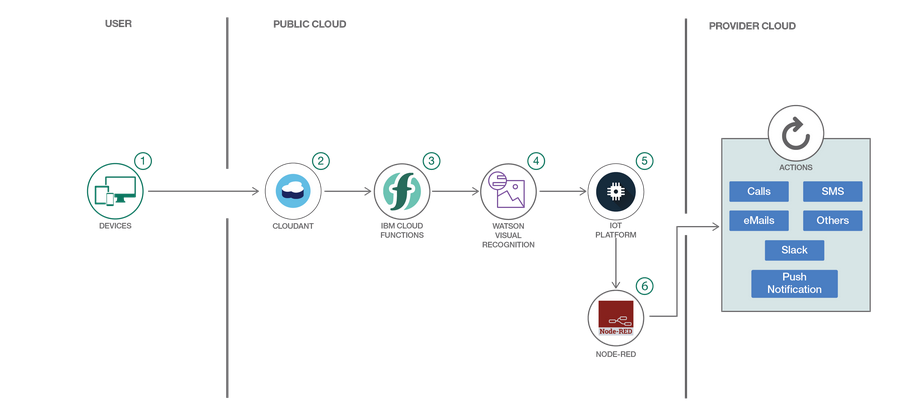
Code Pattern - https://github.com/IBM/image-analysis-iot-alert

Architecture diagram:



You will need the following pre-requisites and tools:

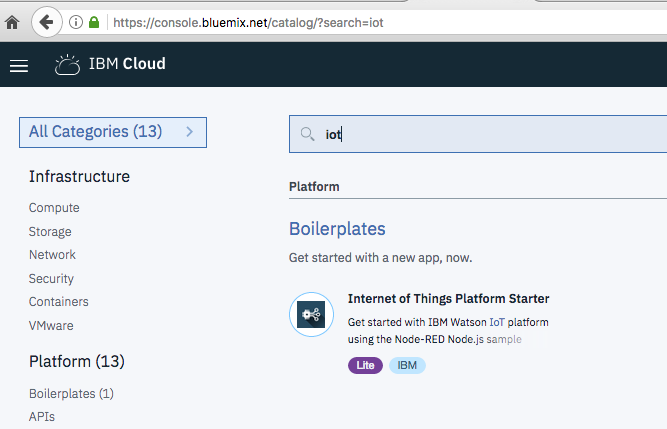
* [IBM Cloud account](https://console.ng.bluemix.net/registration/)
* [Bluemix CLI](https://console.bluemix.net/docs/cli/reference/bluemix_cli/index.html#getting-started)
* [Openwhisk CLI](https://github.com/apache/incubator-openwhisk-cli/releases)
* [nodejs](https://nodejs.org/en/download/) To verify you have nodejs installed run node -v at a command prompt/terminal.

Step1: Clone the github

git clone https://github.com/IBMDevConnect/iot-alert-image-analysis

Step2: Login to IBM Cloud

Search for IOT, you should see the following search result. Click on “Internet of Things Platform Starter”



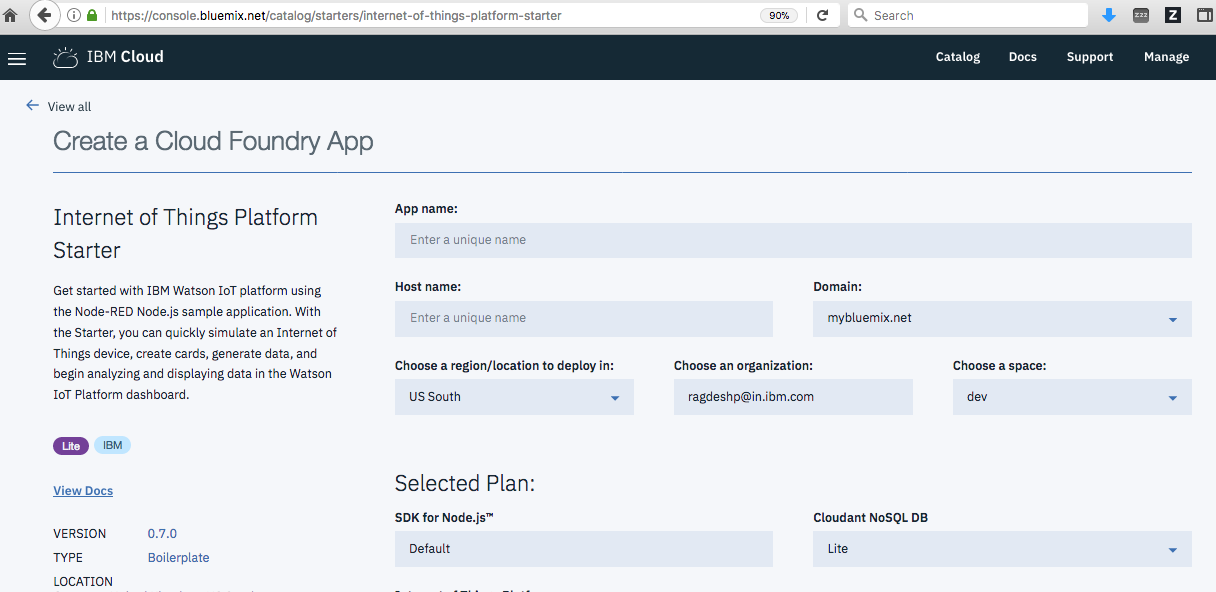
Step3: Next, you should see the following page, add the App name which should be unique.

Then click on ‘Create’. The Watson IOT platform service is created. Select the region, organization and space to the one where you have been mapped to.

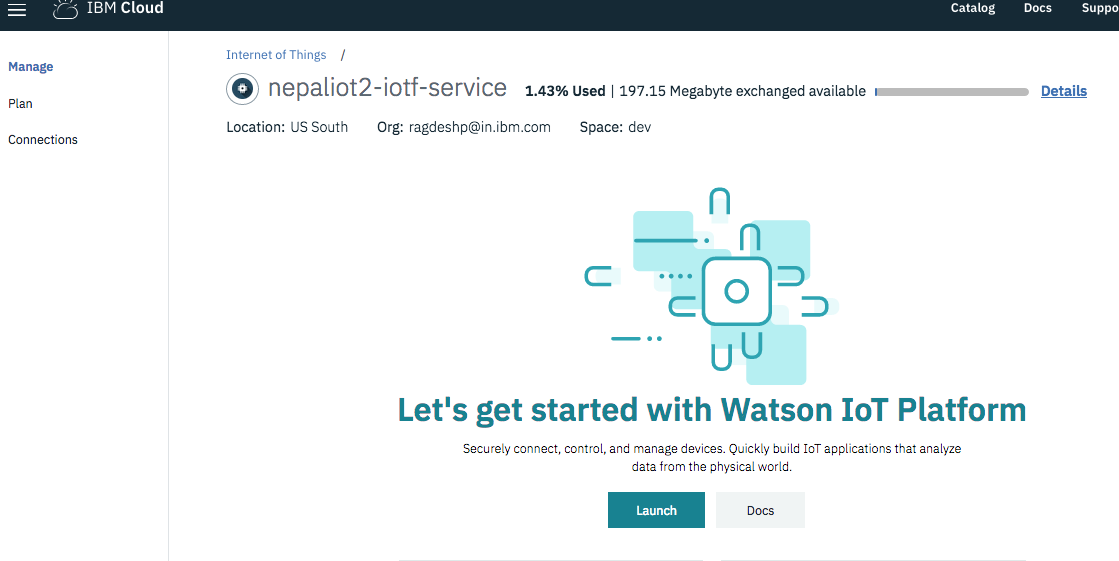
For example:

Organization is your email id.

Dev is your space.

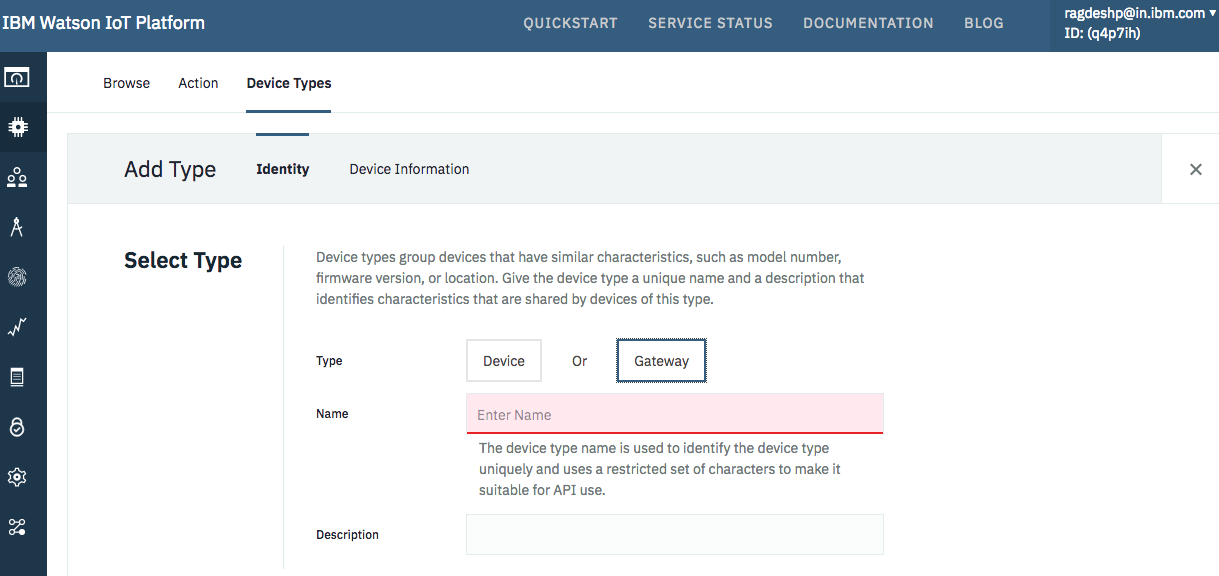


Step4: After creation of service, you should see something like below, click on Launch

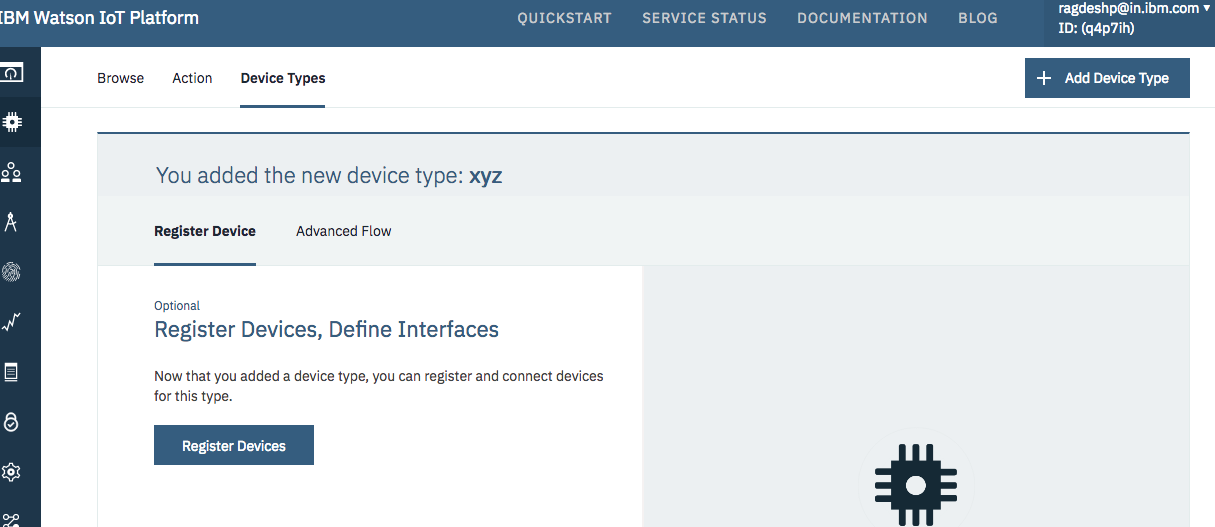


Step5: On Click on Launch, you would be taken to the IBM Watson IOT Platform page

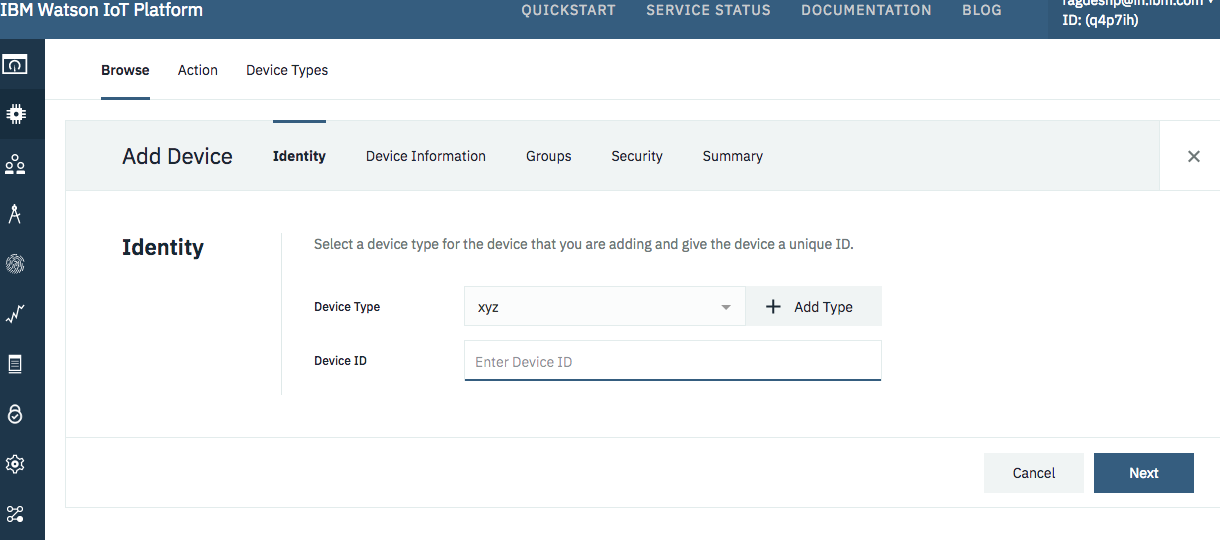
Step6: Click on ‘Device Types’ and create a Gateway by entering a name and click Next and then click Done.



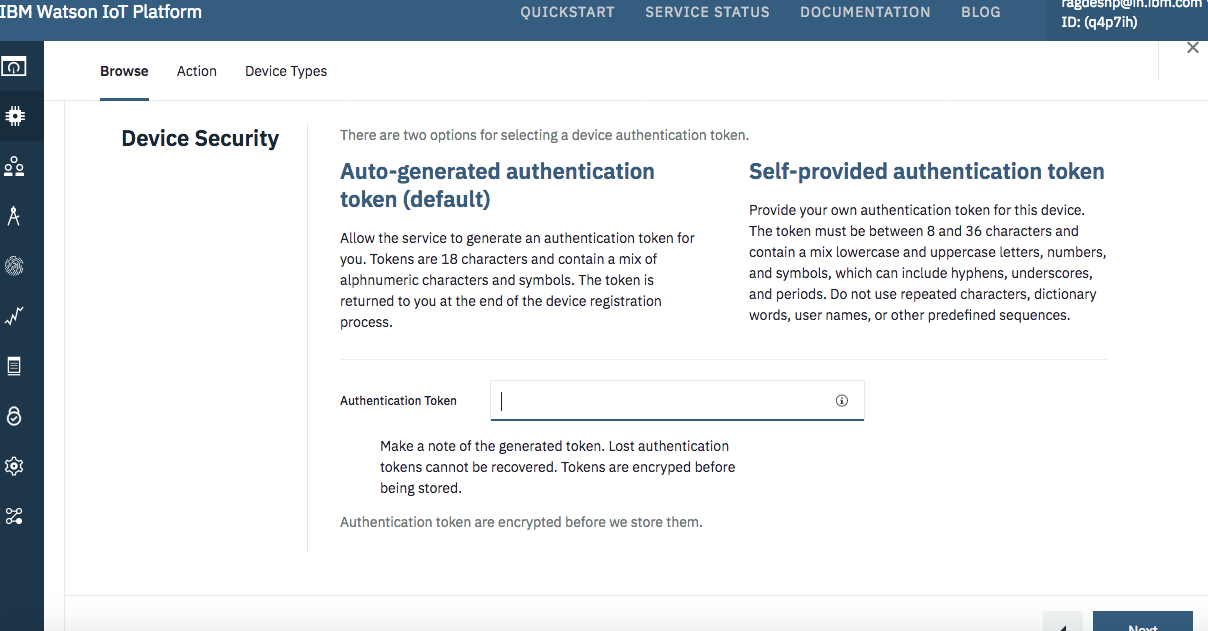
Step7: Add Device to the Gateway: click on “Register Devices”



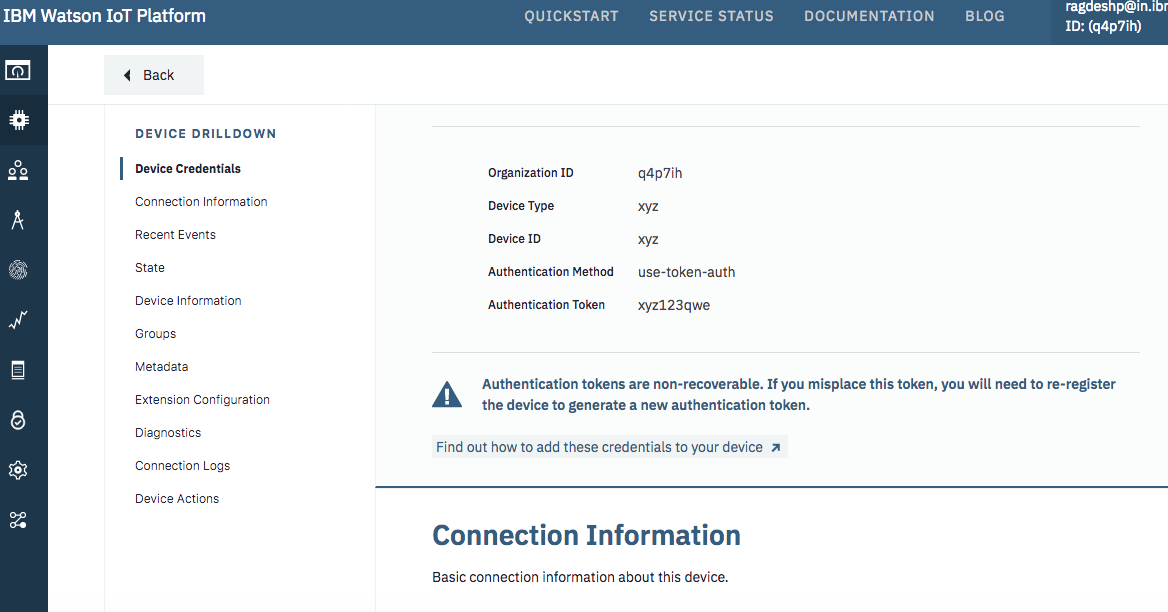
Step8: Add the DeviceID and click Next 🡪 Next 🡪Next.



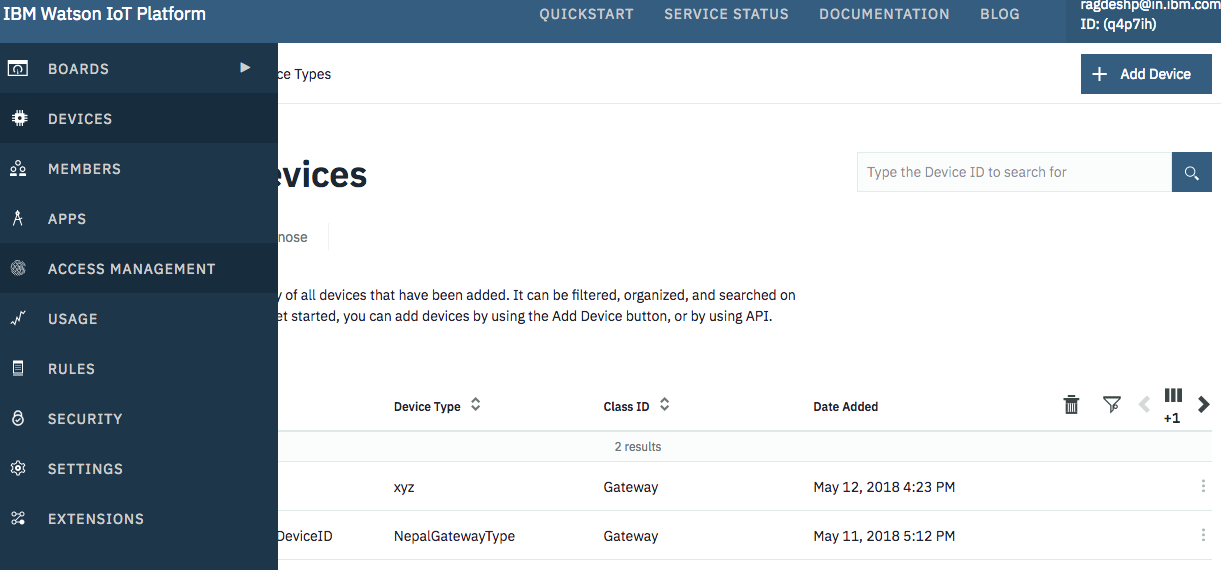
Step9: You should see the following page for Authentication Token. Enter the token which is mix of alphanumeric and special characters. Click Next 🡪 Done



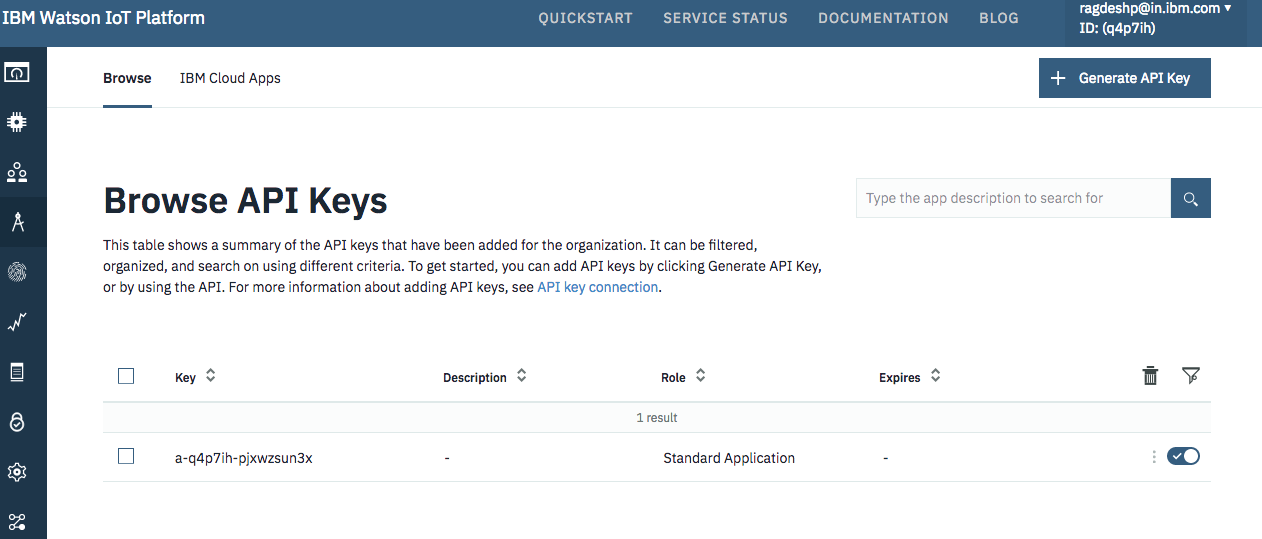
Step10: You should see the detailed information of the device type, deviceID and Authentication Details. Save all these details, this would be required.



Click on the Apps link to generate the API Key and API Token as follows:

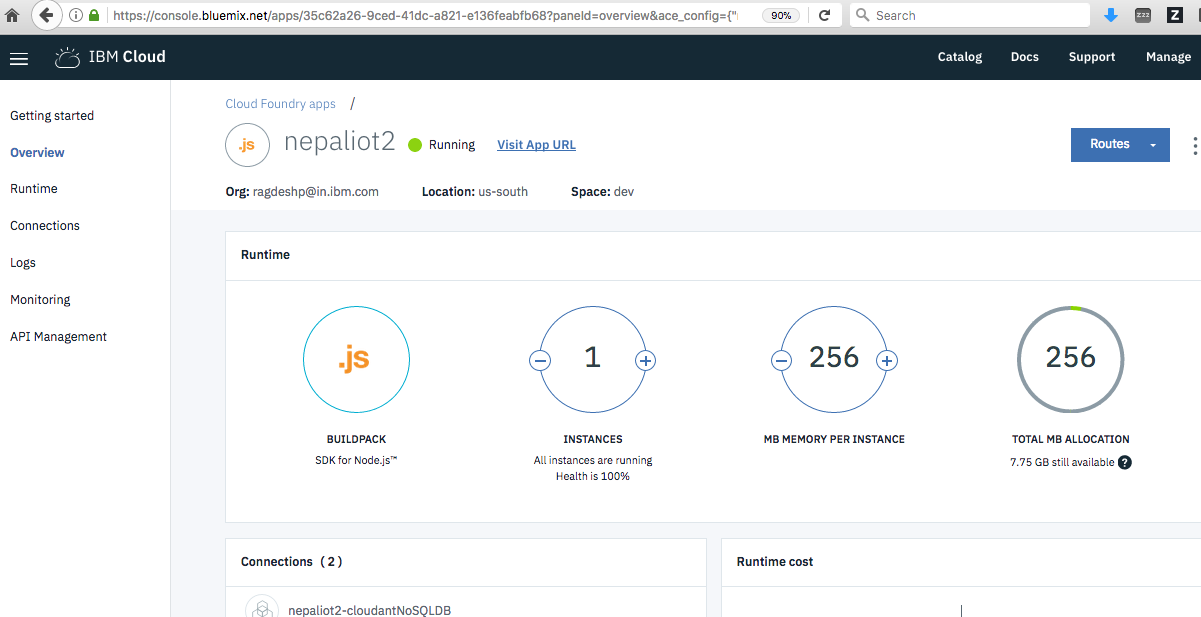


Click on ‘Generate API Key’ to generate the API Key:

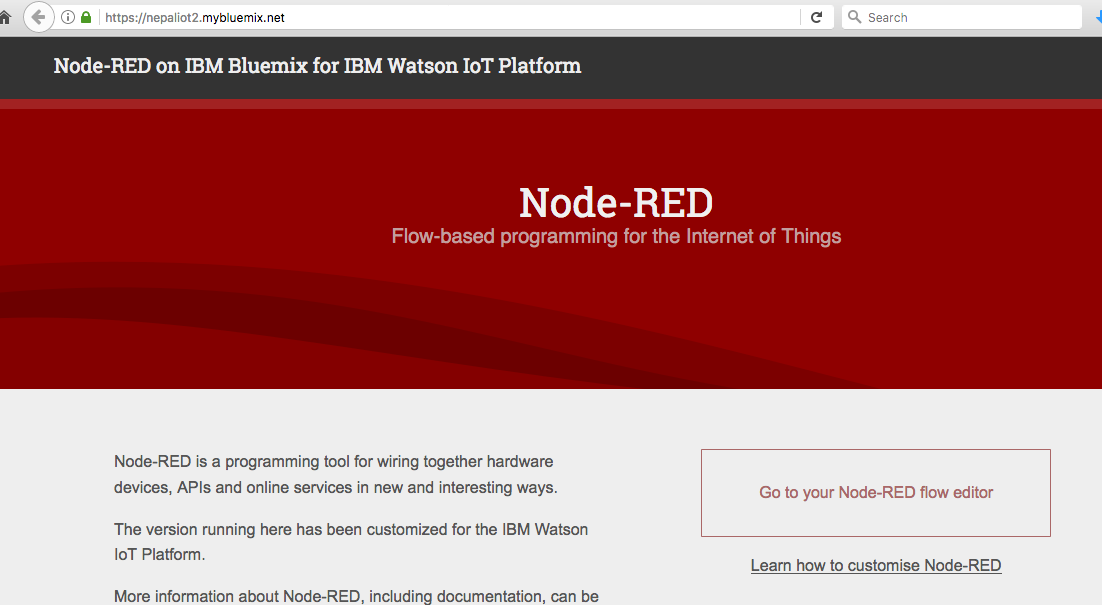


Click Next 🡪 Add Role as ‘Standard Application’ and then click ‘Generate Key’.

Step11: In Cloud Foundry Apps, you should see an App, click on it and it should take you to the following page:

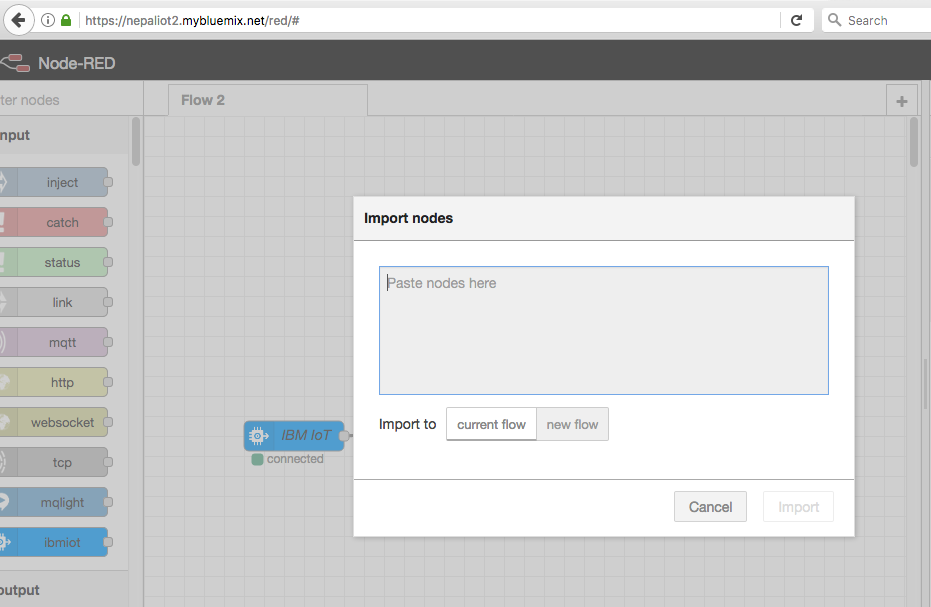


Click on “Visit App URL” link and it should take you to the following page which is NodeRED home page:



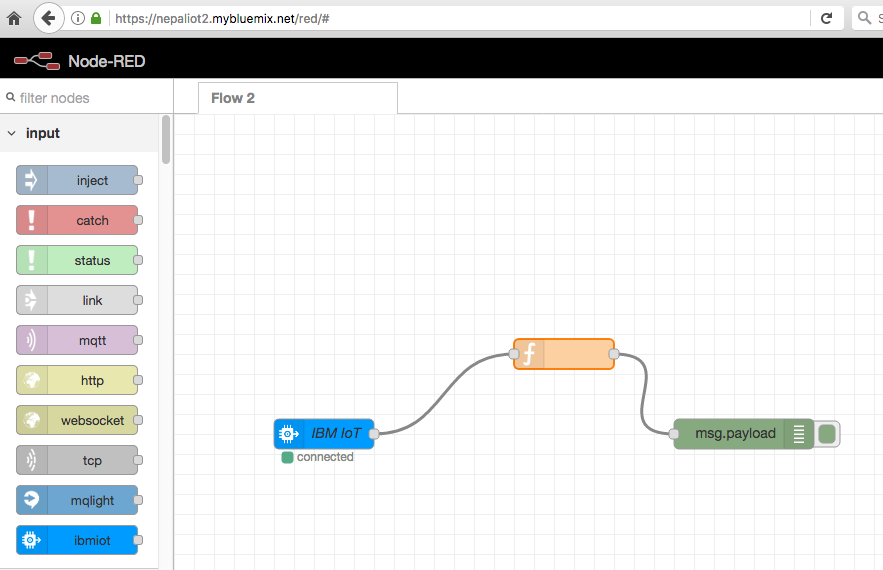
Click on the “Go to your Node-RED flow editor” link and it should open the Node-RED layout.

Step12: In Node-RED, click on the hamburger on the right and click on Import 🡪 Clipboard and you should see the following page:



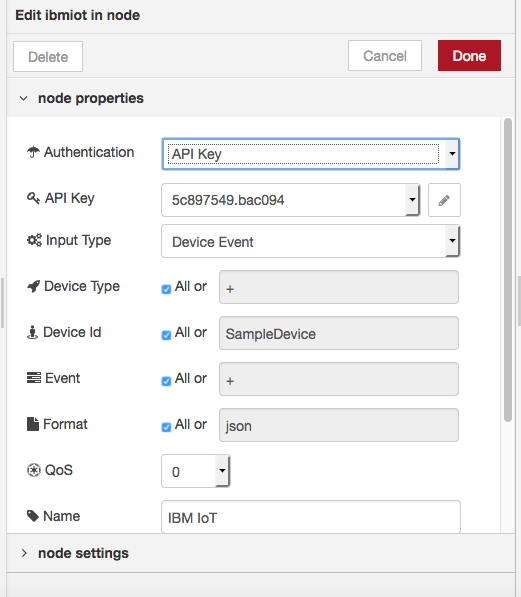
Paste the code in flow.json to this node and click Import.

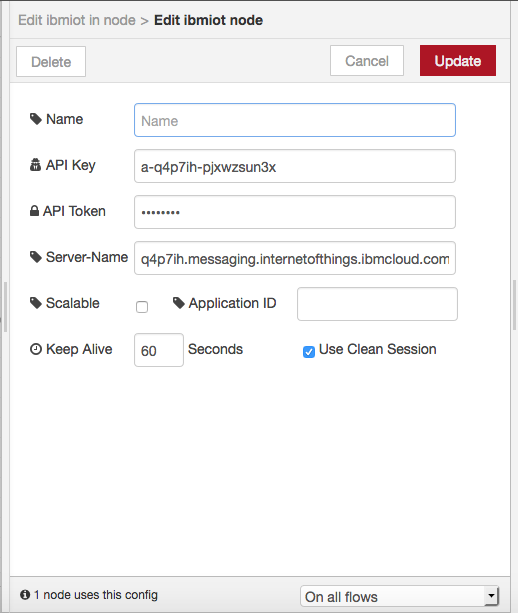
Step 13: You should see the Node-RED diagram as follows:



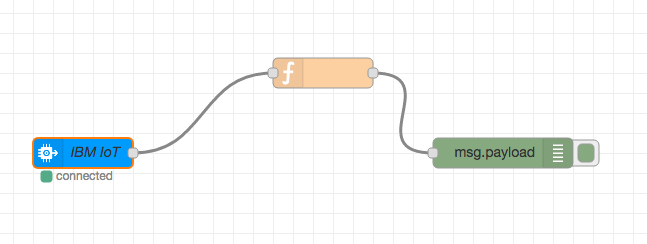
Step14: Click on the IBMIoT Node and should be able to see the properties:

Click on API Key Edit icon and then enter the APIKey and API Token. Select the Input Type as ‘Device Event’ and rest ‘All’ ticked.



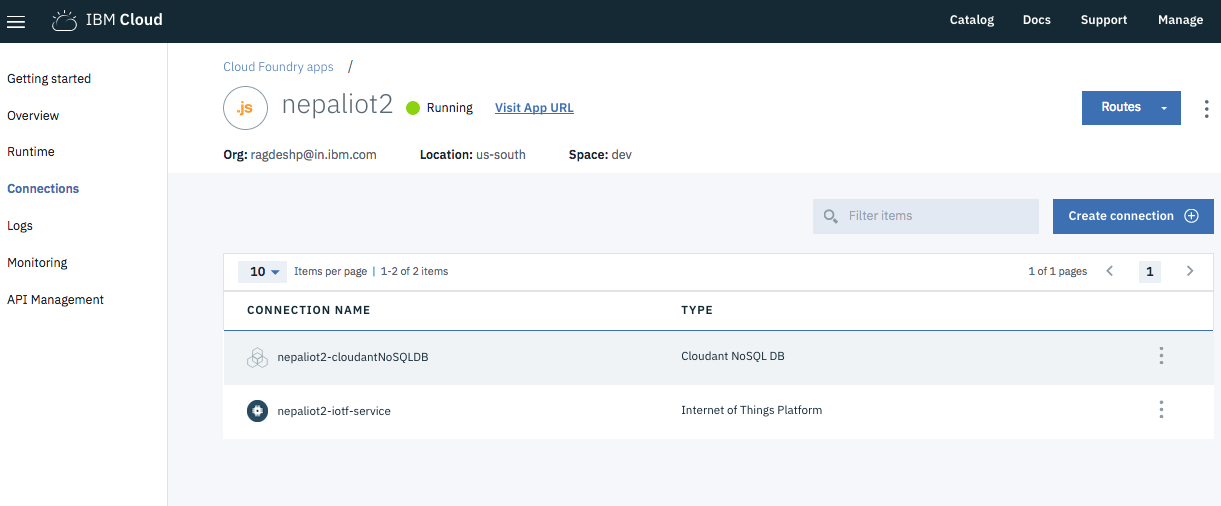


Step15: Click Deploy on the Right Hand side and you should see the connected sign on IBMIoT node as follows:



Step16: Let us get the sample App running which will allows us to upload the images to cloudant database. Before that let us create a database in cloudant.

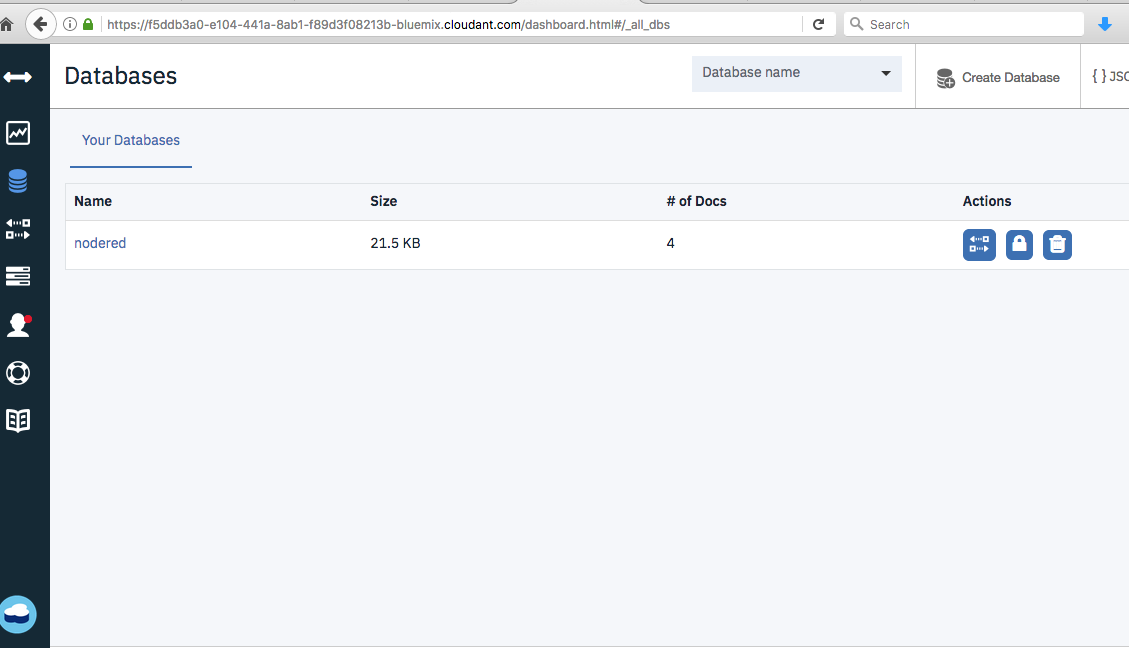
Click on the Connections Link in the Cloud Foundry App and you should see the connections as follows:



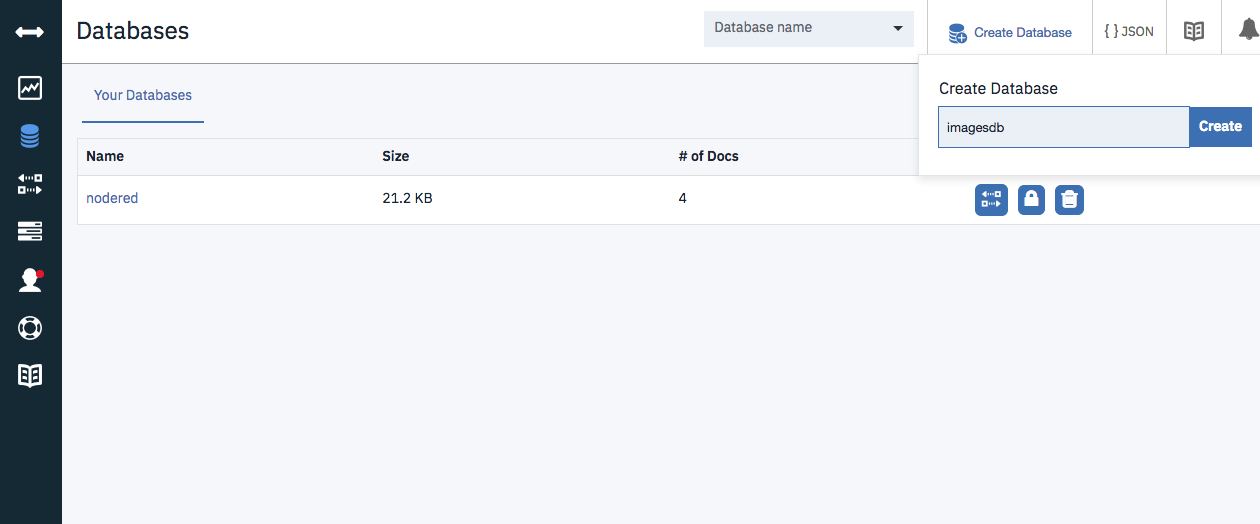
Click on the Cloudant NoSQL DB instance and it should take you to the cloudant dashboard as follows:



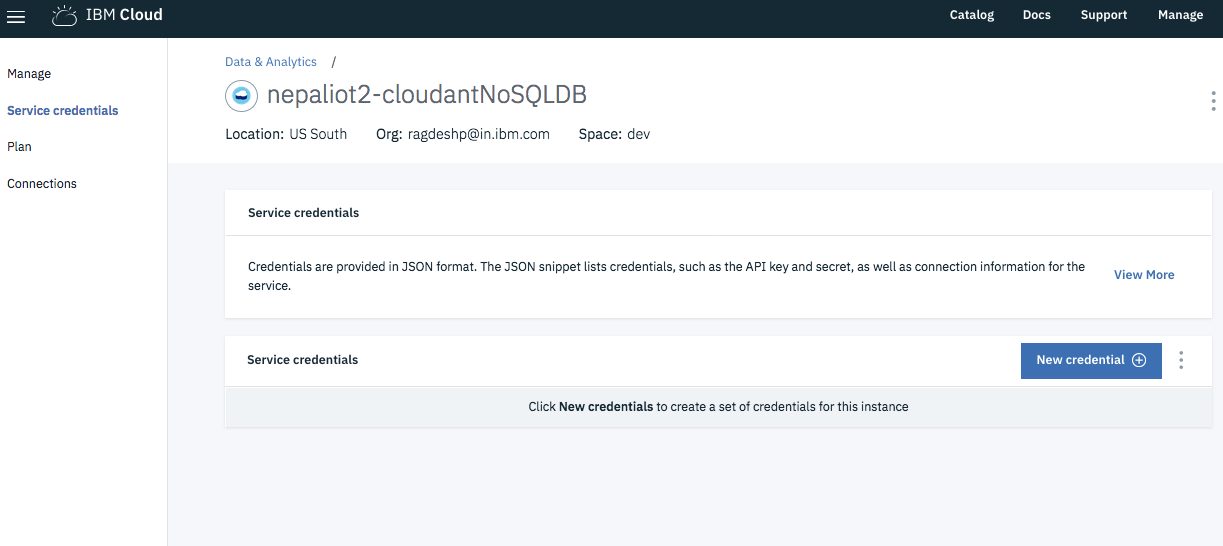
Click on Launch and you should see the following page:



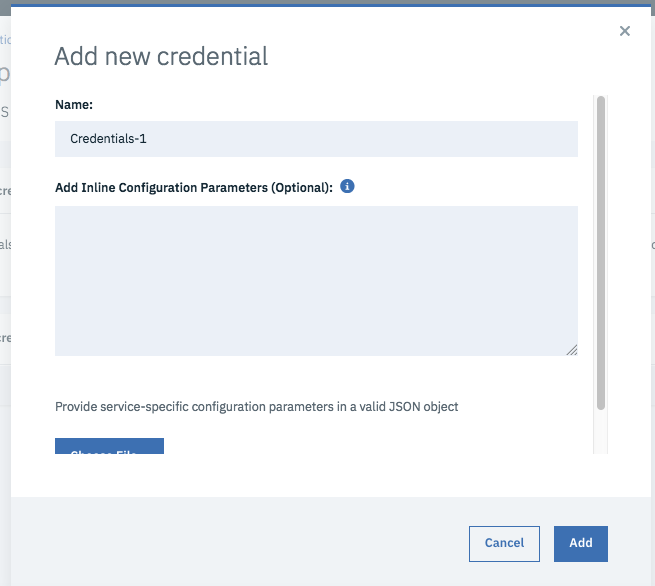
Create a database as ‘imagesdb’



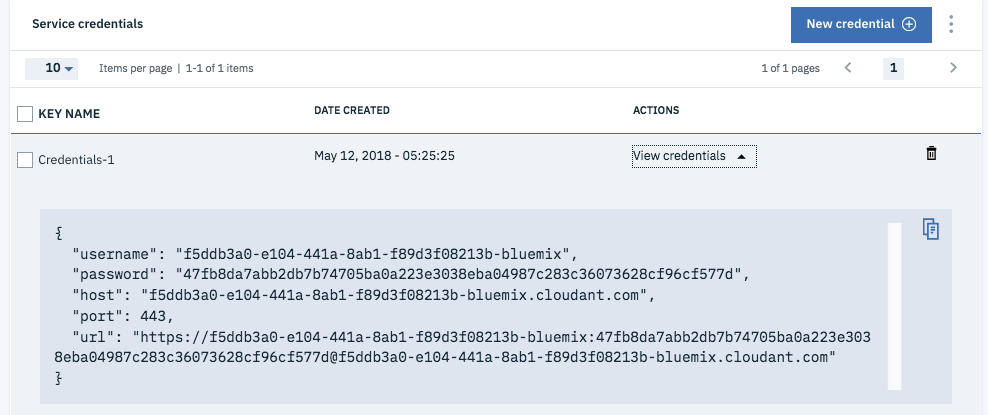
Next we need to get the service credentials for the cloudant instance:



Click on New credential link and Click Add and you should see the credentials.

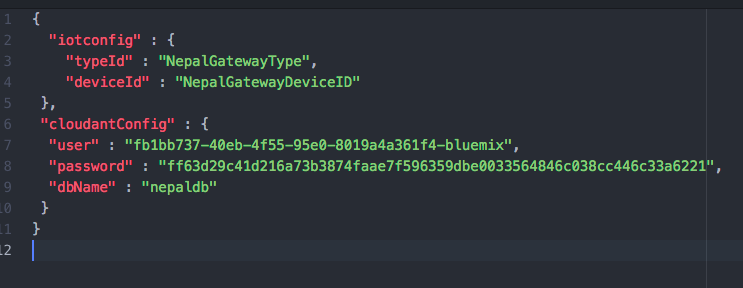


Note the credentials:



Update the cloudant credentials in credentials.json at location - image-analysis-iot-alert/viz-send-image-app/

Update the Gateway Device and cloudant details in credentials.json as follows:



Step17: Next go to the following location /image-analysis-iot-alert/viz-send-image-app

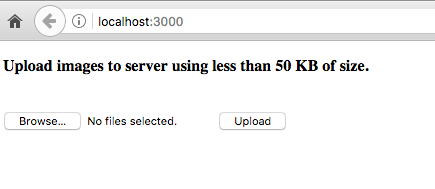
Run the following commands:

cd viz-send-image-app

npm install

npm start

Access the url <http://localhost:3000/> and you should see the following page:



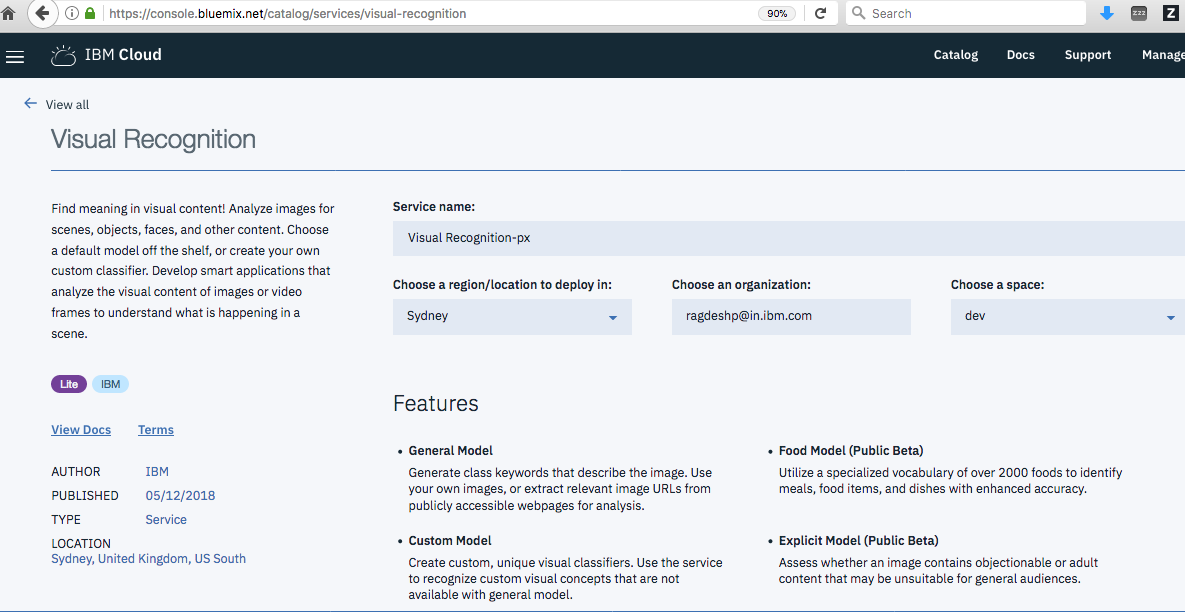
Upload an image and you can check whether there is a record in the cloudant database.

Step18: Add your credentials to credentials.env.example and rename it to credentials.env

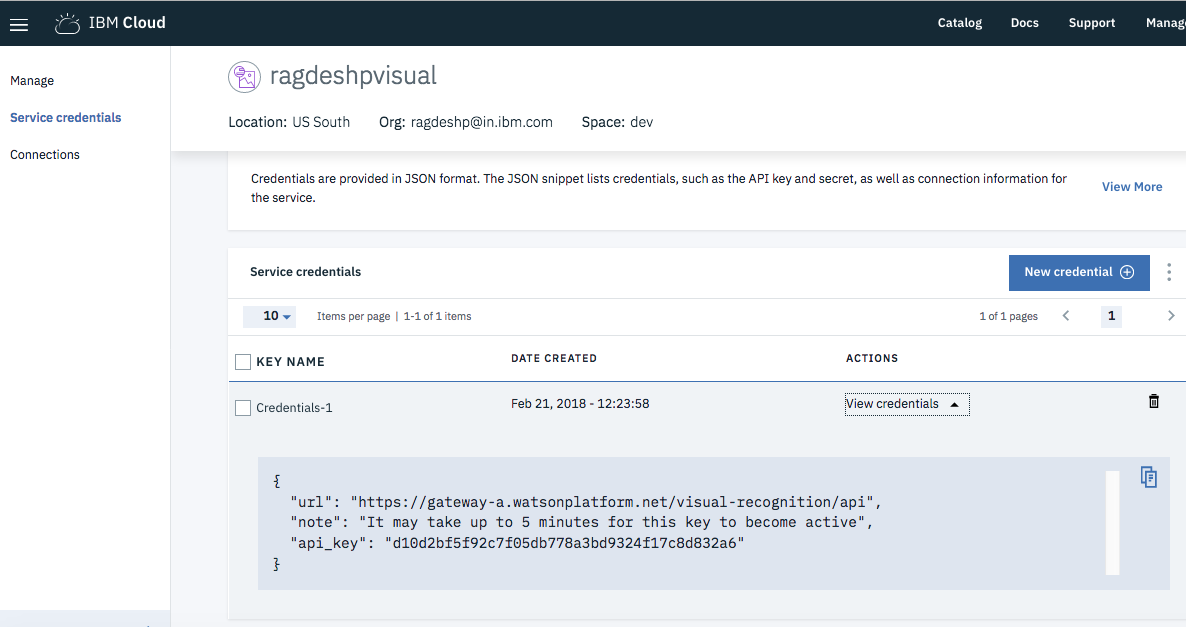
Visual Recognition and IBM Cloud Functions: Enter the following configuration properties in …/image-analysis-iot-alert/viz-openwhisk-functions/credentials.env

18a: Create an instance of Visual Recognition:

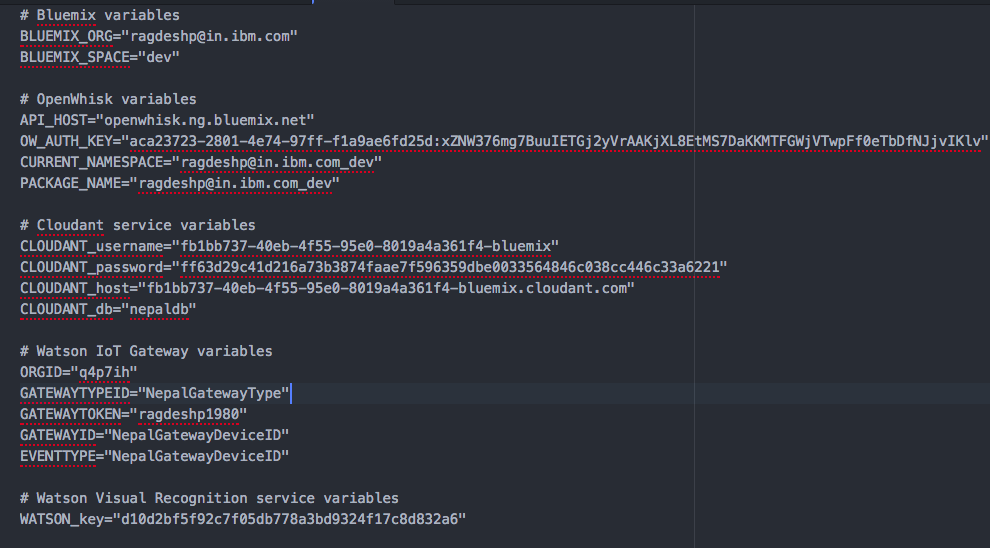
Go to the catalog, search for Visual Recognition and create an instance:



Note the service credentials of the visual recognition – api\_key and enter it as WATSON\_key in …/image-analysis-iot-alert/viz-openwhisk-functions/credentials.env



18b: To get the OW\_AUTH\_KEY, run the following command in your terminal: bx wsk property get --auth



Next Run the following commands depending upon the environment:

For Mac users (mac-ubuntu-linux.sh is for Linux based OS):

$ cd viz-openwhisk-functions

$ ./mac-ubuntu-linux.sh --install

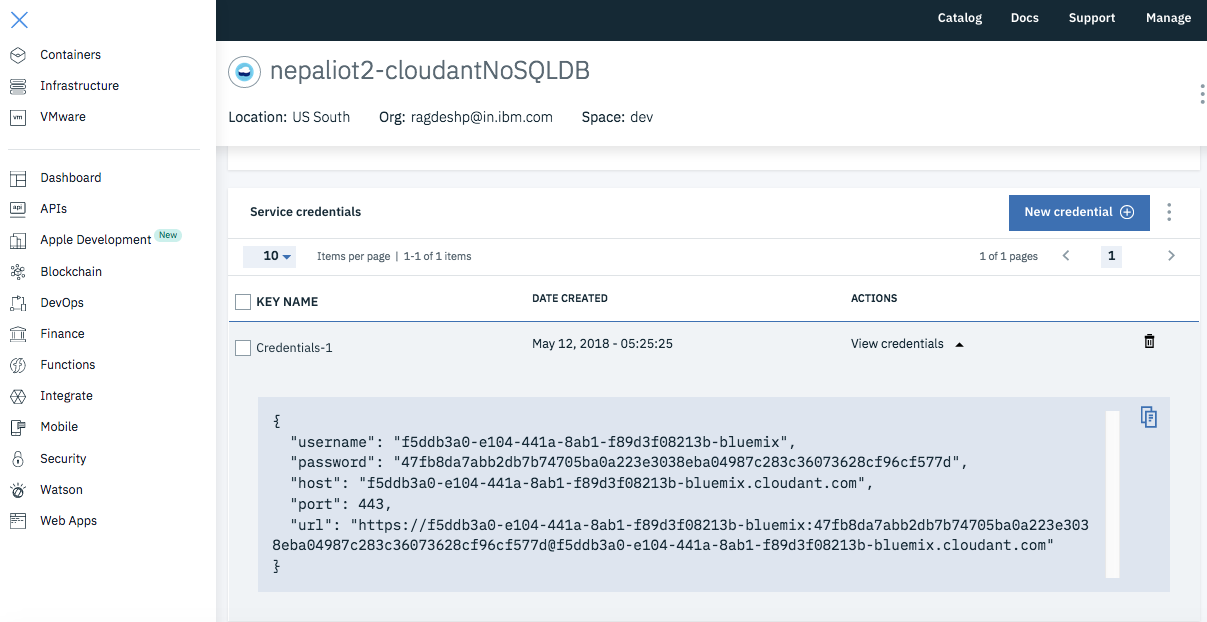
For windows users:

$ cd viz-openwhisk-functions

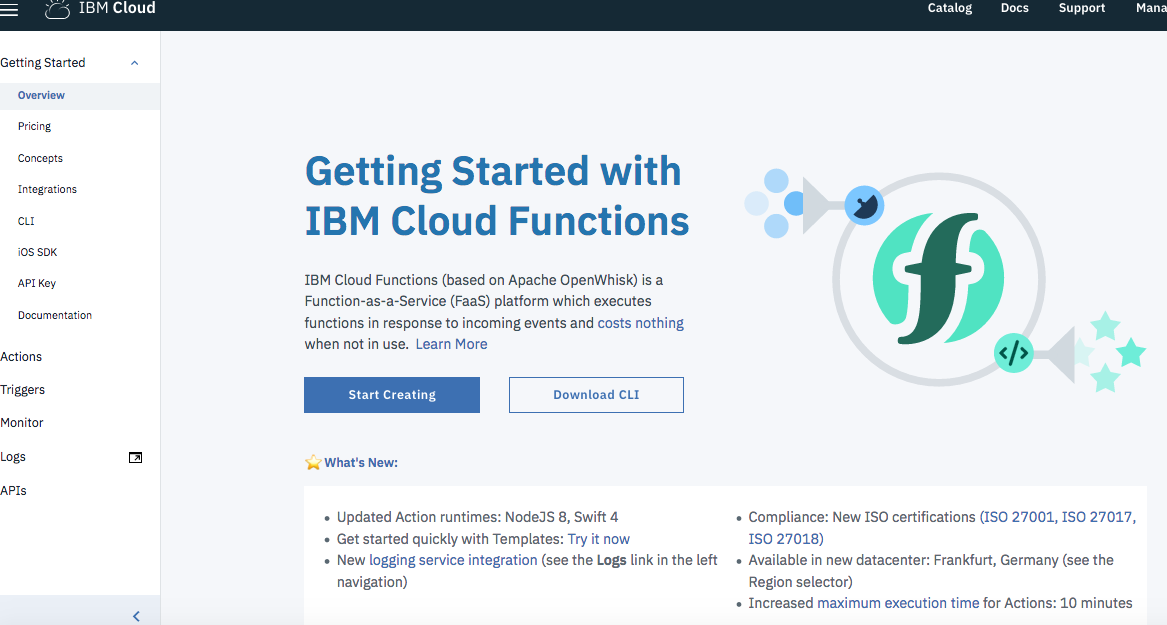
$ windows.bat --install

This script creates actions, triggers, sequence, bindings etc.

To view these click on the hamburger on the left top most of IBM Cloud dashboard and click on Functions



Next, you should see the IBM Cloud Functions dashboard:

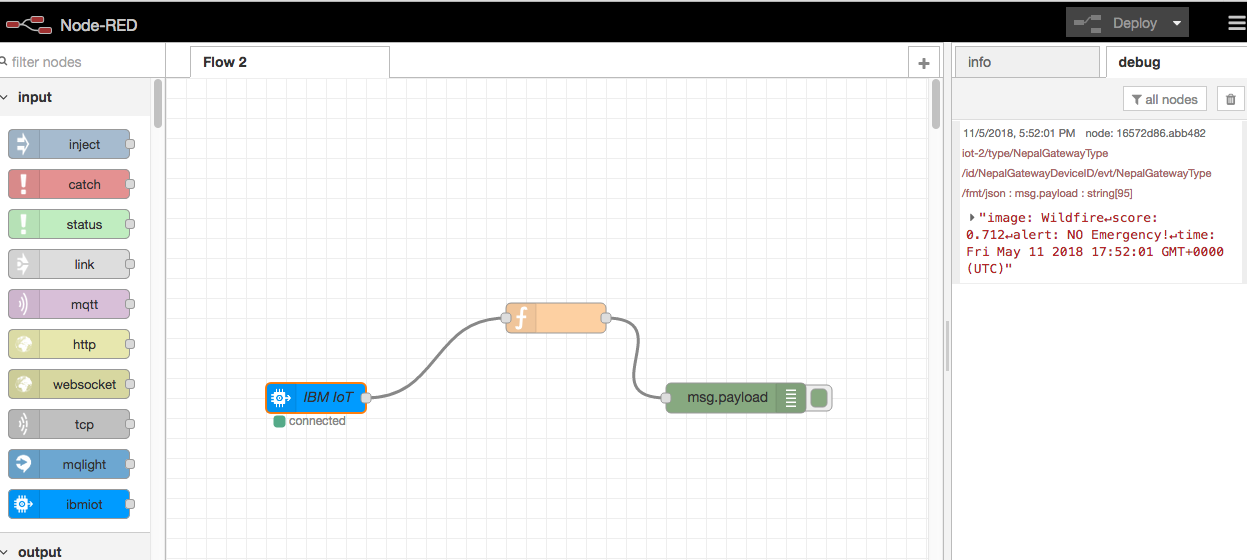


Step18: Running the Application:

Upload an image of fire.jpg through <http://localhost:3000/> link

You would get a successful image upload message.

Then go to the Node-RED dashboard and observe the results as follows:



You see that the image has been recognized by the Visual Recognition API which is called from the IBM Cloud Functions and the analysed results are passed to the IOT platform and from IOT platform it is sent to the Node-RED end user application. Node-RED can be interfaced with other user interfaces to observe the results.