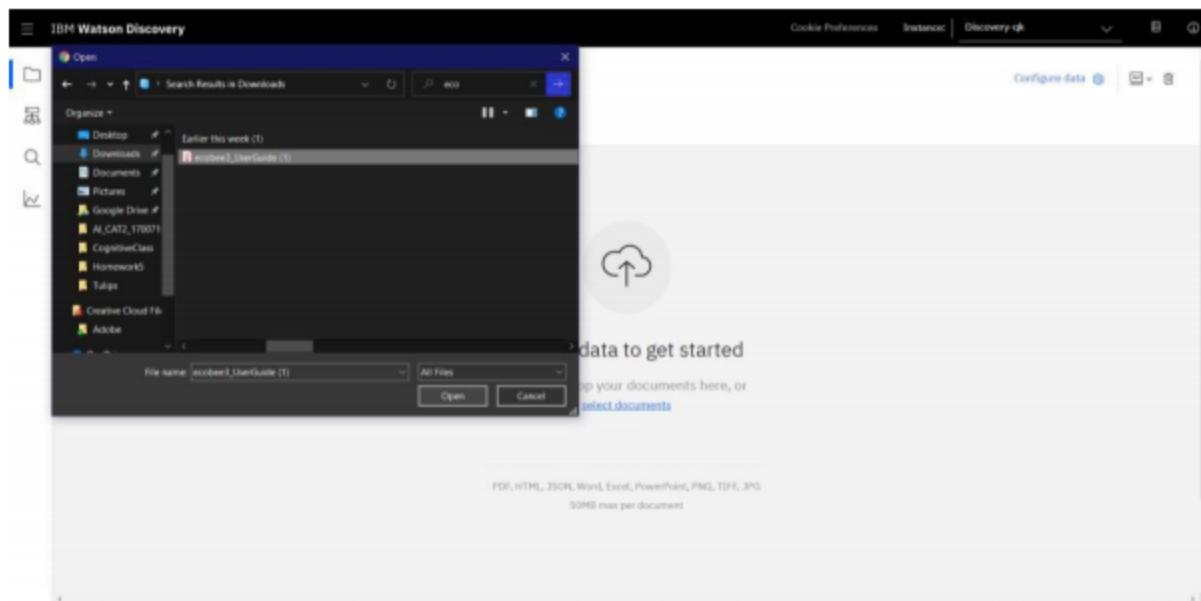
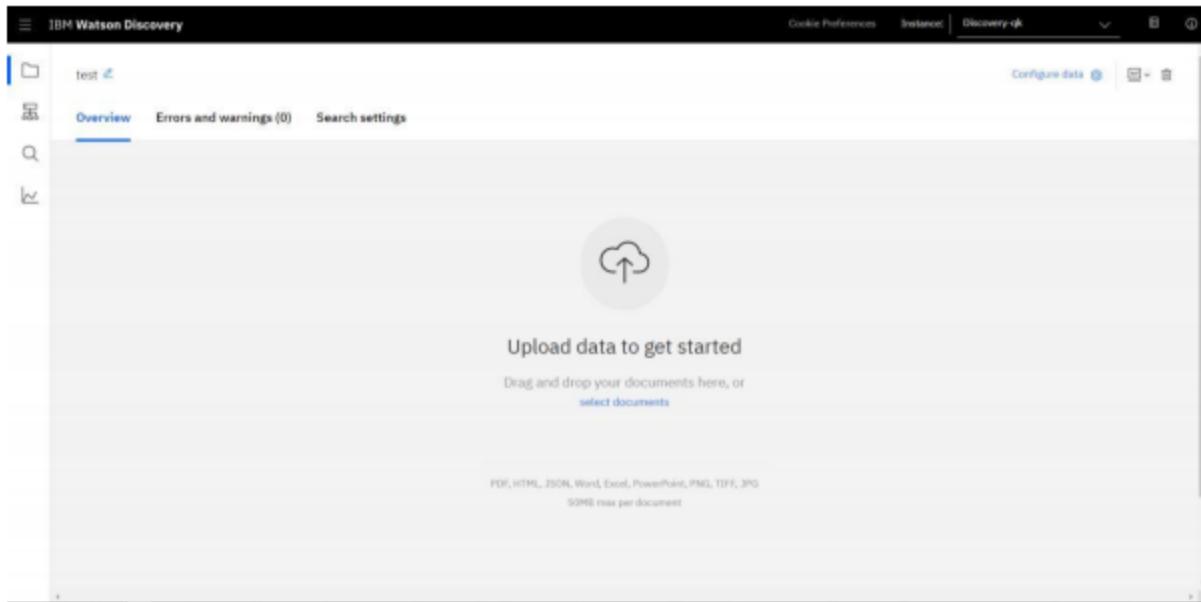
Click on **Launch Watson Assistant** to launch Watson Discovery Service.

The screenshot shows the IBM Watson Discovery interface. At the top, there's a navigation bar with icons for search, refresh, and other settings. Below it, the main title is "Manage data". A sub-instruction says "Collections of your private data and pre-enriched data to configure and query against. Learn more.". There are three main sections: "Create a new data collection" (with "Upload your own data" and "Connect a data source" buttons), "PRE-ENRICHED DATA" (listing "Watson Discovery News" with "News sources: English" dropdown and "Indexed on 5/12/2020"), and "PRIVATE DATA" (listing "owners-manual" with "Indexed").

Click on **Upload your own data** to create a new data collection.

This screenshot shows the same interface as above, but with a modal dialog box overlaid. The dialog is titled "Name your new collection". It has a text input field labeled "collection name" containing "Enter name" and a dropdown menu labeled "Select the language of your documents" set to "English". At the bottom, there are "Cancel" and "Create" buttons, with "Create" being highlighted in blue.

Give the data collection a unique name.



When prompted, select and upload the ecobee3_UserGuide.pdf file located in the data directory of your local repository.

The screenshot shows the IBM Watson Discovery interface. At the top, there are tabs for 'Overview', 'Errors and warnings (1)', and 'Search settings'. Below this, it says '1 document' and '0 documents failed'. There is a 'Upload documents' button. On the left, there's a sidebar with icons for folder, file, search, and refresh. The main area has three sections: 'Identified 1 field from your data' (text), 'Added 4 enrichments to your data' (Entity Extraction, Sentiment Analysis, Concept Tagging, Category Classification), and a sidebar with 'Now you're ready to query!' sections for entity types, documents containing specific terms, and top people related to certain topics.

Before proceeding further, let's learn about Smart Document Understanding(SDU). SDU trains Watson Discovery to extract custom fields in your documents. Customizing how your documents are indexed into Discovery will improve the answers returned from queries. With SDU, you annotate fields within your documents to train custom conversion models. As you annotate, Watson is learning and will start predicting annotations. SDU models can also be exported and used on other collections. Current document type support for SDU is based on your plan:

- Lite plans: PDF, Word, PowerPoint, Excel, JSON, HTML
 - Advanced plans: PDF, Word, PowerPoint, Excel, PNG, TIFF, JPG, JSON, HTML
- 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.

The screenshot shows the IBM Watson Discovery interface. At the top, there are tabs for 'Overview', 'Errors and warnings (1)', and 'Search settings'. Below this, a summary section indicates 1 document, 0 failed documents, and the creation date as 5/16/2020 11:29:11 am EDT. A 'Upload documents' button is also present. The main area displays several cards: 'Identified 1 field from your data' (text), 'Added 4 enrichments to your data' (Entity Extraction: 104 °F, 20 min, 20 minutes, 20 °C, 24-hour; Sentiment Analysis: 100% positive, 0% neutral, 0% negative; Concept Tagging: Air conditioner, Energy recovery, Geothermal heat pump; Category Classification: business and industry/energy). To the right, there are sections for 'Most common entity types and their top entities' (Run), 'Documents that contain Air conditioner, but not Energy recovery' (Run), and 'Top people related to /business and industrial/energy' (Run). A red box highlights the 'Build your own query →' button.

Click on Build your own query. Now, enter queries related to the operation of the thermostat and view the results. As you will see, the results are not very useful, and in some cases, not even related to the question.

The screenshot shows the 'Build queries' interface. On the left, a sidebar lists 'test / Build queries', 'Build a query using one or more of these components. Learn more.', 'Use a sample query', 'Search for documents', 'Use natural language' (selected), and 'Use the Discovery Query Language'. The main area contains a search bar with the query 'how do i turn on the heater'. Below the search bar are buttons for '+ Include analysis of your results' and '+ Filter which documents you query'. At the bottom are 'Run query' and 'Close' buttons. The right side shows the 'Summary' tab of the results, which includes a link to 'Query URL: https://api.eu-gb.discovery.watson.cloud.ibm.com/instances/6...', a 'Passages' section with a note about warranty, and a 'Results' section showing 1 matching document named 'ecobee3_UserGuide (1).pdf' with a sentiment of 'positive'.

Now let's apply SDU to our document to see if we can generate some better query responses. Go back to the Discovery collection panel (previous screen).

The screenshot shows the IBM Watson Discovery interface. At the top, there's a navigation bar with 'IBM Watson Discovery', 'Cookie Preferences', 'Instances', 'Discovery v8', and other icons. Below the navigation is a sidebar with a folder icon labeled 'test'. The main content area has tabs for 'Overview' (selected), 'Errors and warnings (1)', and 'Search settings'. A summary box indicates '1 document', '0 documents failed', and provides creation and last update dates (5/16/2020). A 'Configure data' button is highlighted with a red box. Below this, sections show identified fields (1 field), added enrichments (4 enrichments), and sentiment analysis (100% positive, 0% neutral, 0% negative). Other enrichments listed include Entity Extraction (e.g., 104 °F, 20 min, 20 minutes, 20 °C, 24-hour), Concept Tagging (e.g., Air conditioner, Energy recovery, Geothermal heat pump), and Category Classification (business and industry, energy). To the right, a sidebar says 'Now you're ready to query!' with options for documents about '104 °F', top people related to '/business and industrial/energy', and most common entity types.

Click the Configure data button (located in the top right corner) to start the SDU process. Here is the layout of the Identify fields tab of the SDU annotation panel.

This screenshot shows the 'Identify fields' tab of the SDU annotation panel. It displays a PDF document titled 'ecobee3_UserGuide (1).pdf' with page 1/41. Numbered annotations point to specific features: [1] points to the list of pages on the left; [2] points to the current page being annotated; [3] points to the text selection and labeling area; [4] points to the list of available field labels on the right; and [5] points to the 'Submit page' button at the bottom right.

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] is the list of pages in the manual. As each is processed, a green check mark will appear on the page. [2] is the current page being annotated. [3] is where you select text and assign it a label. [4] is the list of labels you can assign to the page text. Click [5] to submit the page to Discovery. As you go through the annotations one page at a time, Discovery is learning and should start

automatically updating the upcoming pages. Once you get to a page that is already correctly annotated, you can stop, or simply click Submit [5] to acknowledge it is correct. The more pages you annotate, the better the model will be trained. For this specific owner's manual, at a minimum, it is suggested to mark the following:

- The main title page as title
- The table of contents (shown in the first few pages) as table_of_contents
- All headers and sub-headers (typed in light green text) as a subtitle
- All page numbers as footers
- All warranty and licensing information (located in the last few pages) as a footer
- All other text should be marked as text.

The screenshot shows the 'Identify fields' tab in the IBM Watson Discovery interface. The left panel lists 41 pages of the 'ecobee3_UserGuide.pdf' document, each marked with a green checkmark. The right panel contains a 'Field labels' section with a list of document elements and their corresponding colors: answer (orange), author (blue), footer (green), header (blue), question (green), subtitle (pink), table_of_contents (teal), text (yellow), title (red), image (light blue), and table (red). A large blue button labeled 'Apply changes to collection' is located at the top right of the interface.

Click the **Apply changes to collection** [5]

You will be asked to reload the document. Choose the same ecobee3_UserGuide .pdf document as before. Now, click on Manage fields tab on the Configure data panel.

The screenshot shows the 'Manage fields' tab in the IBM Watson Discovery interface. On the left, there's a list of fields: answer, author, footer, header, image, question, subtitle, table, table_of_contents, text, and title. Most fields have their 'On/Off' switch turned off, except for 'subtitle' which is turned on (green switch). A red number '1' is overlaid on the 'subtitle' row. To the right, there's a section titled 'Improve query results by splitting your documents' with a dropdown menu set to 'subtitle', indicated by a red number '2'. At the top right, there's a large red number '3' next to the 'Apply changes to collection' button.

[1] Here is where you tell Discovery which fields to ignore. Using the on/off buttons, turn off all labels except subtitles and text. [2] is telling Discovery to split the document apart, based on subtitle. Click [3] to submit your changes. Once again, you will be asked to reload the document. Choose the same ecobee3_UserGuide .pdf document as before. Now, as a result of splitting the document apart, your collection will look different.

The screenshot shows the 'Overview' tab in the IBM Watson Discovery interface. It displays the following information:

- Documents:** 133 documents
- Failed:** 0 documents failed
- Created on:** 5/12/2020 4:09:01 am EDT
- Last updated:** 5/12/2020 4:09:01 am EDT
- Upload documents:** button

Below this, there are several sections showing analysis results:

- Identified fields:** 6 fields from your data (author, footer, subtitle, table_of_contents, Text, title)
- Added enrichments:** 4 enrichments to your data (Entity Extraction: 0.3°C, 0.5°F, 10 °F, 20 min, 4-digit; Sentiment Analysis: 54% positive, 35% neutral, 11% negative; Concept Tagging: HVAC, Heat, Internet, Temperature; Category Classification: technology and computing/operating systems)
- Now you're ready to query!** sections for Top people related to technology and computing/operating systems and Entities of type Quantity which have negative sentiment.

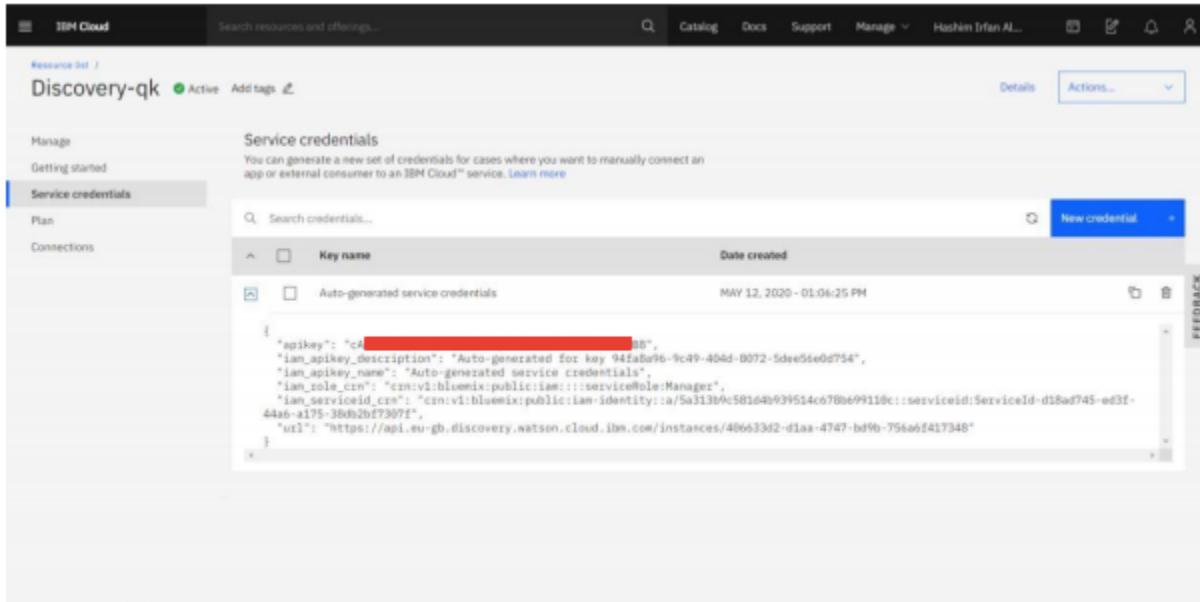
Now click the Build your own query and see how much better the results are.

The screenshot shows the IBM Watson Discovery interface. On the left, there's a sidebar with a folder icon, a magnifying glass icon, and a refresh icon. The main area has a search bar with placeholder text "Build a query using one or more of these components. Learn more." Below it is a "Search for documents" section with two tabs: "Use natural language" (selected) and "Use the Discovery Query Language". A text input field contains "how do i turn on the heater?". To the right of the search bar are buttons for "Use a sample query" and "Run query". Under the search bar, there are three expandable sections: "Include analysis of your results", "Filter which documents you query", and "More options". The "More options" section includes a "Run query" button and a "Close" button. On the far right, there's a "Summary" tab and a "JSON" tab. The "Summary" tab shows a query URL: "https://api.eu-gb.discovery.watson.cloud.ibm.com/instances/4". Below this is a "Passages" section with several text snippets. One snippet discusses a menu for testing wiring and connections. Another snippet about HVAC system settings depends on the type of system. A third snippet is about generating alerts for furnace filters. The "JSON" tab is partially visible on the right.

In upcoming steps, you will need to provide some credentials to access your Discovery collection so to Store credentials for future use follow the below steps. The Collection ID and Environment ID values can be found by clicking the located at the top right side of your collection panel.

The screenshot shows the "Overview" screen of a Watson Discovery collection named "owners-manual". The top navigation bar includes "Cookie Preferences", "Instances", "Discovery API", and a "Train Watson to improve results" button. The main area has a sidebar with a folder icon, a magnifying glass icon, and a refresh icon. The main content area shows the collection details: "132 documents", "0 documents failed", "Created on 5/12/2020 4:09:01 am EDT", and "Last updated 5/12/2020 4:09:01 am EDT". Below this, there are sections for "Identified 6 fields from your data" (author, footer, subtitle, table_of_contents, text, title) and "Added 4 enrichments to your data" (Entity Extraction, Sentiment Analysis, Concept Tagging, Category Classification). On the right, there are three cards: "Configure data" (Collection ID, Configuration ID, Environment ID), "Entities of type Quantity which have negative sentiment" (Run), and "Top people related to /technology and computing/operating systems" (Run). At the bottom, there's a note "5 enrichments available. Add enrichments".

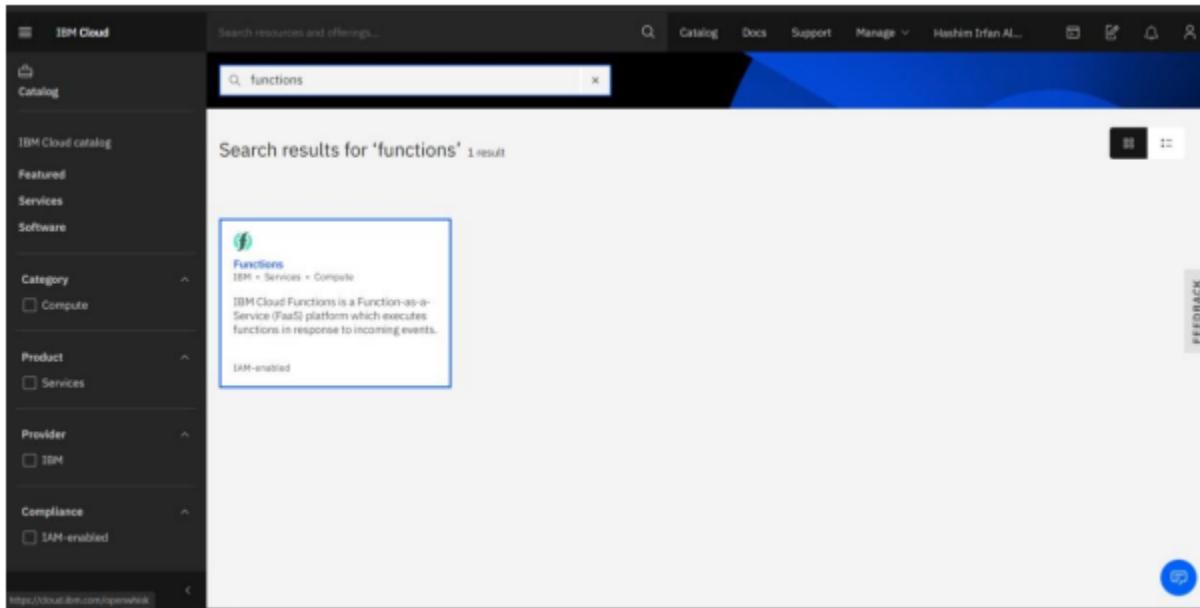
Now go to the Watson Discovery Resource List Screen. Here, select service credentials. The apikey and URL endpoint for your service can be found here.



The screenshot shows the IBM Cloud Service Credentials page for a resource named 'Discovery-qk'. The 'Service credentials' tab is selected. A table lists a single entry: 'Auto-generated service credentials' (Key name), created on MAY 12, 2020 - 01:06:25 PM. The JSON content of the credential is partially visible:

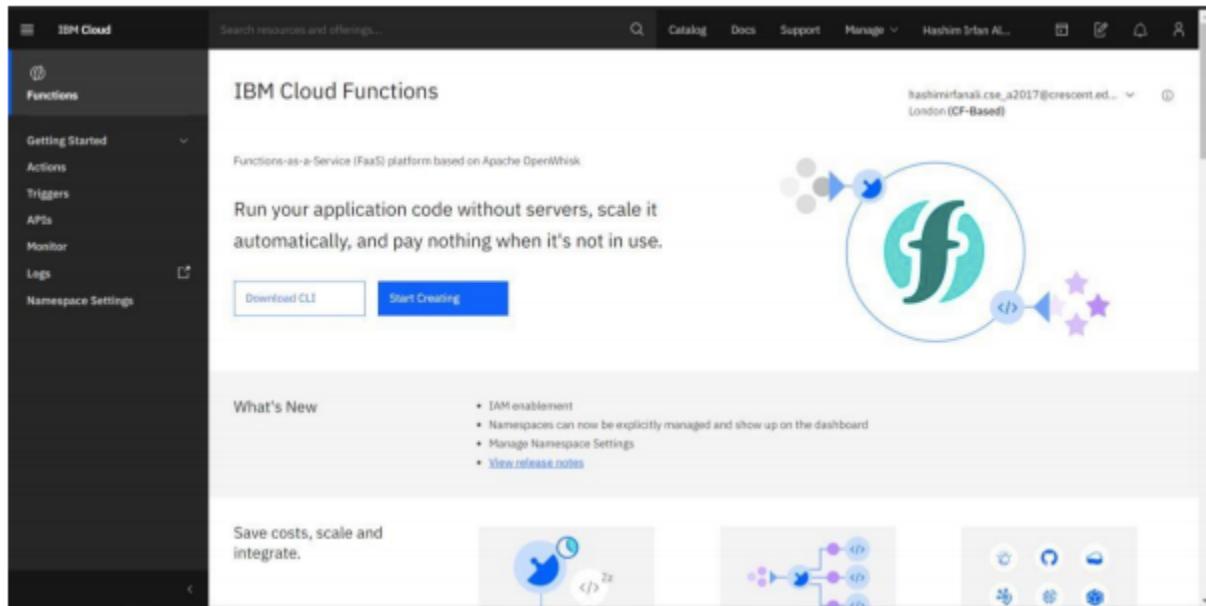
```
{ "apikey": "ca[REDACTED]8", "iam_apikey_description": "Auto-generated for key 94fa8096-9c49-404d-b072-5dee56e0d754", "iam_apikey_name": "Auto-generated service credentials", "iam_credential_crn": "cnr:ibm:bluemix:public:iam::::serviceRole:Manager", "iam_serviceid_crn": "cnr:ibm:bluemix:public:iam:identity::a:5a313bfc58164b939514c678b699110c::serviceId:ServiceId-d18ad745-ed3f-4486-a175-380bb2f79071", "url": "https://api.eu-gb.discovery.watson.cloud.ibm.com/instances/406633d2-d1aa-4747-bd9b-756a6f417348" }
```

4. Create IBM Cloud Functions action - Now let's create the web action that will make queries against our Discovery collection. Go to IBM Cloud Dashboard, click on Create Resource and search for Functions



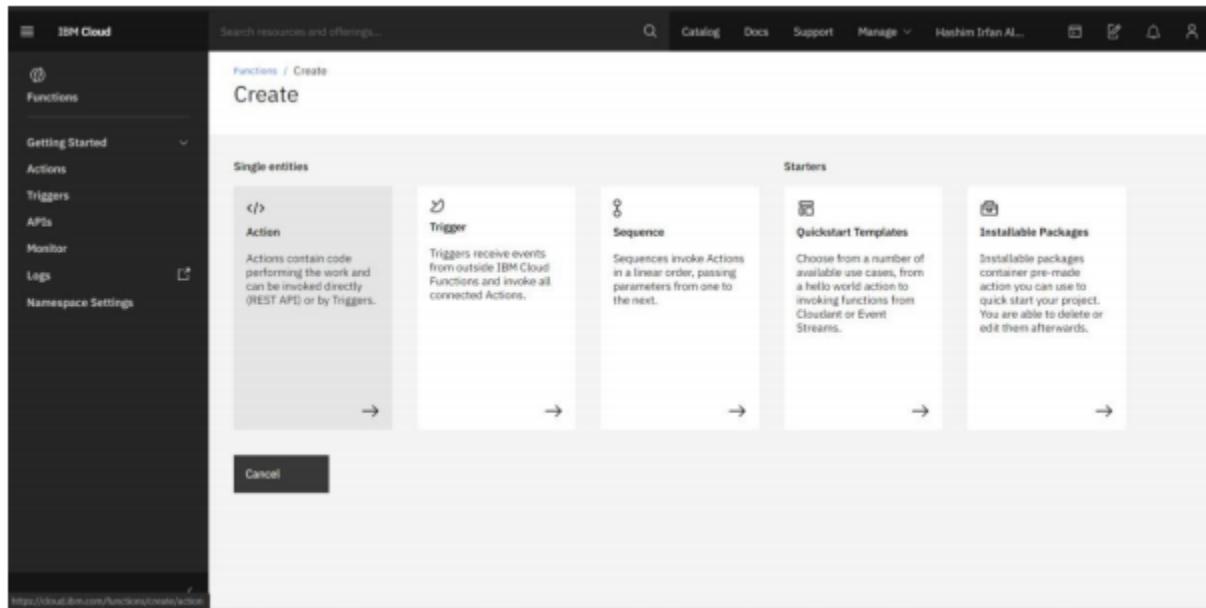
The screenshot shows the IBM Cloud Catalog search results for 'functions'. The search bar contains 'functions'. One result is displayed: 'Functions' under the 'IBM + Services > Compute' category. The description states: 'IBM Cloud Functions is a Function-as-a-Service (FaaS) platform which executes functions in response to incoming events.' The card is labeled 'IAM-enabled'.

Select the Functions Card.



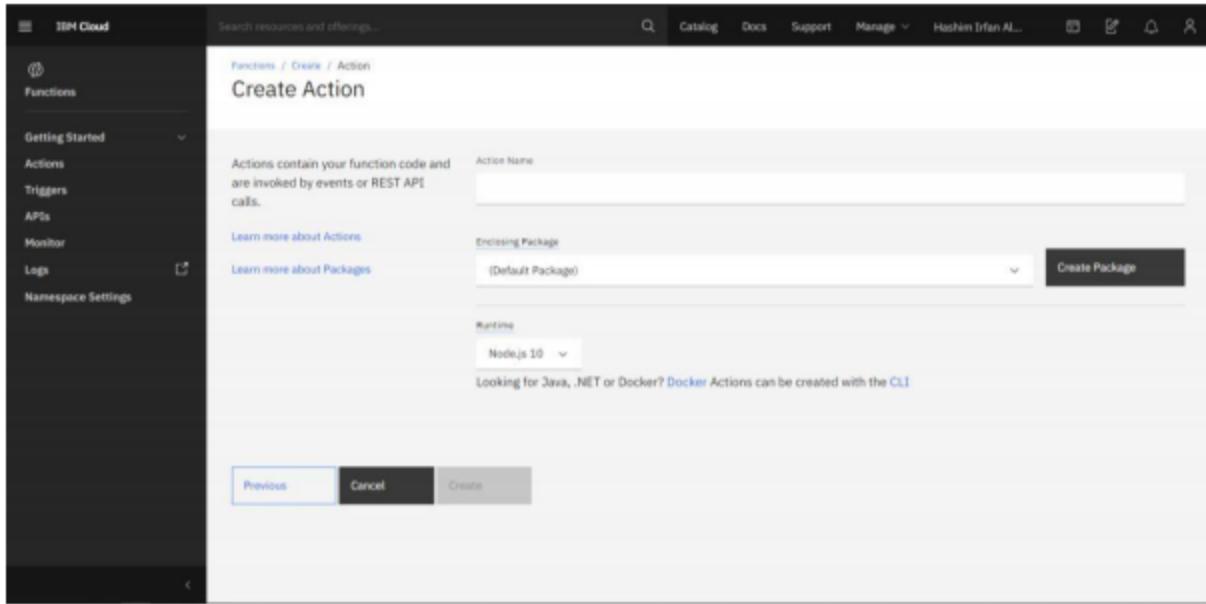
The screenshot shows the IBM Cloud Functions main panel. On the left, there's a sidebar with navigation links: Getting Started, Actions, Triggers, APIs, Monitor, Logs, and Namespace Settings. The main content area is titled "IBM Cloud Functions" and describes it as a "Functions-as-a-Service (FaaS) platform based on Apache OpenWhisk". It encourages users to "Run your application code without servers, scale it automatically, and pay nothing when it's not in use." Below this, there are two buttons: "Download CLI" and "Start Creating". To the right, there's a large circular icon with a stylized green "f" inside, surrounded by various icons representing different functions like triggers and sequences. Below the icon, there's a section titled "What's New" with a bulleted list: "IAM enablement", "Namespaces can now be explicitly managed and show up on the dashboard", "Manage Namespace Settings", and a link to "View release notes". At the bottom, there's a section titled "Save costs, scale and integrate." with three small icons illustrating integration concepts.

From the Functions main panel, click on Start Creating.



The screenshot shows the "Create" dialog box. The sidebar on the left remains the same as the main panel. The main area is titled "Create" and has a section titled "Single entities" with four options: "Action", "Trigger", "Sequence", and "Starters". Each option has a brief description and a corresponding icon. Below these are "Starters" and "Installable Packages" sections. At the bottom of the dialog is a "Cancel" button.

Here, select Actions.



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 function is created, click on code tab.

```

1 /**
2  * @param {object} params
3  * @param {string} params.lan_apkey
4  * @param {string} params.url
5  * @param {string} params.username
6  * @param {string} params.password
7  * @param {string} params.collection_id
8  * @param {string} params.configuration_id
9  * @param {string} params.input
10 */
11
12 /**
13  * @return {object}
14 */
15
16
17 const assert = require('assert');
18 const DiscoveryV1 = require('watson-developer-cloud/discovery/v1');
19
20 /**
21  * main() will be run when you invoke this action
22  *
23  * @param Cloud Functions actions accept a single parameter, which must be a JSON object.
24  * @param {Object} params
25  * @return The output of this action, which must be a JSON object.
26  */
27
28 function main(params) {
29  return new Promise(function (resolve, reject) {
30
31    let discovery;
32
33    if (params.lan_apkey) {
34
35      discovery = new DiscoveryV1({
36        auth: {
37          apkey: params.lan_apkey
38        }
39      });
40
41      discovery
42        .collection(params.collection_id)
43        .configuration(params.configuration_id)
44        .url(params.url)
45        .username(params.username)
46        .password(params.password)
47        .lanApKey(params.lan_apkey)
48        .get()
49        .then(function (response) {
50          resolve(response);
51        })
52        .catch(function (err) {
53          reject(err);
54        });
55    }
56  }
57}

```

In the code editor window [1], cut and paste in the code from the disco-action.js file found in the actions directory of your local repository. The code is pretty straight-forward - it simply connects to the Discovery service, makes a query against the collection, then returns the response. Now if click the invoke [2] button, it will fail due to credentials not being defined yet. To solve this, select parameter[1] tab

The screenshot shows the IBM Cloud Functions interface. The top navigation bar includes 'IBM Cloud', a search bar, and links for Catalog, Docs, Support, Manage, and Hashim Irfan Ali... The user is in the 'Actions' section under 'disco_action'. A large red number '1' is overlaid on the left sidebar. The main content area shows the 'Parameters' tab selected, displaying four parameters: 'url' (value: "https://api.eu-gb.discovery.watson.cloud.ibm.com/instances/406633d2-e1a8-474"), 'environment_id' (value: "d...45"), 'collection_id' (value: "3...5f"), and 'iam_apikey' (value: "c...88"). The right side of the interface shows a 'Namespace: Namespace-W25(Dallas)' header and a 'Logs' section.

Add the following keys: • url • environment_id • collection_id • iam_apikey. For the above values, please use the values associated with the Discovery service you created in the previous step. Enclose your values in double quotes. Now that the credentials are set, return to the Code panel tab and press the Invoke button again. Now you should see actual results returned from the Discovery service.

IBM Cloud

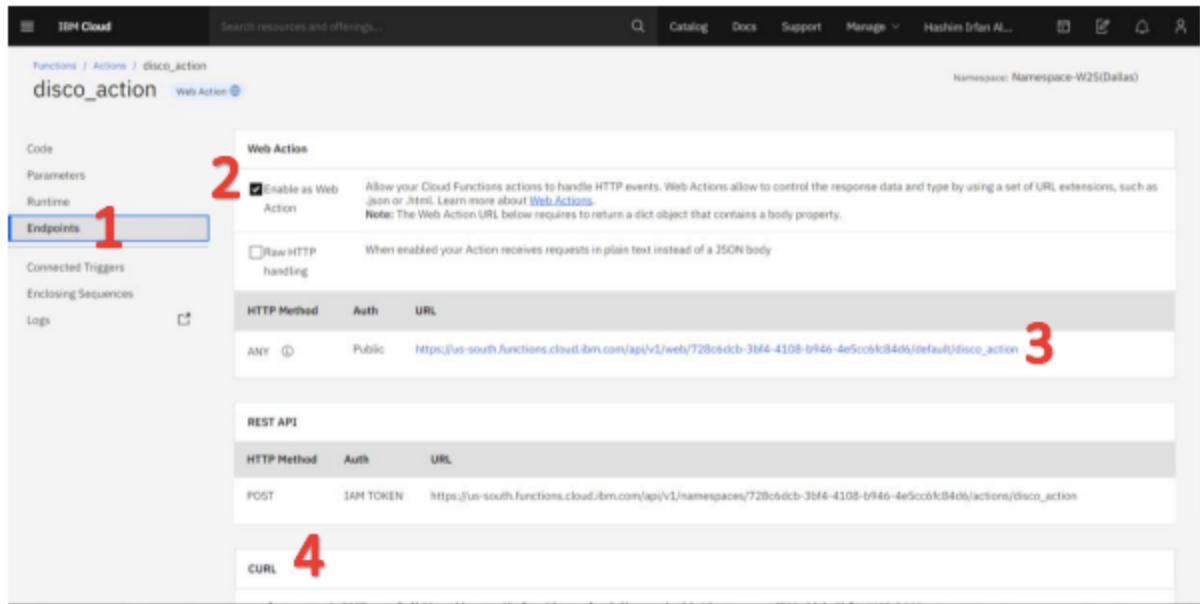
Search resources and offerings...

Catalog Docs Support Manage Hashim Irfan Ali

Functions Actions disco_action Namespace: Namespace-W2S(Dallas)

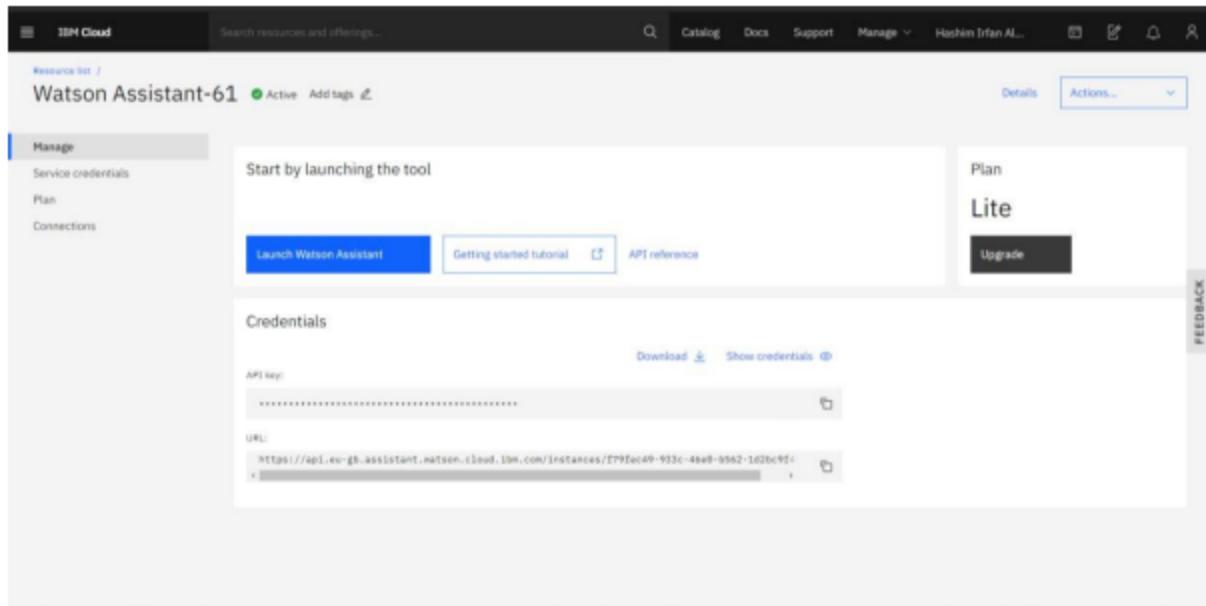
disco_action Web Action

Code	Code Node.js 10	Invoke with parameters	Invoke	Activations	Collapse	Clear
Code	<pre>1 // ... 2 * 3 * @param [object] params 4 * @param [string] params.lae_apikey 5 * @param [string] params.url 6 * @param [string] params.environment 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 * main() will be run when you invoke this action 22 * 23 * @param Cloud Functions actions accept a single parameter, which must be a JSON object. 24 * @param {Object} context 25 * @return The output of this action, which must be a JSON object. 26 */ 27 28 */ 29 * @function main[params] 30 * @return new Promise(function (resolve, reject) { 31 * let discovery; 32 * 33 * if (params.lae_apikey) { 34 *</pre>			Activations		
Parameters				disco_action 1450 ms 5/16/2020, 22:24:23		
Runtime				Activation ID: 5f41		
Endpoints				Results:		
Connected Triggers				<pre>{ "matching_results": 132, "passages": [], "results": [{ "enriched_text": { "categories": [{ "label": "/business and industrial/energy", "score": 0.6953 }, { "label": "/business and industrial/green solutions", "score": 0.643533 }, { "label": "/business and industrial/business operations", "score": 0.633224 }], "concepts": [{ "label": "... }] } }] }</pre>		
Enclosing Sequences						
Logs						

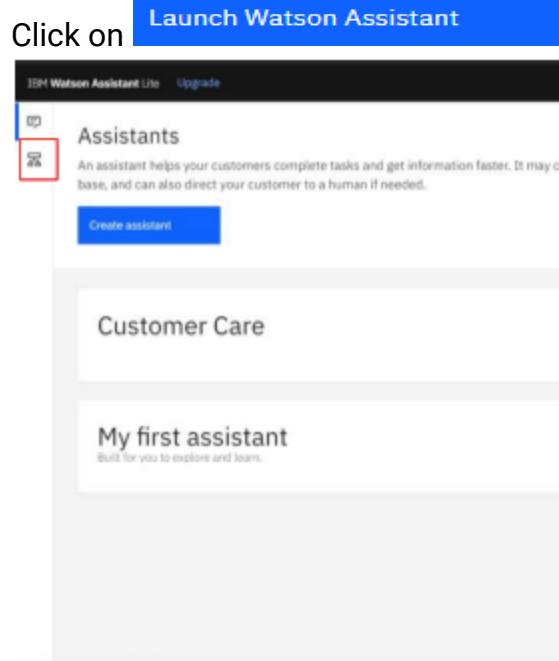


Now, click on endpoints[1] tab. Click the checkbox for Enable as Web Action [2]. This will generate a public endpoint URL [3]. Take note of the URL value [3], as this will be needed by Watson Assistant in a future step. To verify you have entered the correct Discovery parameters, execute the provided curl command [4]. If it fails, re-check your parameter values. [An IBM Cloud Functions service will not show up in your dashboard resource list. To return to your defined Action, you will need to access Cloud Functions by selecting Create Resource from the main dashboard panel (as shown at the beginning of this step).]

5. Configure Watson Assistant - Go back to the IBM Dashboard from the resource list screen. click to open Watson Assistant service.



The screenshot shows the IBM Cloud interface for managing a Watson Assistant service named "Watson Assistant-61". The top navigation bar includes "IBM Cloud", a search bar, and links for Catalog, Docs, Support, Manage, and a user profile. The main content area displays the service details: "Watson Assistant-61" (Active), "Add tags", "Details", and "Actions...". A "Manage" tab is selected, showing options for Service credentials, Plan, and Connections. The "Plan" section indicates a "Lite" plan with a "Upgrade" button. Below this, the "Credentials" section shows an API key and URL. A large blue button labeled "Launch Watson Assistant" is prominently displayed.



The screenshot shows the "Watson Assistant" service dashboard. At the top, there's a header with "IBM Watson Assistant Lite", "Upgrade", and other navigation icons. The main content area is titled "Assistants". It features a "Customer Care" card with a "Skills (1)" section containing "Customer Care Sample Skill" and an "Integrations (1)" section with a link. Below it is a "My first assistant" card with a "Skills (1)" section containing "My first skill" and an "Integrations (0)" section. A blue circular icon with a question mark is located in the bottom right corner of the dashboard area.

Click on **Launch Watson Assistant**

Click on the skills tab.

The screenshot shows the 'Skills' section of the IBM Watson Assistant interface. At the top, there's a 'Create skill' button highlighted with a red box. Below it, two skills are listed:

- Customer Care Sample Skill**
 - Type: Dialog – English (US)
 - Created: May 1, 2020 3:18 PM IST
 - Updated: May 12, 2020 2:52 PM IST
 - Linked Assistants: Customer Care (1)
- My first skill**
 - Type: Dialog – English (US)
 - Created: May 1, 2020 3:18 PM IST
 - Updated: May 1, 2020 3:18 PM IST
 - Linked Assistants: My first assistant (1)

Click Create Skill

The screenshot shows the 'Create a skill' wizard in progress. The current step is 'Select skill type'. It offers two options:

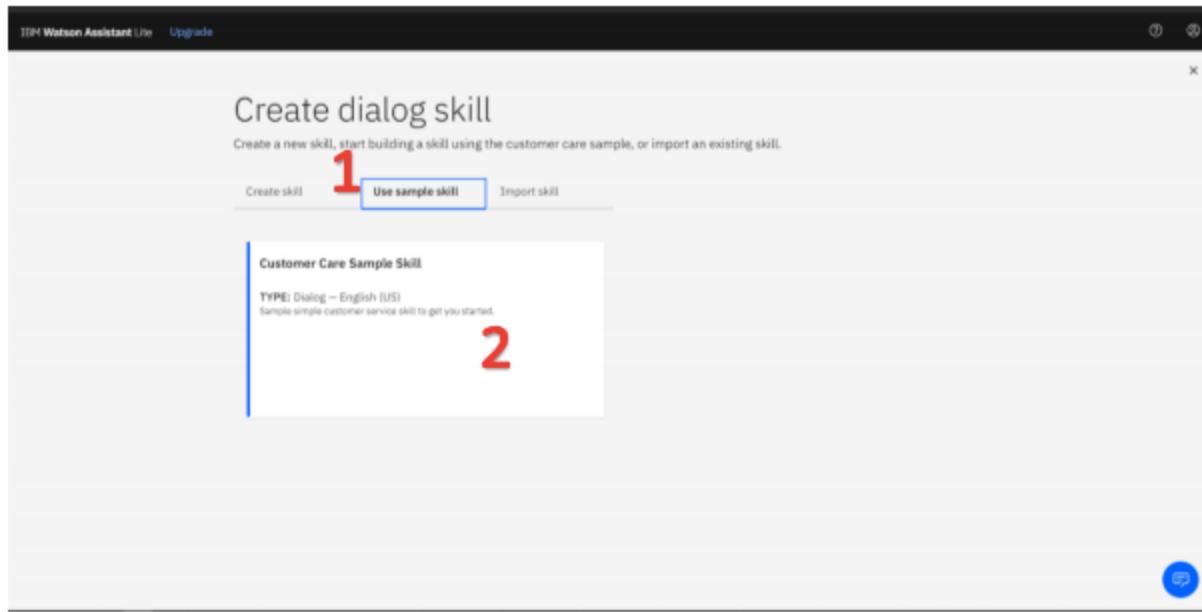
- Dialog skill**

Dialog flows are designed to address customer issues. Dialog skills expose the mechanics involved in natural language processing and responding appropriately to customers. [Learn more](#)
- Search skill**

For customer questions or requests that require lengthy or complex responses, add a search skill. It can access your existing self-service content to find information to share with your customers. [Learn more](#)

A 'Try Plus plan' button is visible between the two options. At the bottom, there's a 'Next' button.

Select Dialog Skill Card and Click next.



Select Use Sample Skill [1] and select Customer Care Sample Skill [2]. This dialog skill contains all of the nodes needed to have a typical call centre conversation with a user.

<input type="checkbox"/>	Intents (10) ↑	Description	Modified ↑	Examples ↑
<input type="checkbox"/>	#Cancel	Cancel the current request	15 days ago	7
<input type="checkbox"/>	#Customer_Care_Appointments	Schedule or manage an in-store appointment.	15 days ago	20
<input type="checkbox"/>	#Customer_Care_Store_Hours	Find business hours.	15 days ago	48
<input type="checkbox"/>	#Customer_Care_Store_Location	Locate a physical store location or an address.	15 days ago	25
<input type="checkbox"/>	#General_Connect_to_Agent	Request a human agent.	15 days ago	47
<input type="checkbox"/>	#General_Greetings	Greetings	15 days ago	30
<input type="checkbox"/>	#Goodbye	Good bye	15 days ago	6
<input type="checkbox"/>	#Help	Ask for help	15 days ago	8
<input type="checkbox"/>	#Product_Information		4 days ago	3

As the default customer care dialog does not have a way to deal with any questions involving outside resources, so we will need to add new intent. 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 at a minimum, enter the following example

questions to be associated with it.

The screenshot shows the IBM Watson Assistant interface for creating a new intent. The intent name is '#Product_Information'. Under the 'User example' section, there are three examples listed:

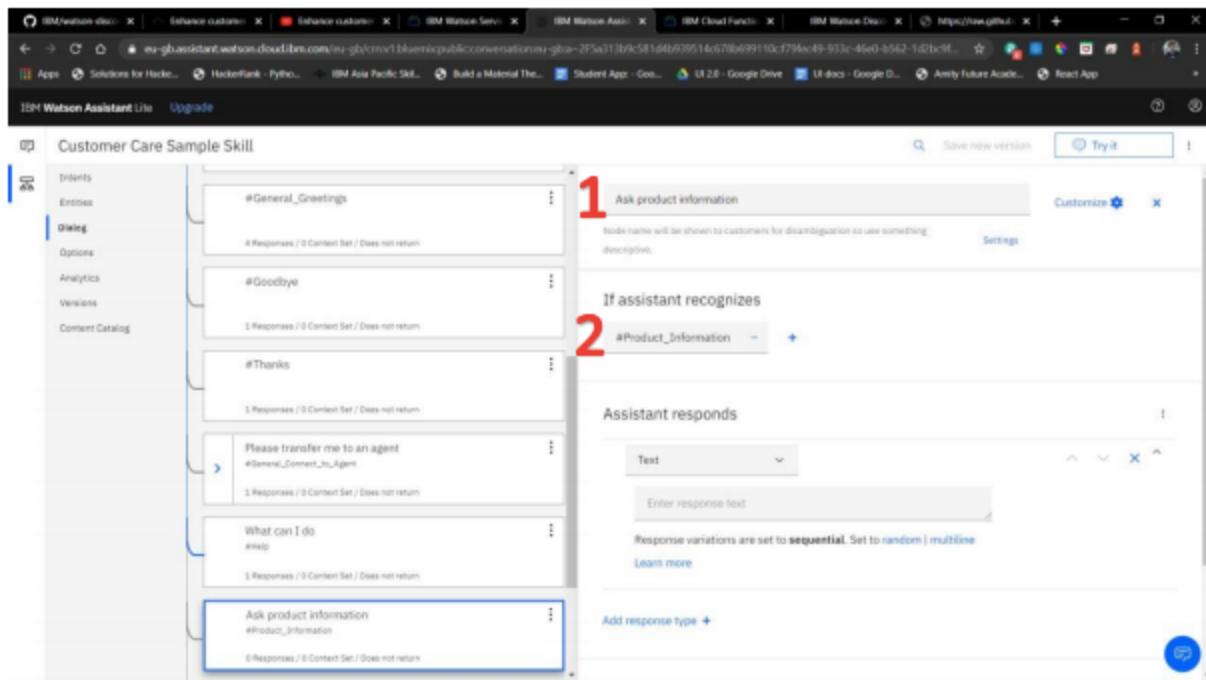
- How do I access the settings
- How do I set the time
- Type a user example here, e.g. I want to pay my credit card bill.

At the bottom, there are buttons for 'Add example' and 'Show recommendations'.

Go back to the previous page after doing this, then click on Dialog Tab and add a node below "What can I do node".

The screenshot shows the 'Dialog' tab in the IBM Watson Assistant interface for a 'Customer Care Sample Skill'. The dialog tree is as follows:

- Opening: welcome
 - 1 Response / 1 Context Set / Does not return
- What are your hours?
 - #Customer_Care_Store_Hours
 - 0 Responses / 0 Context Set / Jump to / Does not return
- Where are you located?
 - #Customer_Care_Store_Location
 - 3 Responses / 0 Context Set / Skip user input / Does not return
- I want to make an appointment
 - #Customer_Care_Appointments
 - 3 Responses / 1 Context Set / 5 Stars / Does not return
- #General_Greetings
 - 4 Responses / 0 Context Set / Does not return
- #About_us
 - 0 Responses / 0 Context Set / Does not return



Name the node "Ask product information" [1] and assign it our new intent #Product_Information [2]. 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. Before proceeding further, let's learn about webhook. A webhook is a mechanism that allows you to call out to an external program based on something happening in your 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 you specify or that you collect from the user during the conversation and save in context variables, and sends the data to the Webhook request URL as an HTTP POST request. 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.

The screenshot shows the IBM Watson Assistant interface with the 'Customer Care Sample Skill' selected. The left sidebar has tabs for Entities, Dialog, Options, and Webhooks, with 'Webhooks' currently selected (marked with a red '1'). The main content area is titled 'Webhooks' and describes it as a mechanism to call external programs. It includes a 'Webhook setup' section with a URL input field containing 'https://us-south.functions.cloud.ibm.com/api/v1/web/728c6dc8-3bf4-410e'. Below this is a 'Headers' section for adding HTTP headers. A 'Next step' note at the bottom suggests enabling webhooks from a dialog node. A 'Try it' button is in the top right.

Click Webhooks[1] tab and enter the URL[2] to enable web hook for the IBM Cloud Functions action you created in Step 4. Add .json to the end of the URL to specify the result should be in JSON format. Return to the Dialog tab, and click on the Ask product information node. From the details panel for the node, click on Customize, and enable Webhooks for this node.

The screenshot shows the 'Customer Care Sample Skill' dialog flow with the 'Ask product information' node selected. A customization modal is open over the dialog nodes. In the 'Webhooks' section of the modal, there is a green success message: 'Webhook URL Your webhook URL is configured.' A 'Settings' tab is visible in the background. The 'Apply' button is highlighted in blue at the bottom of the modal.

Click Apply. The dialog node should have a Return variable [1] set automatically to

\$webhook_result_1. This is the variable name you can use to access the result from the Discovery service query.

The screenshot shows the IBM Watson Assistant interface for a "Customer Care Sample Skill". On the left, the skill's navigation menu includes Intents, Entities, Dialog, Options, Analytics, Versions, and Content Catalog. The main workspace displays a flowchart of intents: #Goodbye, #Thanks, Please transfer me to an agent, What can I do, anything_else, Ask product information (@Product_Information), and another Ask product information node. The second Ask product information node is highlighted with a blue selection bar. A red box labeled '1' points to the "Return variable" field in the configuration panel on the right, which contains "webhook_result_1". Another red box labeled '2' points to the "Value" field, which contains "<?Input.text?>". The configuration panel also includes sections for "Ask product information" (with a note about node names) and "Then callout to my webhook" (with parameters for "Key" (input) and "Value" (<?Input.text?>)). A success message at the bottom states "Webhook URL Your webhook URL is configured. Options".

You will also need to pass in the users question via the parameter input [2]. The key needs to be set to the value: "" If you fail to do this, Discovery will return results based on a blank query. Optionally, you can add these responses to aid in debugging:

This screenshot shows the same "Customer Care Sample Skill" interface. The flowchart and the configuration of the "Ask product information" node are identical to the previous screenshot. However, the configuration panel on the right now includes an "Assistant responds" section. It lists two entries under "If assistant recognizes": 1. \$webhook_result_1, which is associated with a response of "\$webhook_result_1" and has a red box around it; 2. anything_else, which is associated with a response of "Try again later". The "Respond with" column contains the variable names for each response. The rest of the configuration panel remains the same, including the "Ask product information" section and the "Then callout to my webhook" section.

Now we should test the assistant by clicking on Try it button located at the top right side of the panel.

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.

For upcoming steps, you will need to provide some credentials to access your assistant so to store credentials for future use follow these steps below. Go back to the skills tab,

click [1] and then [2]

The screenshot shows the 'Skills' section of the IBM Watson Assistant interface. There are two skills listed: 'Customer Care Sample Skill' and 'My first skill'. The 'Customer Care Sample Skill' has a context menu open, with the second item in the list highlighted. The menu items are: View API Details, Export, Duplicate, Rename, and Delete.

The Skill ID and API Key is to be noted.

The screenshot shows the 'Skill details' modal for the 'Customer Care Sample Skill'. It displays the skill name, skill ID (617d5db3-c85f-47b5-8f90-35db631250aa), and the legacy v1 workspace URL. The 'Skill ID' field is highlighted with a red box. Below it, the 'Service credentials' section shows the service credentials name and API key, both of which are highlighted with red boxes.

Go Back to the Watson Assistant Resource List, Select Service Credentials [1] and make note of the URL.APIKEY can be found here too.

The screenshot shows the IBM Cloud dashboard for a Watson Assistant resource named "Watson Assistant-61". The "Service credentials" tab is selected, indicated by a red number "1". A table lists service credentials, with one entry highlighted by a red box containing the URL: "url": "https://api.eu-gb.assistant.watson.cloud.ibm.com/instances/f79fec49-933c-46e0-b562-1d2bc9f497b5".

6. Build Node-RED Flow to Integrate All Services - Now it's time to create Node-Red, go to IBM Cloud Dashboard, click on Create Resource and search for node-red[1].

The screenshot shows the IBM Cloud catalog search results for "node-red". A red number "1" is above the search bar. A red box surrounds the "Node-RED App" tile, which is described as a Starter Kit for Node-RED.

Click on the Node-RED App tile [2]. This will show you an overview of the Starter Kit and what it provides.

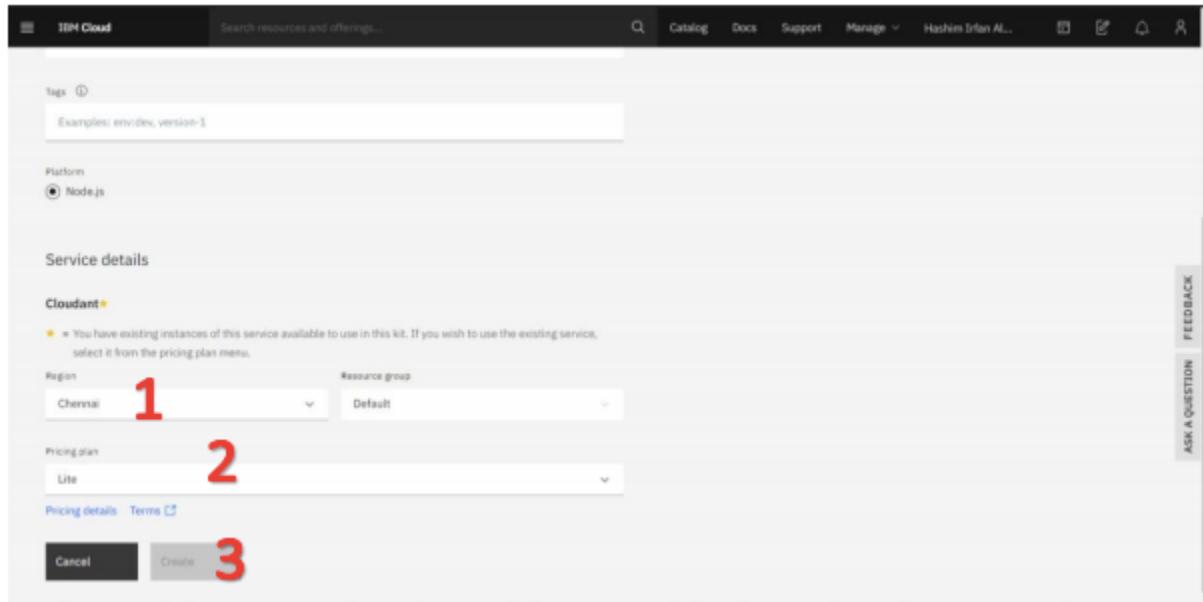
The screenshot shows the IBM Cloud Catalog interface. At the top, there's a search bar and navigation links for Catalog, Docs, Support, Manage, and user profile. Below the search bar, the URL is Catalog / Create app / Node-RED. The main content area displays the 'Node-RED' starter kit. A large red number '1' is overlaid on the 'Create' button. The 'Create' button is highlighted in blue. To the left of the 'Create' button is a 'About' tab. The 'Overview' section contains details about the starter kit, including its author (IBM), updated date (2/11/2020), type (Starter Kit), and source code link (GitHub). It also lists helpful links for tutorials and provides a brief overview of what the starter kit offers. The 'What's included?' section lists the Cloudant service, which is described as free to start and has a 'View pricing' link. A 'Get started' button is located at the bottom of this section. On the right side of the page, there are 'ASK A QUESTION' and 'FEEDBACK' buttons.

Click on Create [1].

The screenshot shows the IBM Cloud Catalog interface, similar to the previous one but on a different page. The URL is Catalog / Create app / Node-RED. The main content area is titled 'App details'. A large red number '1' is overlaid on the 'App name' input field. The input field contains the text 'Node RED KVHBI'. Below the input field are sections for 'Resource group' (set to 'Default'), 'Tags' (with an example 'env:dev, version-1'), and 'Platform' (set to 'Node.js'). At the bottom of the page, there's a 'Service details' section. On the right side, there are 'ASK A QUESTION' and 'FEEDBACK' buttons.

Now you need to configure the Node-RED Starter application. On the App details page, a randomly generated name will be suggested – Node RED KVHBI in the screenshot above. Either accept that default name or provide a unique name for your application [1]. This will become part of the application URL. Note: If the name is not unique, you will see an error message and you must enter a different name before you continue. The Node-RED Starter application requires an instance of the Cloudant database service to store your application flow configuration. To do this, Select the region [1] the service

should be created in and what pricing plan it should use. You can only have one Cloudant instance using the Lite plan and you can have more than one Node-RED Starter application using the same Cloudant service instance. If you have already got an instance, you will be able to select it from the Pricing plan select box [2]. Click the Create button [3] to continue. This will create your application, but it is not yet deployed to IBM Cloud.



At this point, you have created the application and the resources it requires, but you have not deployed it anywhere to run, so this step shows how to setup the Continuous Delivery feature that will deploy your application into the Cloud Foundry space of IBM Cloud. Click on Deploy your App[1].

Deployment Automation

Configure Continuous Delivery
Continuous Delivery is not enabled for this app. Enable Continuous Delivery to automate builds, tests, and deployments through Delivery Pipeline, GitLab, and more.

Deploy your app **1**

Services (1)

Name	Resources	Actions
Cloudant	Documentation	

Credentials

```
{
  "cloudant": [
    {
      "name": "node-red-kvhbi-cloudant-1599811961473",
      "connection": {
        "url": "https://091362bd-012a-4322-8bae-f9e7bc4a396-blaemix.cloudant.com",
        "password": "*****"
      }
    }
  ]
}
```

You will need to create an IBM Cloud API key to allow the deployment process to access your resources. Click the New button (1) to create the key. A message dialog will appear. Read what it says and then confirm and close the dialog.

Select the deployment target Configure the DevOps toolchain

Deployment Automation

Select your deployment target and configure your DevOps toolchain. After you click **Create**, the toolchain is created, and the deployment process is started automatically.

Deployment target

Cloud Foundry
IBM
Deploy and run your applications without managing servers or clusters. A live plan is available for quick and easy deployment.

IBM Cloud API key
IBM Cloud API key **New** **1**
The value is required.

Getting started with apps

Step 1. Select the deployment target
Select your deployment target, and then provide the configuration information.

IBM Cloud Foundry
Cloud Foundry is the premier industry standard Platform-as-a-Service (PaaS) that ensures fast, easy, and reliable deployment of cloud-native apps. Cloud Foundry ensures that the build and deploy aspects of coding remain carefully coordinated with any attached services – resulting in quick, consistent and reliable iteration of applications. Cloud Foundry has a live plan that allows quick deployments for testing purposes.

Before you begin

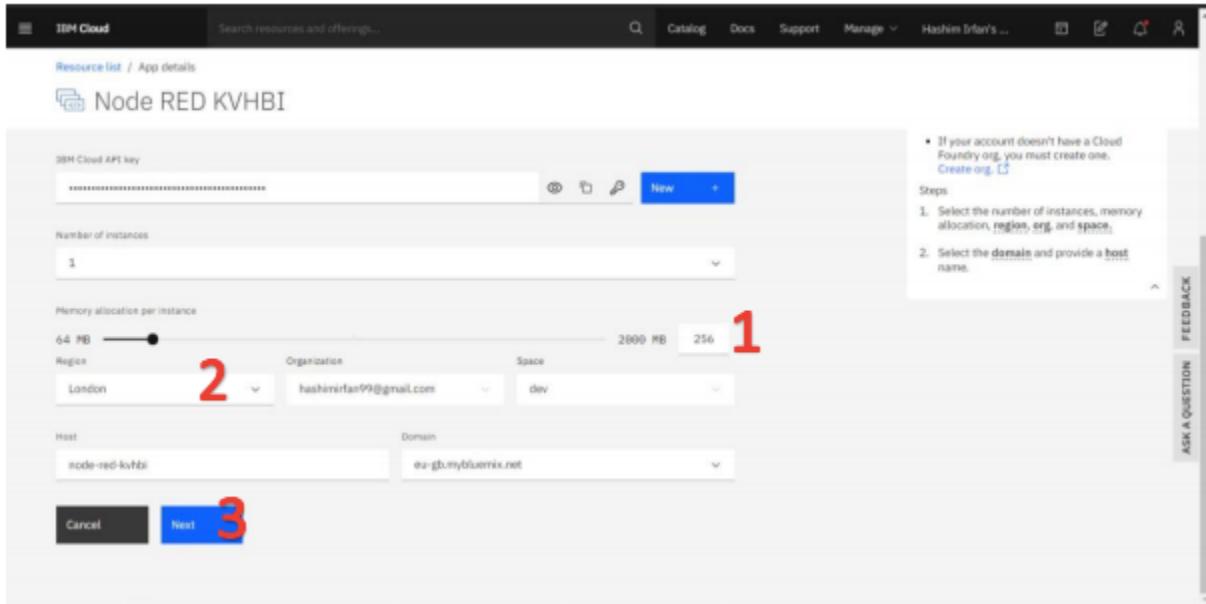
- If your account doesn't have a Cloud Foundry org, you must create one. [Create org.](#) **1**

Steps

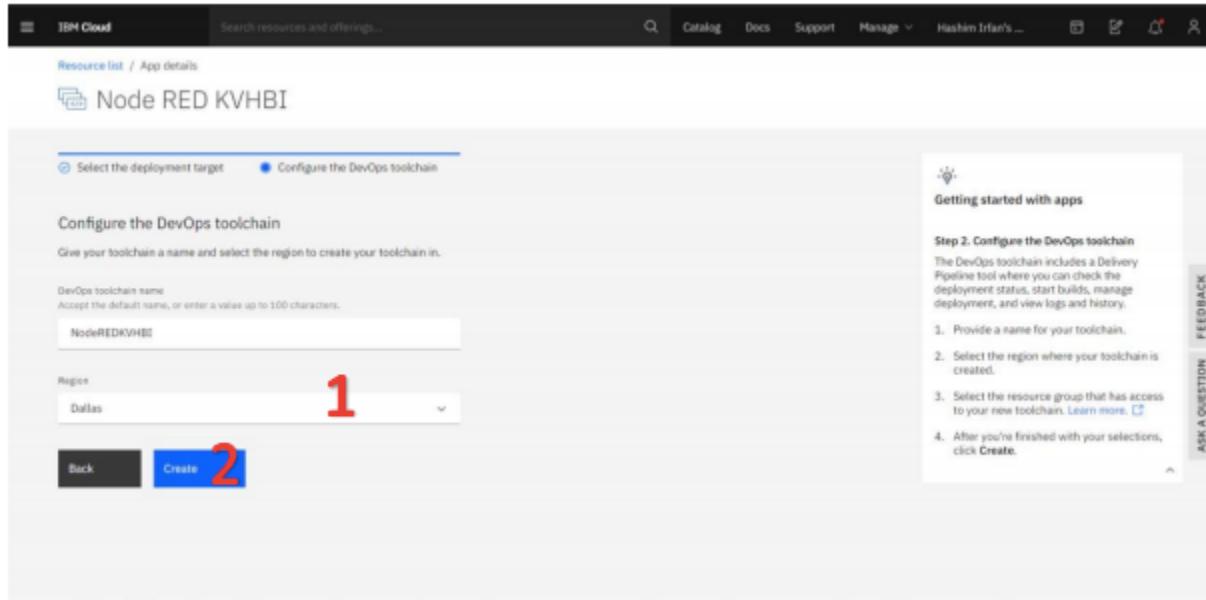
- Select the number of instances, memory allocation, [region](#), [org](#), and [space](#).

After creating the API Key, Increase the Memory allocation per instance slider [1] to 256MB. If you do not increase the memory allocation, your Node-RED application might not have sufficient memory to run successfully. The Node-RED Starter kit only supports deployment to the Cloud Foundry space of IBM Cloud. Select the region [2] to deploy your application to. This should match the region you created your Cloudant instance

in. Click Next [3].



Now, select the region [1] to create the DevOps toolchain and then Click Create [2].



This will take you back to the application details page.

IBM Cloud

Resource list / App details / Node RED KVHBI Add tags

Actions...

Details

App URL: You must deploy your app first

Source: Download code

Resource group: Default

Deployment target: You must deploy your app first

Created: 5/18/2020

Services (1)

Cloudant Documentation

Connect existing services Create service

Credentials

```
{ "cloudant": { "name": "node-red-kvhbi--cloudant-1899811961AT3", "credentials": { "apikey": "*****", "host": "0913e23d-012a-4122-88aa-7ecada396-bluemix.clo", "password": "*****", "port": 443, "username": "node-red-kvhbi--cloudant-1899811961AT3" } } }
```

Deployment Automation

Configure Continuous Delivery

Continuous Delivery is not enabled for this app. Enable Continuous Delivery to automate builds, tests, and deployments through Delivery Pipeline, GitLab, and more.

Checking cache...

Feedback Ask a question

The Continuous Delivery section will refresh with the details of your newly created Toolchain. The Status field of the Delivery Pipeline will show "In progress". That means your application is still being built and deployed.

IBM Cloud

Resource list / App details / Node RED KVHBI Add tags

Actions...

Details

App URL: You must deploy your app first

Source: <https://us-south.git.cloud.ibm.com/hashimifar/f93/NodeREDKVHBI>

Resource group: Default

Deployment target: You must deploy your app first

Created: 5/18/2020

Services (1)

Cloudant Documentation

Connect existing services Create service

Credentials

```
{ "cloudant": { "name": "node-red-kvhbi--cloudant-1899811961AT3", "credentials": { "apikey": "*****", "host": "0913e23d-012a-4122-88aa-7ecada396-bluemix.clo", "password": "*****", "port": 443, "username": "node-red-kvhbi--cloudant-1899811961AT3" } } }
```

Deployment Automation

Name: NodeREDKVHBI

Location: Dallas

Tool integrations:

Delivery Pipelines

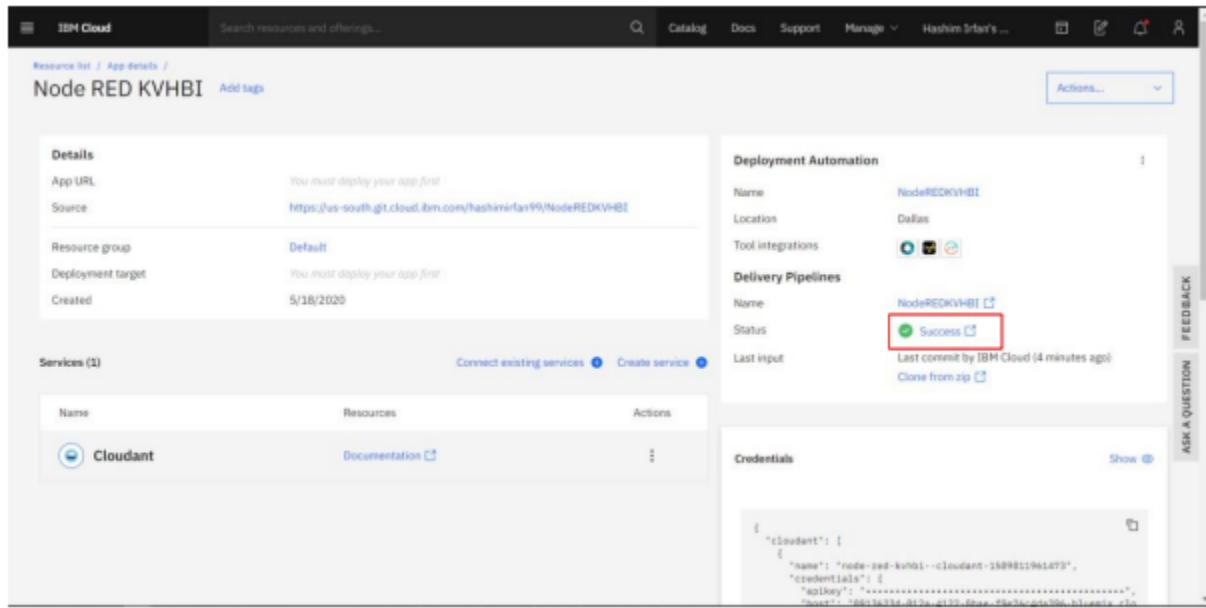
Name: NodeREDKVHBI

Status: In progress

Last commit by IBM Cloud (1 minute ago)
Clone from zip

Feedback Ask a question

The Deploy stage will take a few minutes to complete. Eventually the Deploy stage will go green to show it has passed. This means your Node-RED Starter application is now running.



The screenshot shows the IBM Cloud App Details page for an application named "Node RED KVHBI".

Details:

- App URL: <https://us-south-gt.cloud.ibm.com/hashmirfan99/NodeREDKVHBI>
- Source: <https://us-south-gt.cloud.ibm.com/hashmirfan99/NodeREDKVHBI>
- Resource group: Default
- Deployment target: You must deploy your app first
- Created: 5/18/2020

Services (1):

Name	Resources	Actions
Cloudant	Documentation	⋮

Deployment Automation:

- Name: NodeREDKVHBI
- Location: Dallas
- Tool integrations: GitHub, Jenkins, CircleCI

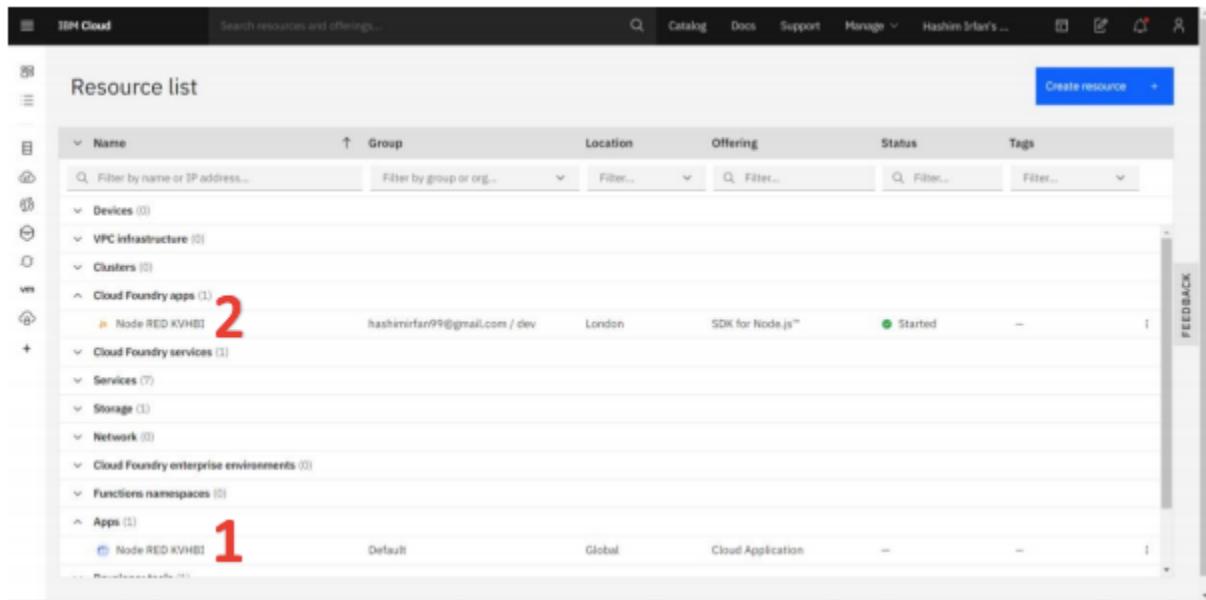
Delivery Pipelines:

- Name: NodeREDKVHBI
- Status: Success
- Last commit by IBM Cloud (4 minutes ago)
- Clone from zip

Credentials:

```
{ "cloudant": { "name": "node-red-kvhbi-cloudant-1b89811961473", "credentials": { "apikey": "*****", "host": "node-red-kvhbi-1b89811961473.firebaseio.com" } } }
```

Now that you've deployed your Node-RED application, let's open it up! Open your IBM Cloud Resource list. You will see your newly created Node-RED Application listed under the Apps section [1]. You will also see a corresponding entry under the Cloud Foundry apps section [2].



The screenshot shows the IBM Cloud Resource list.

Resource list:

Name	Group	Location	Offering	Status	Tags
Node RED KVHBI	hashmirfan99@gmail.com / dev	London	SDK for Node.js™	Started	-

Cloud Foundry apps:

- Node RED KVHBI [2]

Cloud Foundry services:

- Cloud Foundry services [1]

Click on this Cloud Foundry app entry to go to your deployed application's details page

The screenshot shows the IBM Cloud Node-RED Overview page for a resource named 'Node RED KVHBI'. The page has a left sidebar with links like 'Getting started', 'Overview' (which is selected), 'Runtime', 'Connections', 'Logs', 'APT Management', 'Autoscaling', and 'Availability Monitoring'. The main content area is divided into several sections: 'Instances' (Health: 100%, 1/1 instance(s) are running), 'Runtime' (SDK for Node.js™, a pie chart showing 256 Total MB allocation with 1.7% Free and 98.3% Used), 'Runtime cost' (\$0.00 current charges for billing period, \$0.00 estimated total for May 1, 2020 - May 31, 2020), and 'Connections (1)' (node-red-kvhbi--cloudant-1589811961473-32813). A red box labeled '1' highlights the 'Edit' link in the Runtime section, and another red box labeled '2' highlights the 'Visit App URL' link.

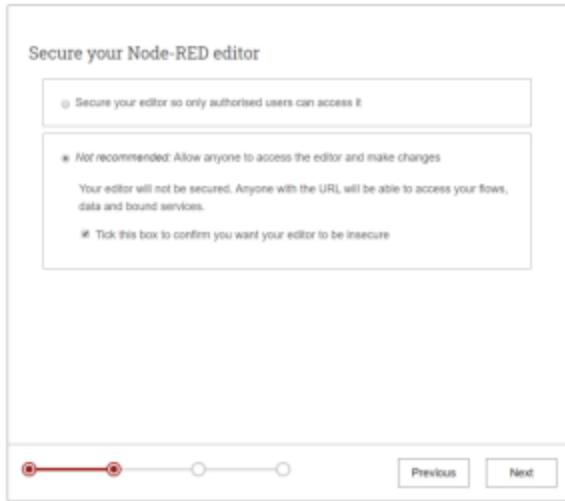
Special Cases: If your Runtime Instance is running full (0MB Free), Click on Edit[1] and reduce memory per instance to 128mb. If you have Free space on your runtime skip the previous step[1] and Click on Visit App URL[2].

The screenshot shows the welcome screen for a new Node-RED instance. It features a title 'Welcome to your new Node-RED instance on IBM Cloud' and a message: 'We know you're eager to start wiring up your flows, but first there are a couple of tasks you should do:' followed by a bulleted list: 'Secure your Node-RED editor' and 'Learn how to install additional nodes'. At the bottom, there is a progress bar with four steps, where the first step is highlighted in red, and buttons for 'Previous' and 'Next'.

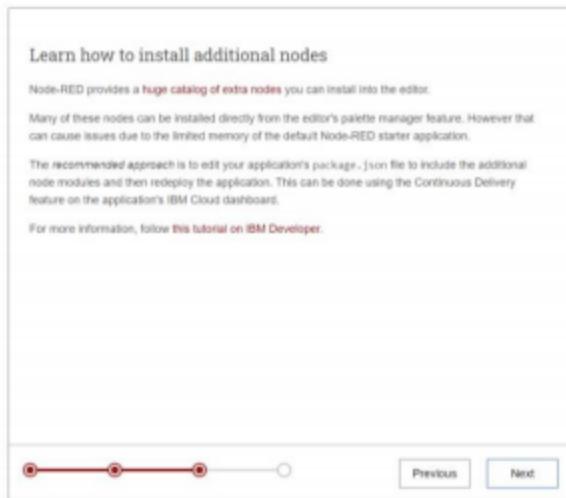
Click on Next button.



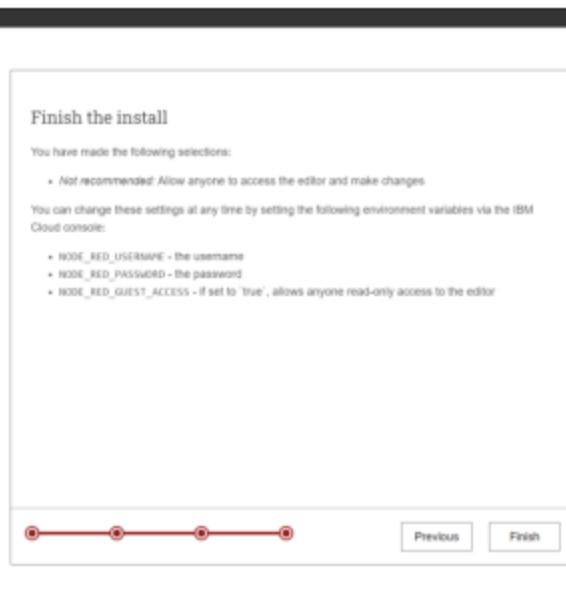
You can choose to secure your Node-RED editor by providing a username and password. I am selecting the other option which is Allow anyone to access the editor and make changes.



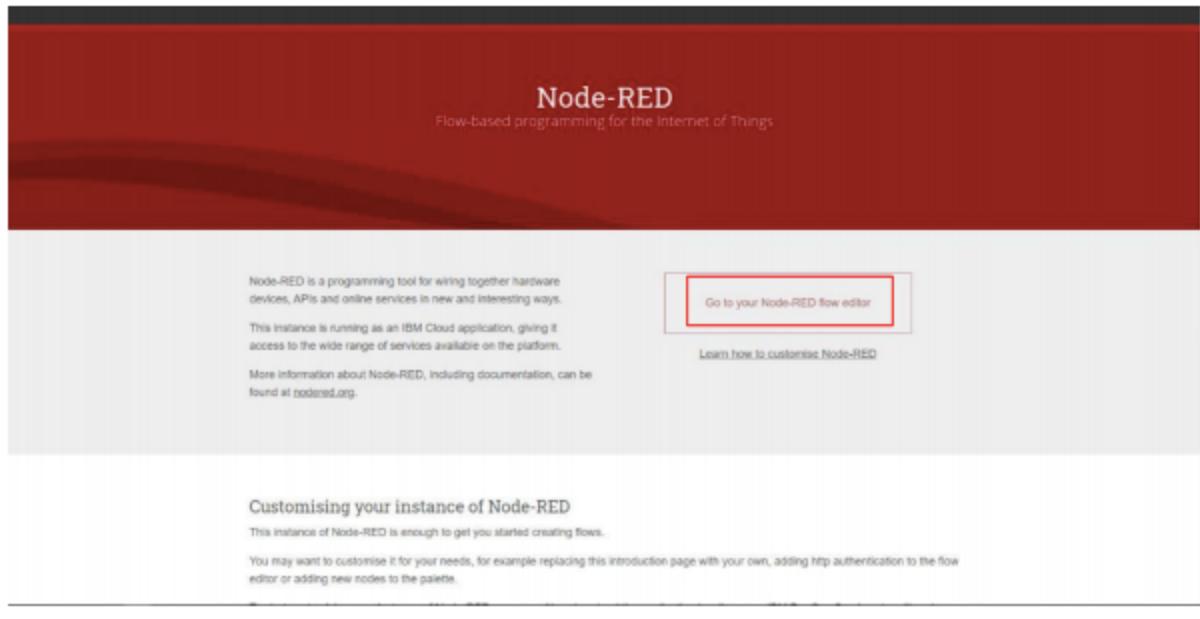
Tick the box, and click Next.



Click Next.

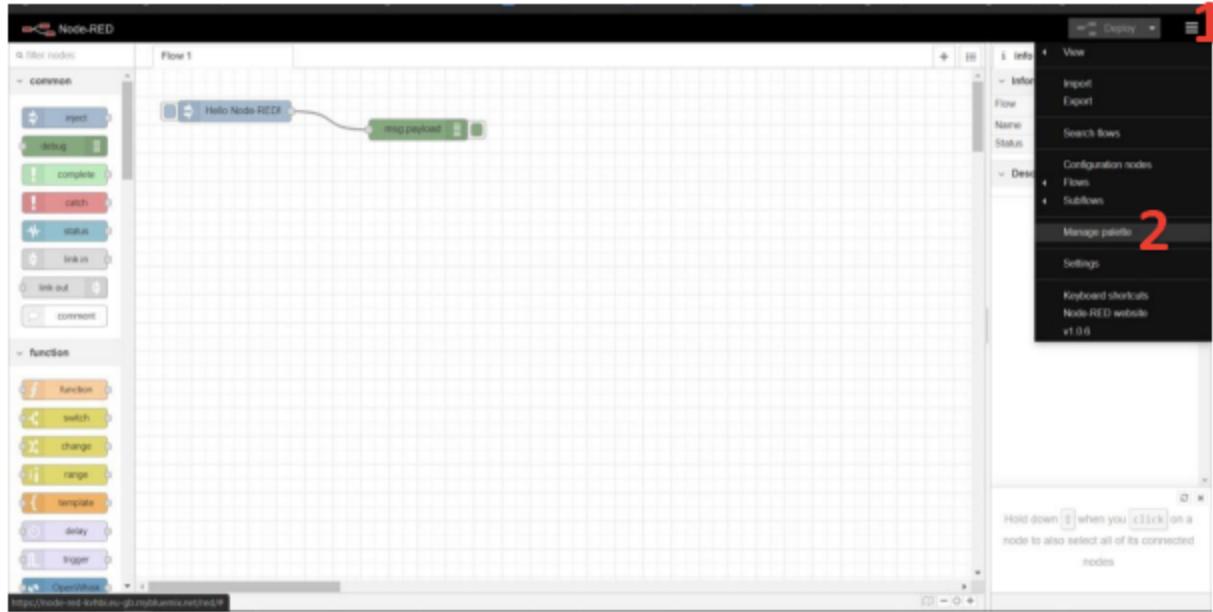


The final screen summarizes the options you've made and highlights the environment variables you can use to change the options in the future. Click Finish to proceed. Node-RED will save your changes and then load the main application. From here you can click the Go to your Node-RED flow editor button to open the editor.

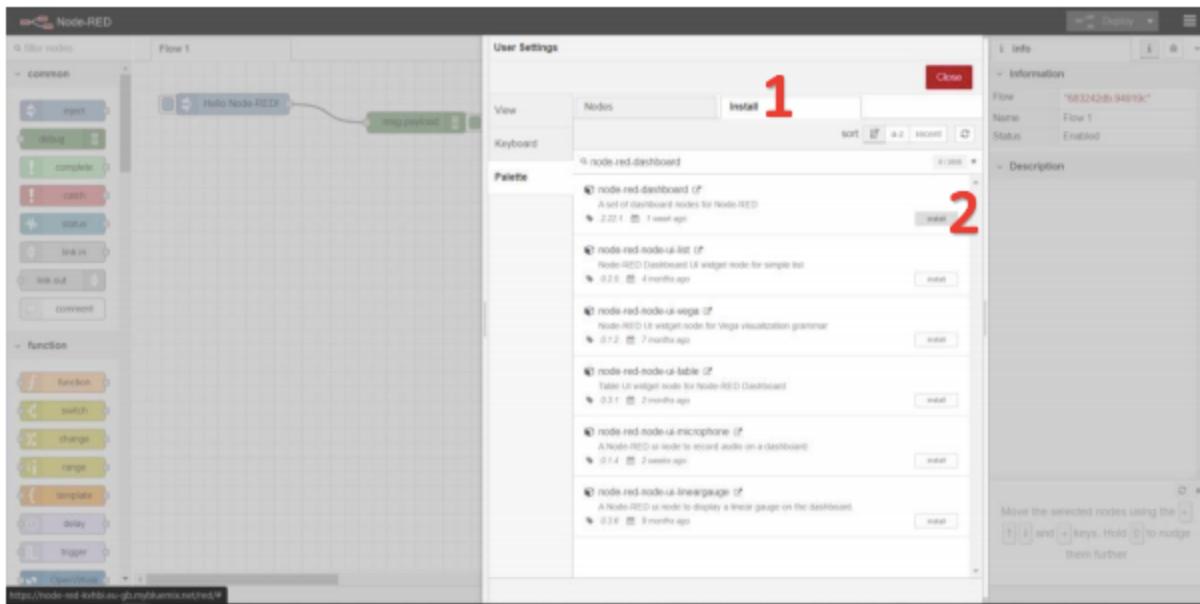


The Node-RED editor opens showing the default flow.

7. Configure the nodes and Build A Web Dashboard in Node-RED - To add Nodes to integrate Assistant, click [1] and then select Manage Palette [2].



Go to Install Tab [1] and search for node-red-dashboard and Install [2] it.

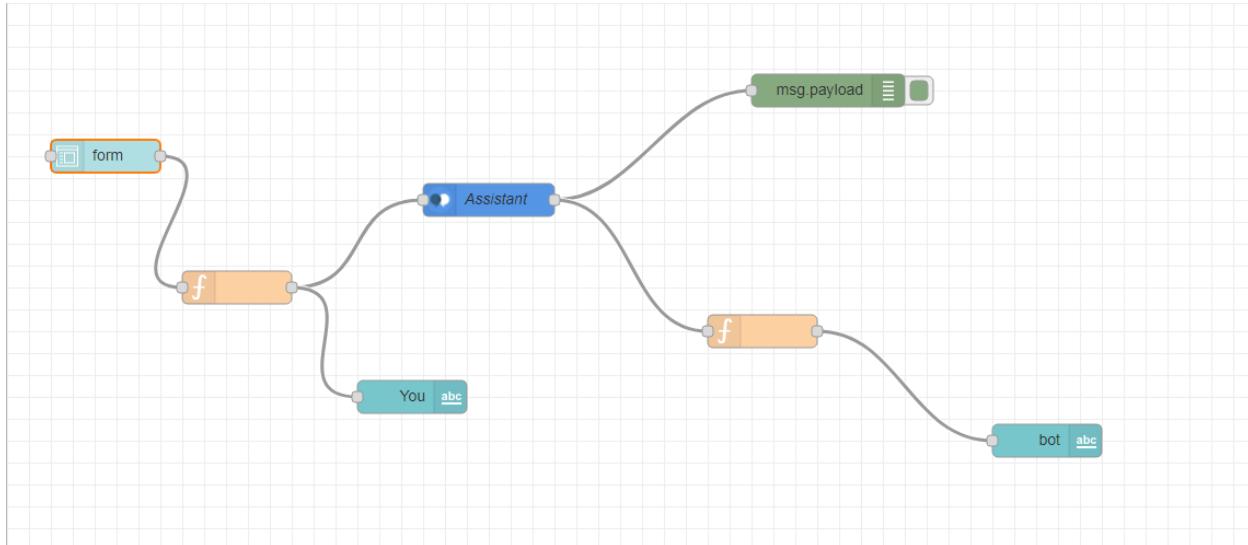


5. FLOWCHART

Create flow and configurenodenode:

At first go to manage pallete and install node-red-dashboard. Now, Create the flow with the help of following node:

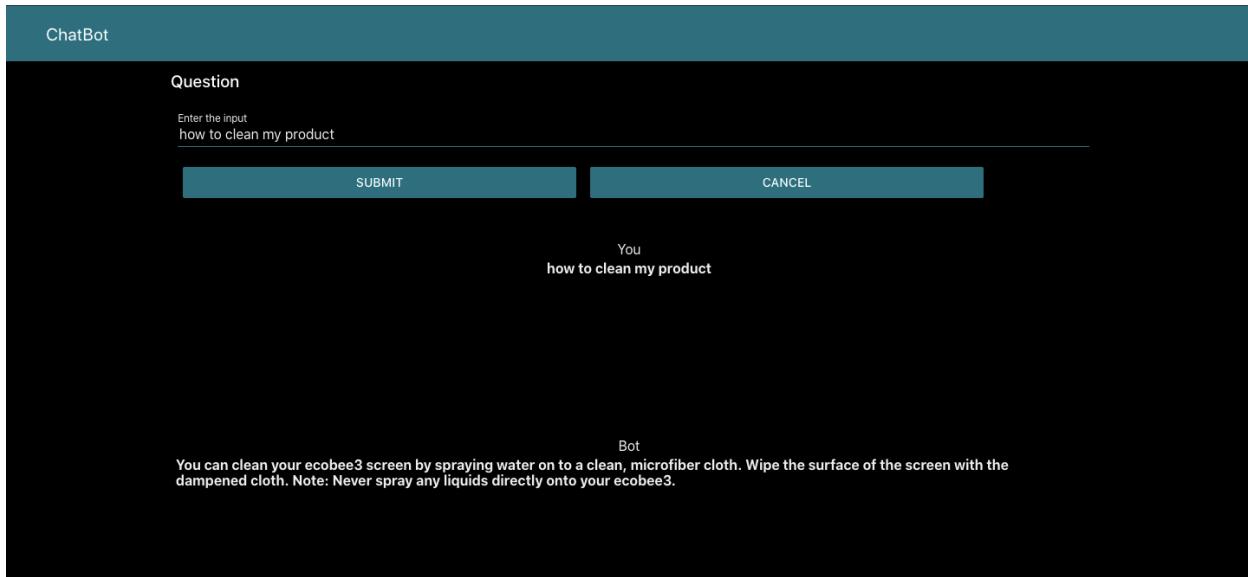
- Inject
- Assistant
- Debug
- Function
- Ui_Form
- Ui_Text



6. RESULT

Finally our Node-RED dash board integrates all the components and displayed in the Dashboard UI by typing URL-

https://node-red-gargsul.eu-gb.mybluemix.net/ui/#/0?socketid=ohIIRTBGxWEnr1tWAA_Bc



7. ADVANTAGES & DISADVANTAGES

Advantages:

- Companies can deploy chatbots to rectify simple and general human queries.
- Reduces man power.
- Cost efficient.
- No need to divert calls to customer agent and customer agent can look on other works.
- Faster Customer Service.
- Increased Customer Satisfaction.
- 24-7 availability.
- Multiple Customer Handling.

Disadvantages:

- Some times chatbot can mislead customers.
- Giving same answer for different sentiments.
- Some times cannot connect to customer sentiments and intentions.
- Limited Responses for Customers.
- Maintenance.

8. APPLICATIONS

- It can deploy in popular social media applications like facebook,slack,telegram.
- Chatbot can deploy any website to clarify basic doubts of viewers.

9. CONCLUSION

By doing the above procedure and all we successfully created Intelligent helpdesk smart chart-bot using Watson assistant, Watson discovery, Node-RED and cloud-functions.

10. FUTURE SCOPE

We can include Watson studio text to speech and speech to text services to access the chat-bot hands-free. This is one of the future scope of this project.

11. BIBILOGRAPHY

APPENDIX

A.Source Code (cloud function) -

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

B. Reference

- 1.<https://developer.ibm.com/patterns/enhance-customer-help-desk-with-smart-document-understanding/>
 - 2.<https://github.com/IBM/watson-discovery-sdu-with-assistant>
 - 3.<https://www.youtube.com/watch?v=-yniuX-Poyw&feature=youtu.be>
 - 4.<https://developer.ibm.com/tutorials/how-to-create-a-node-red-starter-application/>
- THE END