MIRACLE
SOFTWARE SYSTEMS

Dev Workshop | IoT APIs 101 with StrongLoop and IBM Bluemix

IOT Developer Lab | Miracle Innovation Labs

Chanakya Lokam
Director – Marketing and Innovation
Miracle Software Systems, Inc.

February 14, 2016

Dev Workshop | IoT APIs 101 with StrongLoop and IBM Bluemix

Goal

In this lab we will be using a OBD II Simulator to get real-time vehicle data and then connect it to the IBM Watson Internet of Things Foundation. We will then use NodeRed to orchestrate this data into a data source such as Cloudant.

Pre-Requisites

- Active Account for IBM Bluemix
- Browser to access Cloudant and Bluemix
- Internet Access

Technology Involved

- NodeRed
- Node.js
- IBM Bluemix
- Cloudant and NoSQL

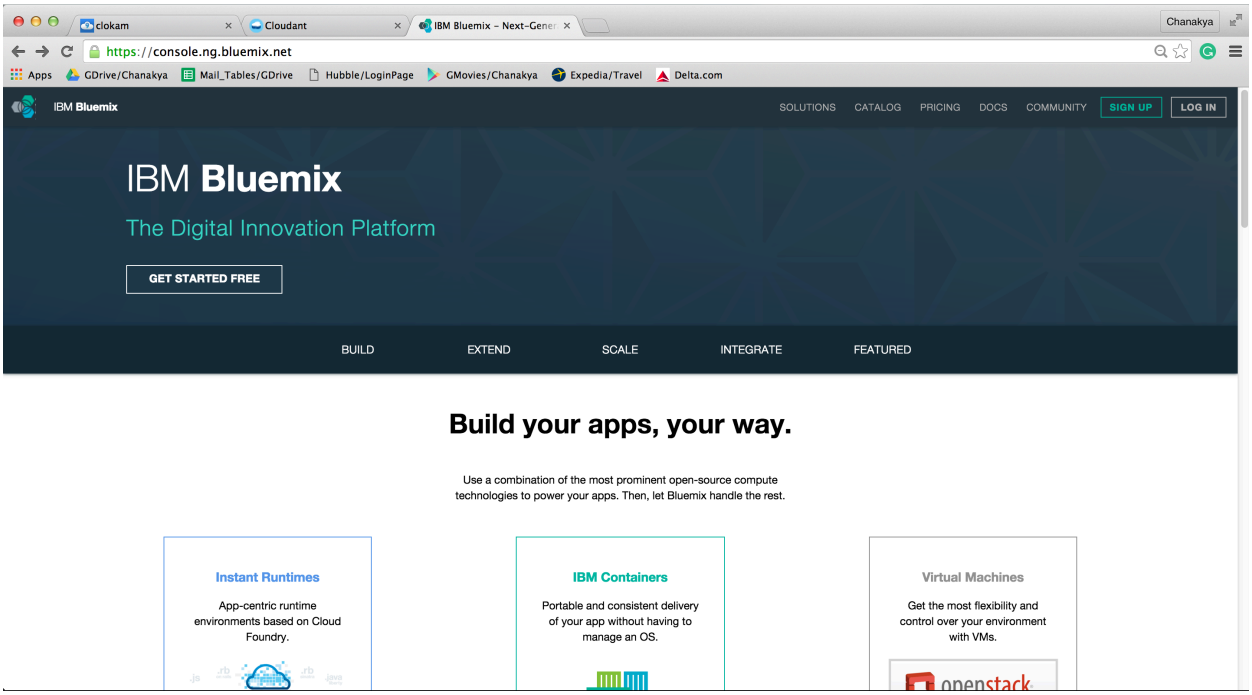
Download the tutorial at https://github.com/MiracleLabs/IotApis101Lab

Lab Steps

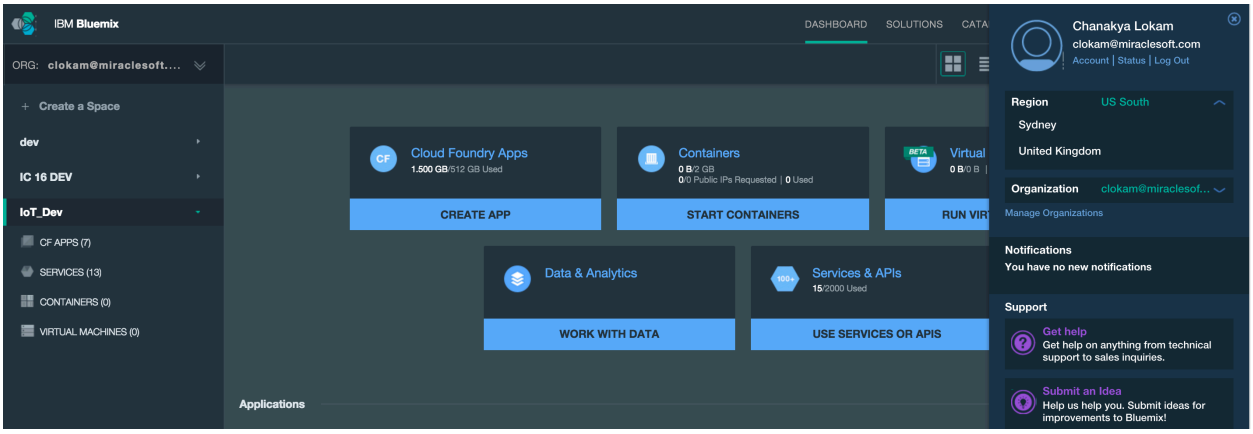
So, let us get started with the lab!

#1 | Get Access to IBM Bluemix

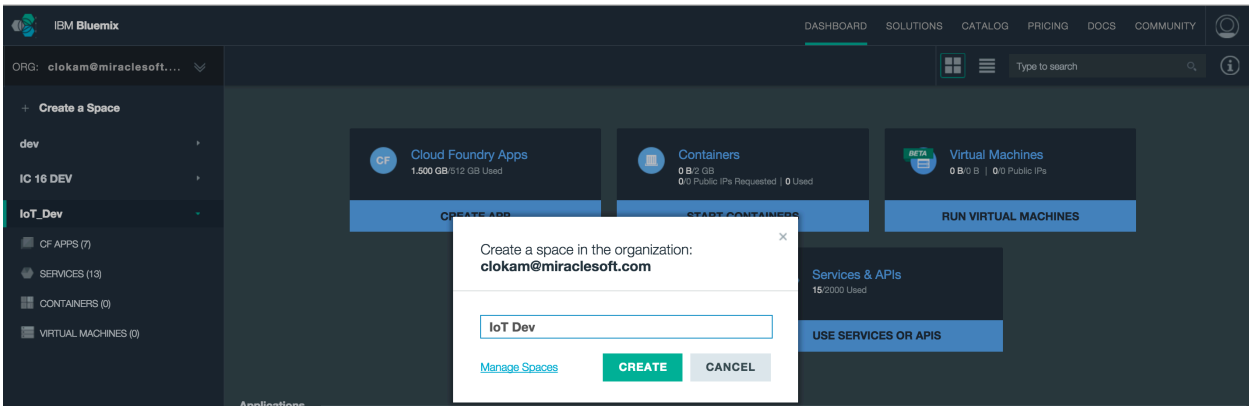
The first step of the lab will be to gain access to IBM Bluemix. If you already have an account then proceed by logging in from http://bluemix.net (or) sign up today.



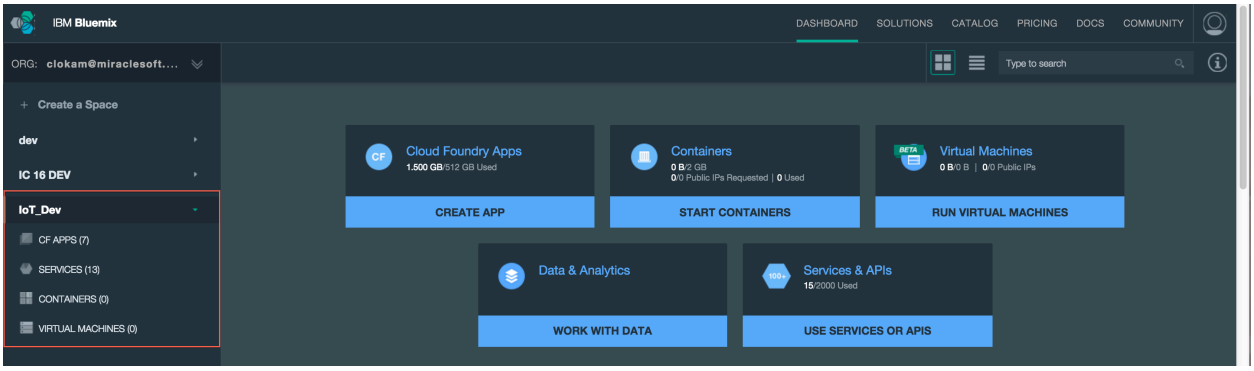
Once you have logged in you can see your dashboard and catalog screens. Go to your profile screen at the top and change the **Cloud Availability Region** to **"US South"**. Note that the IoT Foundation Service is only available in that region.



The next step will be to create your space within Bluemix to create your applications and services. Click on the **"+Create a Space"** on the left side corner of your dashboard and create a new space, for example **"IoT_Dev"**.

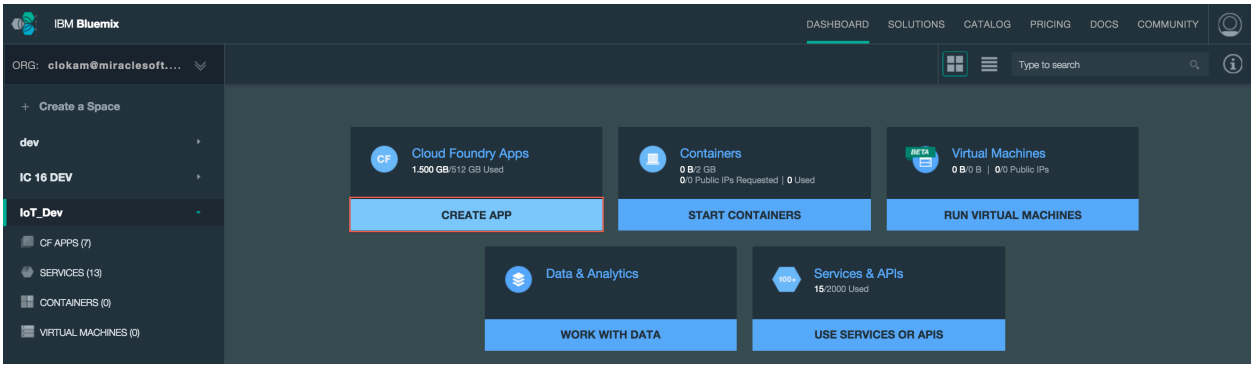


Note that the space creation will take a few minutes. Once the space has been created you will be able to see the new space on the left side panel.

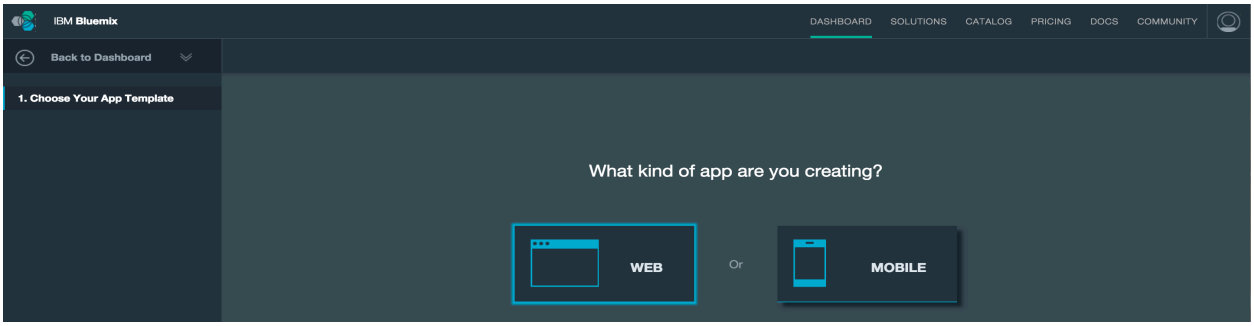


#2 | Create the IoT Starter Application and Service

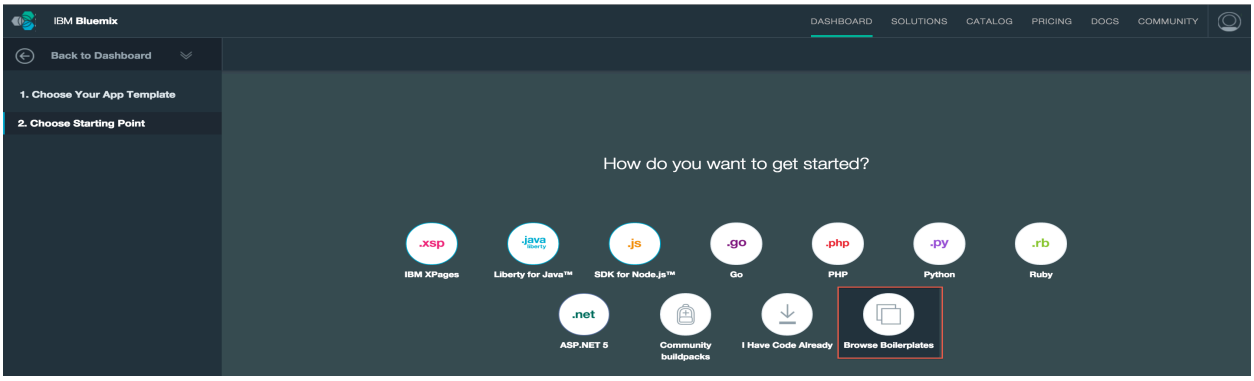
The next step will be to create the IoT Starter Application. Select the **Cloud Foundry Apps** option and click on **"Create App"**.



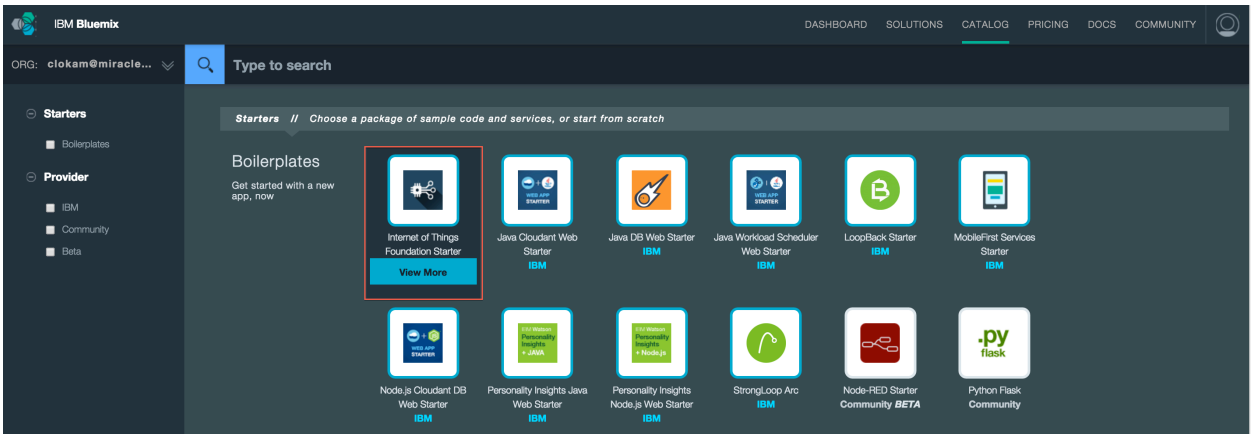
When prompted select **"Web Application"**.



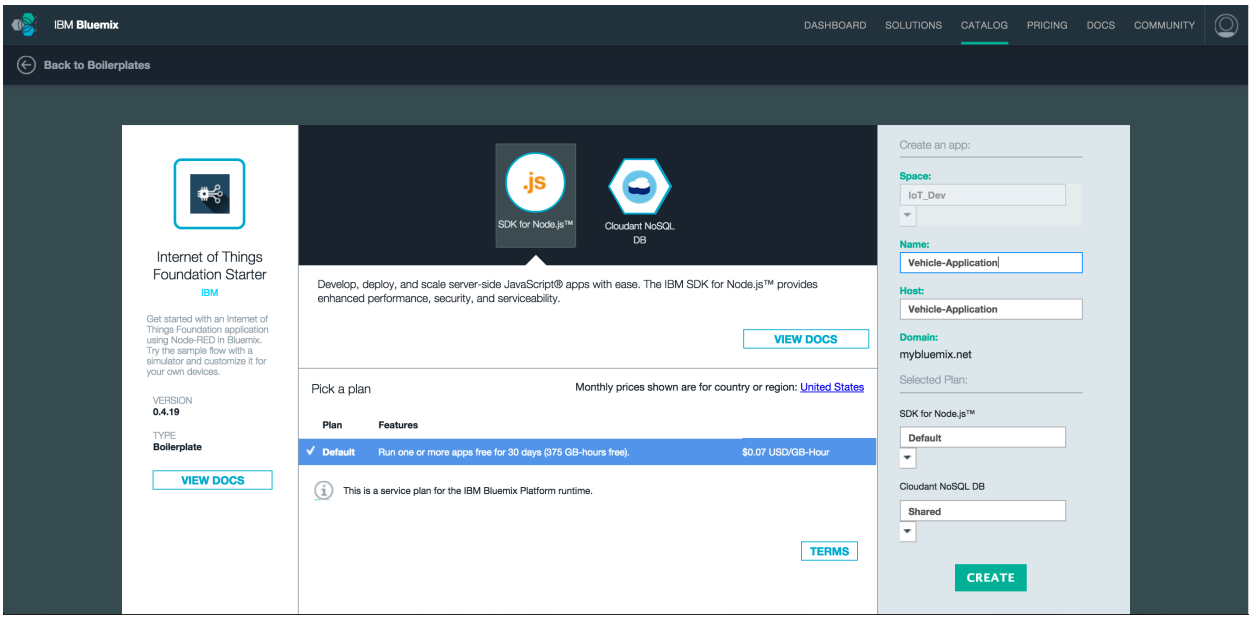
In the **"How do you want to get started?"** page, go to the **"Browse Boilerplates"** option.



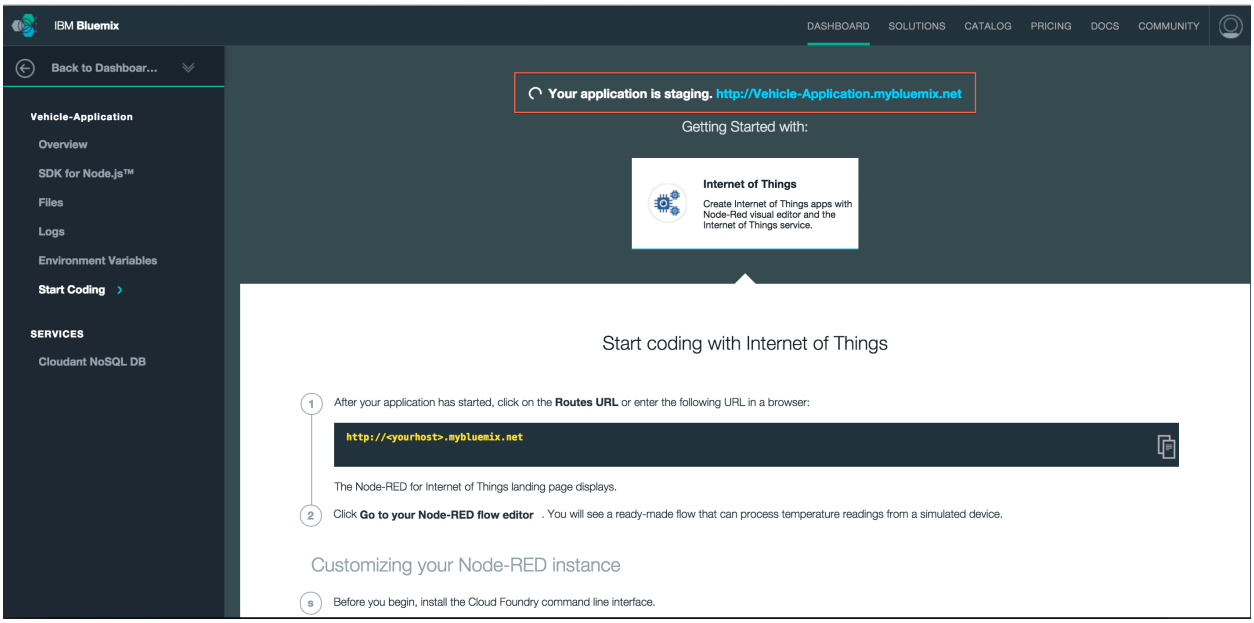
Out of the available boilerplates, select the **"Internet of Things Foundation Starter"** option.



In the **“Create an App”** page give your application name and create the application with the default options within the free tier. The application will have the Node.js SDK and Cloudant DB Services included within it by default.

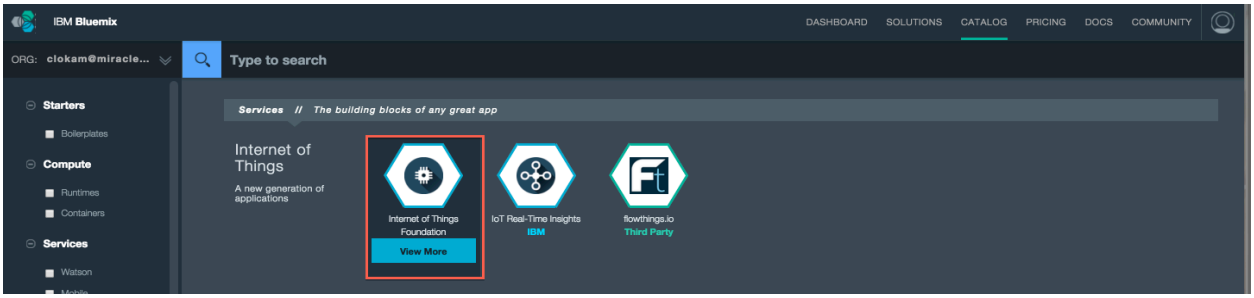


Note that the application will take a few minutes to stage and then start running.

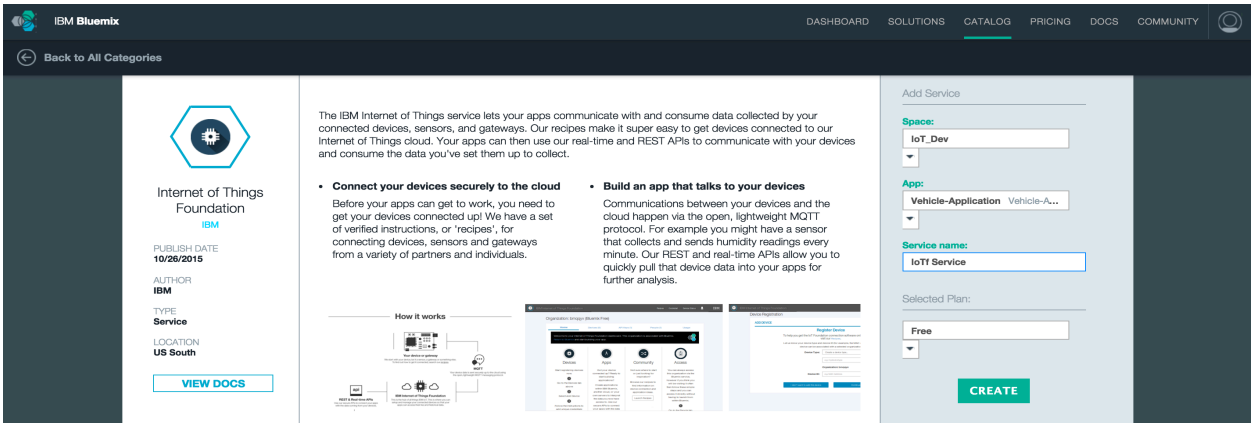


#3 | Adding IoTF Service to your Application

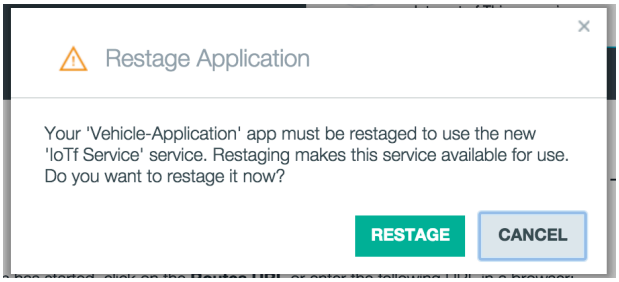
Go to the **"Catalog"** on the top right menu and select **"Internet of Things"** in the left side menu. Within the given options select the **"Internet of Things Foundation Service"**.



Keep the space as **"<Your Space Name>"** and in the **"App"** field bind your service to the IoT Starter Application that you created in the previous step.



At this point of time you will be asked to restage your application.

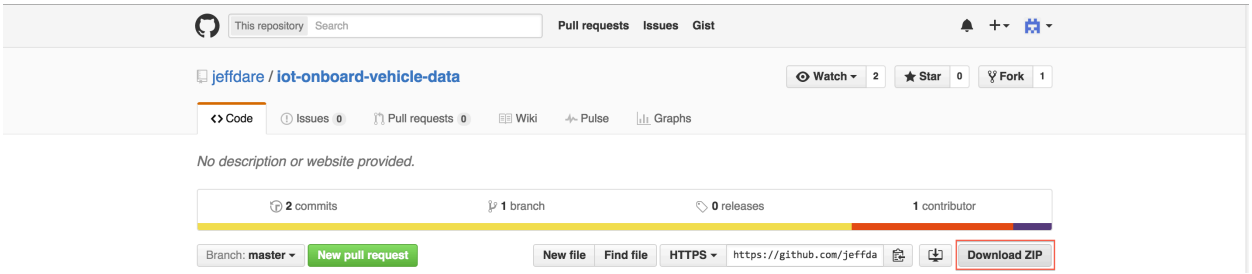


Note : This process will once again take a few minutes

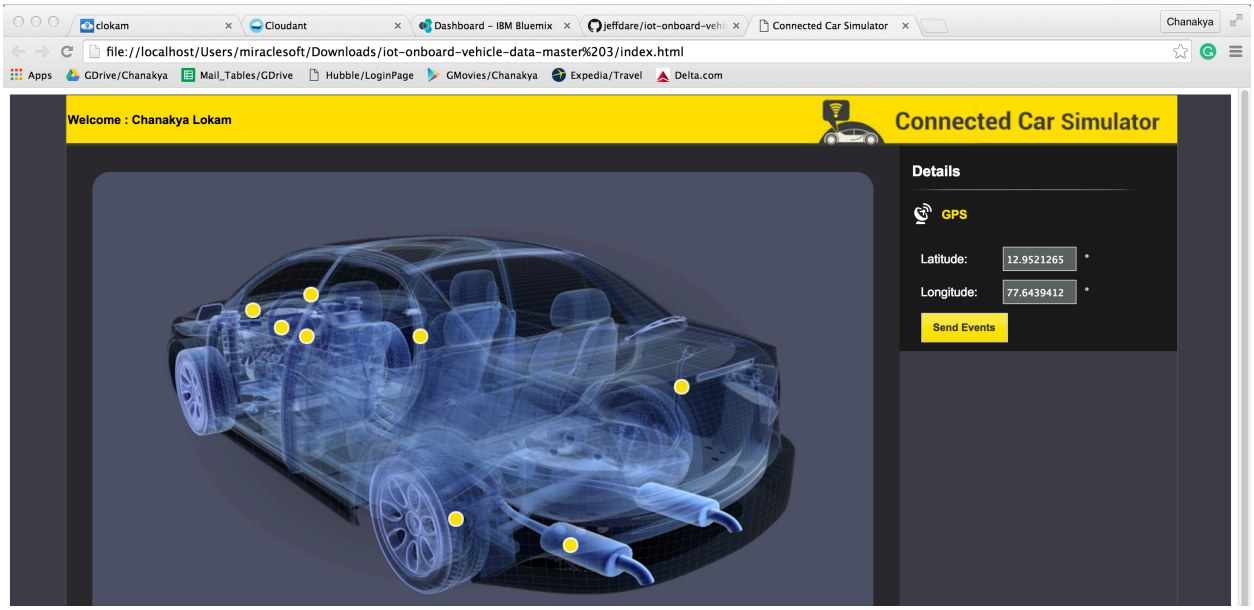
#4 | Get the OBD II Vehicle Simulator

You can find the Vehicle Simulator within the Git Repository that you downloaded at the start of this tutorial at,

https://github.com/MiracleLabs/IotApis101Lab

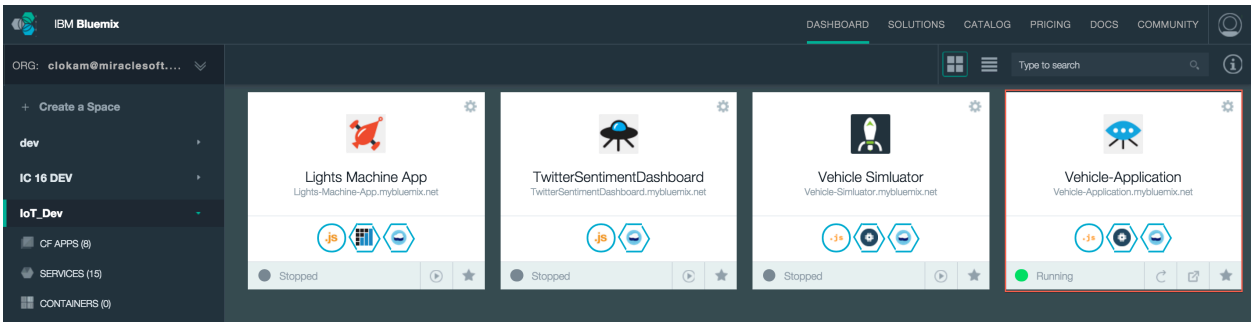


Once you have downloaded the files navigate to the **iot-onboard-vehicle-data-master** folder and open the **index.html** file with a browser. When prompted give your name, and then you can see the car simulators with data such as Fuel, GPS and more. (See Below)

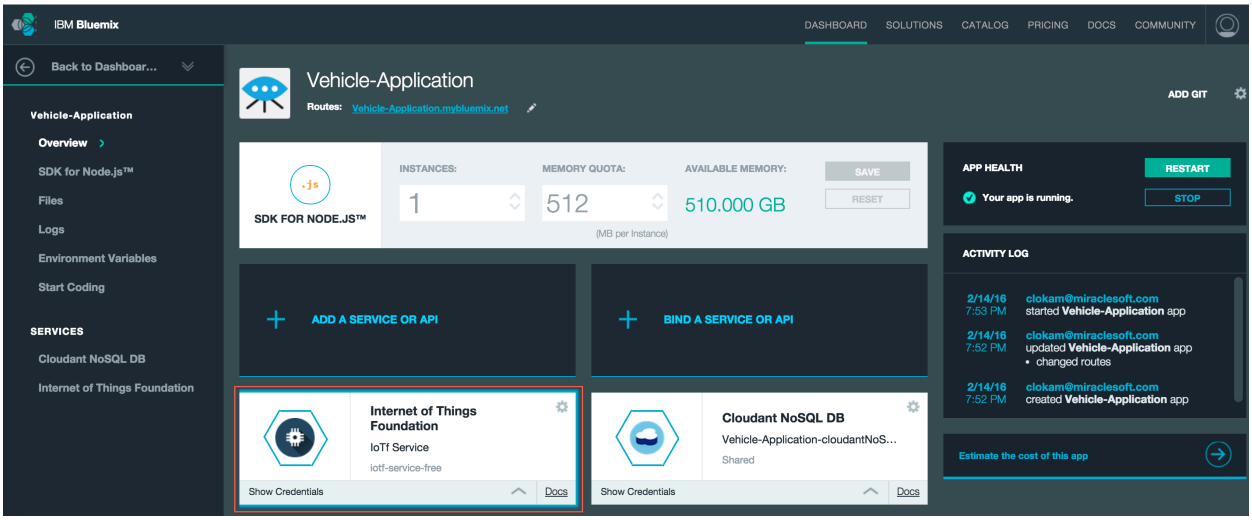


#5 | Register your Vehicle with the IoTF Service

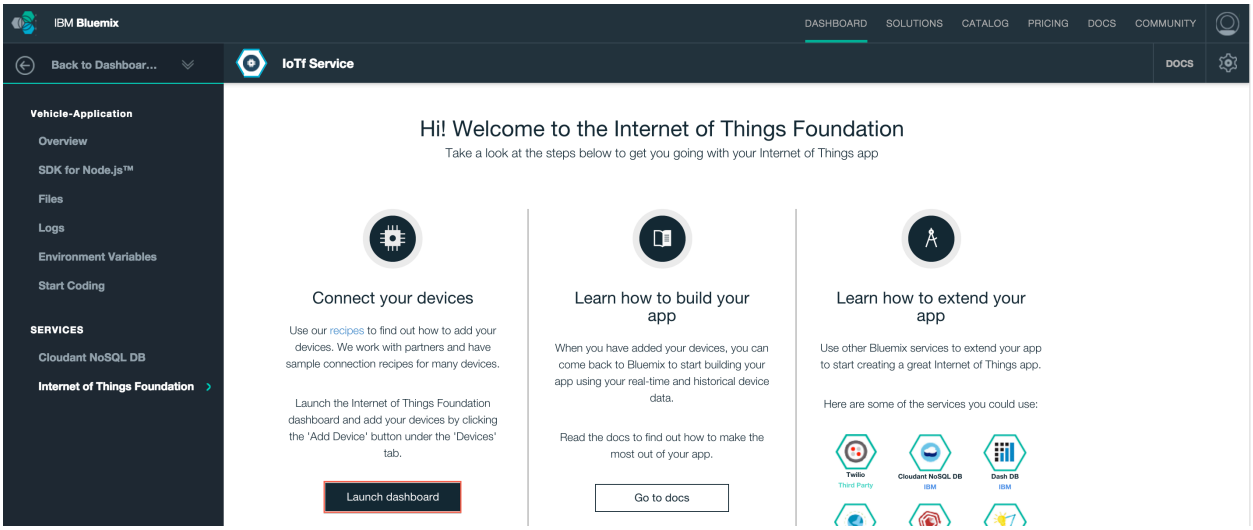
Now go back to your Bluemix Account and open your application through your dashboard. Now that we have our basic setup completed you will have to configure, register and connect your Vehicle Simulator to the IBM Bluemix Platform so that we can send data events from the vehicle simulator to the Cloud.



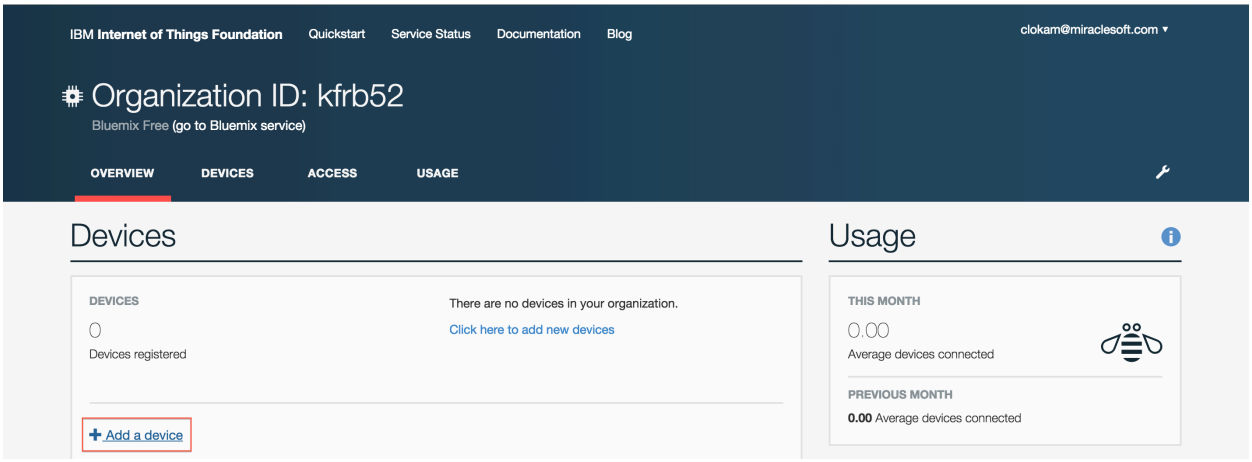
Select the IoTf Service that you created previously and bound to your application. Click on it to take you to the IoT Foundation Service Dashboard.



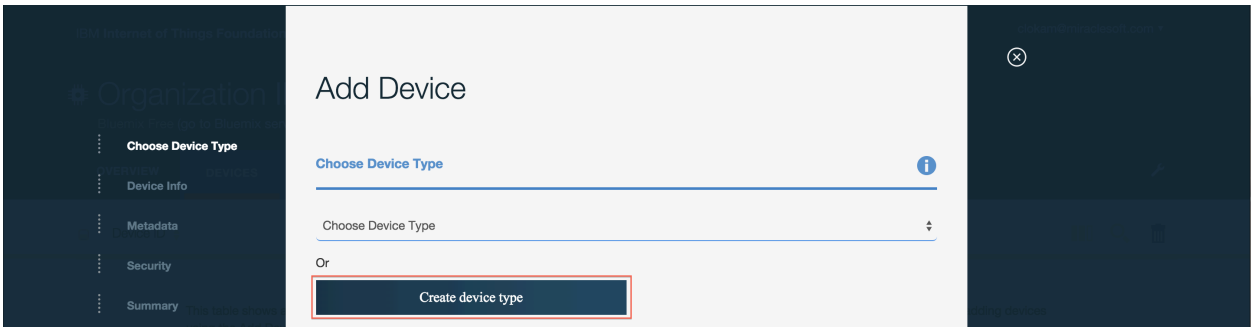
Once the Service Dashboard is opened up, select the **"Launch Dashboard"** option under the **"Connect your devices"** piece.



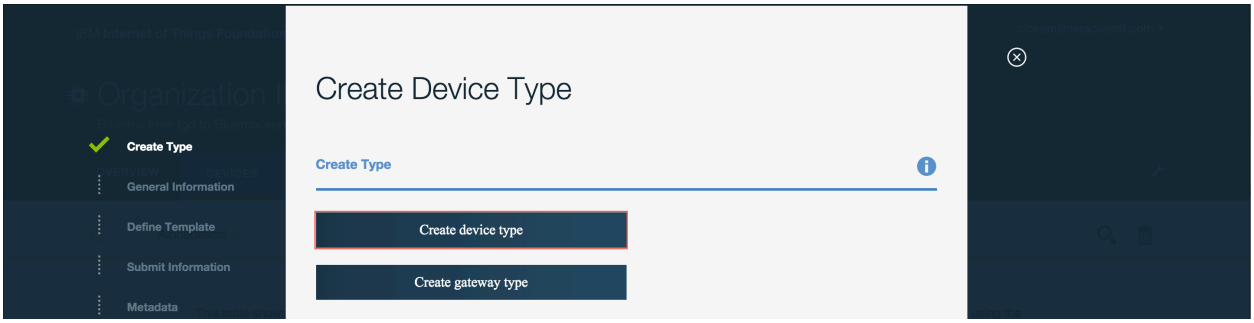
Click on **“+Add a device”**.



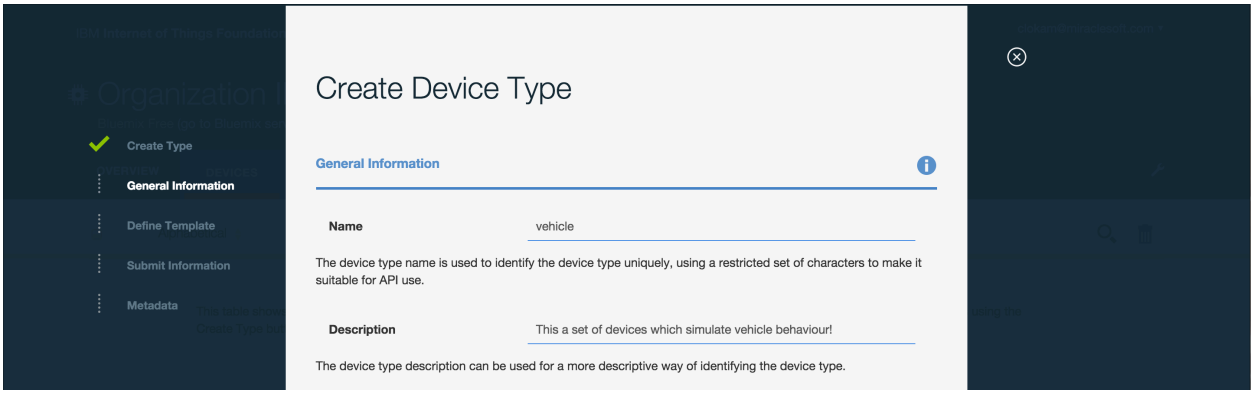
You will be prompted to create a new device type.



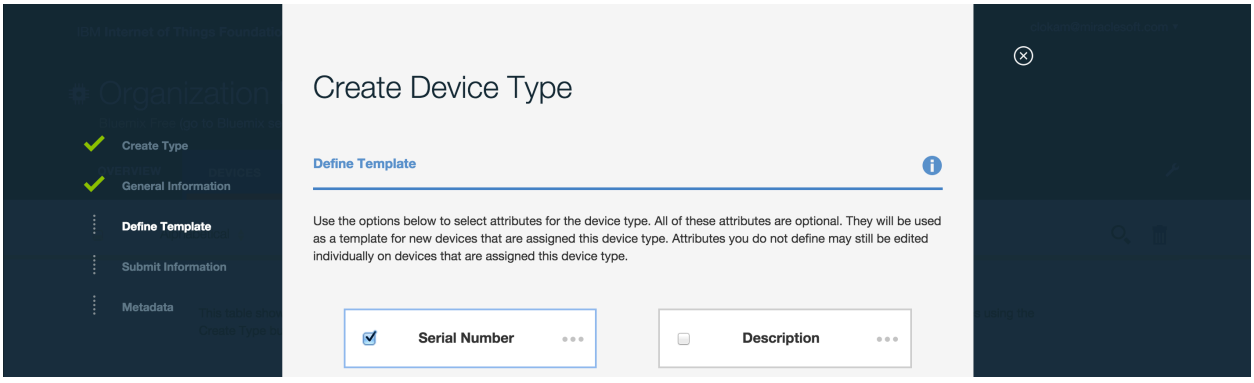
In the next dialog select **“Create Device Type”** as we want to add a device.



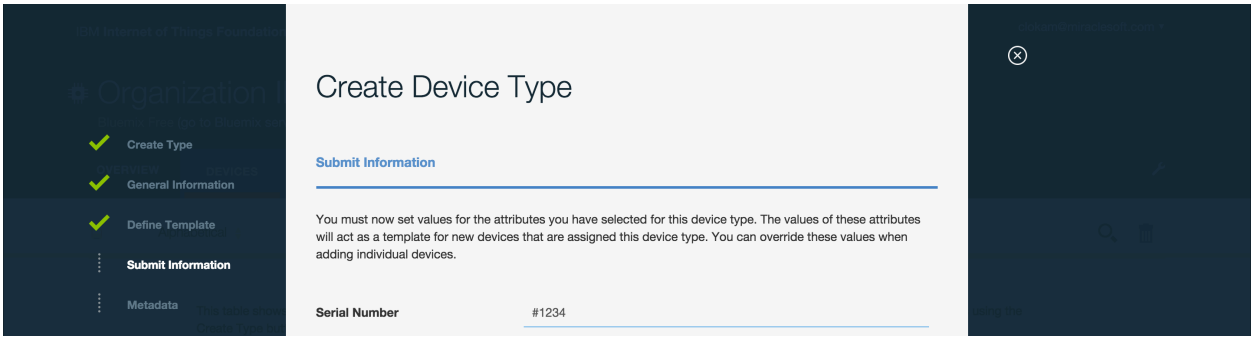
Enter the **name**(For Example “vehicle”) and **description** and click on next.



In the next page you can select which attributes you want to add for this device type. Let us select **“Serial Number”** and leave the rest as optional.

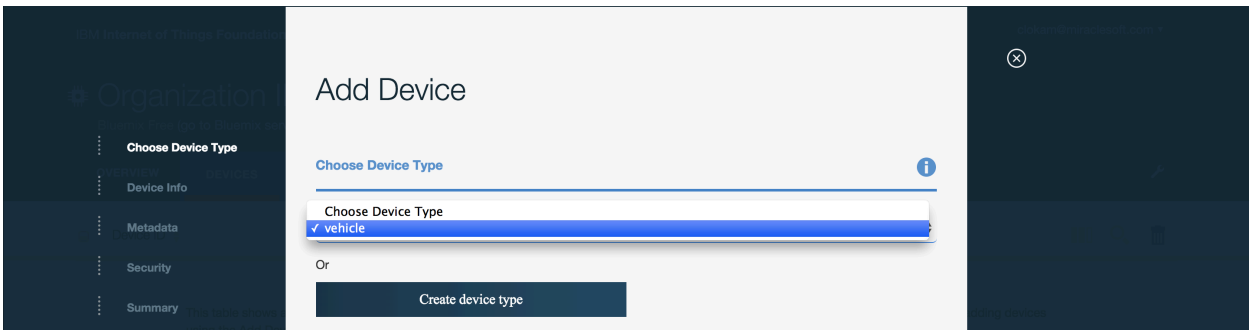


In the next dialog give a sample **serial number** to define its formats.

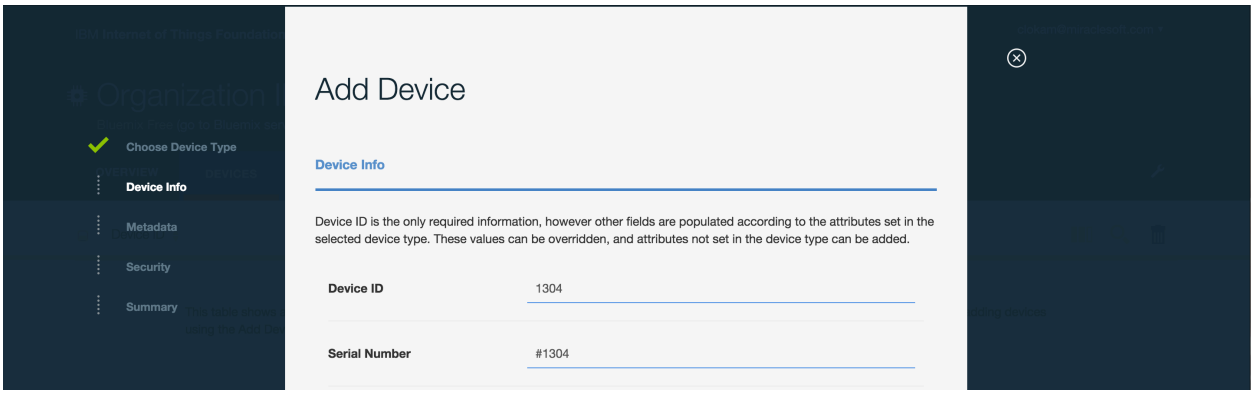


Note : You can skip the optional step to add Meta Data which comes next

Now you can add your device by selecting the “vehicle” as the device type.

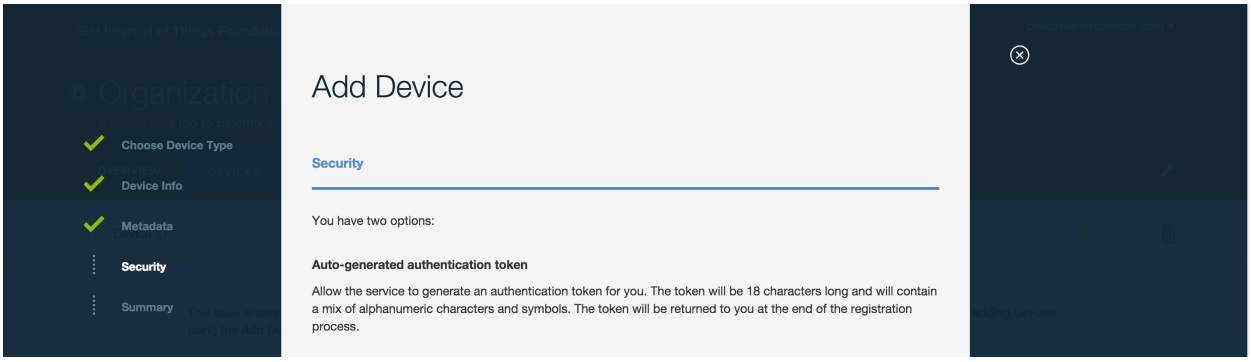


Give the Device ID and the Serial Number. Note that these values can be anything you want them to be.

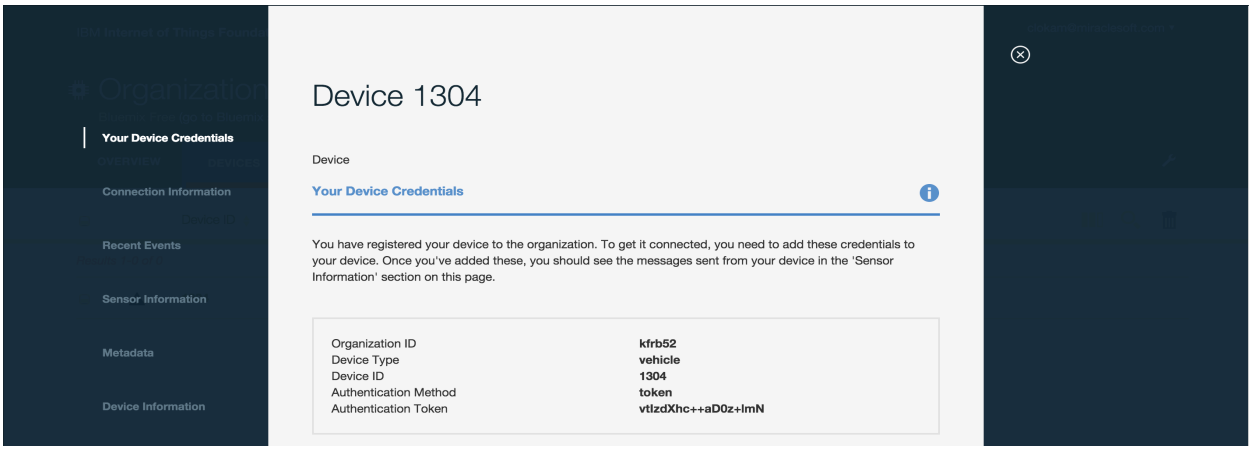


Note : You can skip the optional step to add Meta Data to the Device

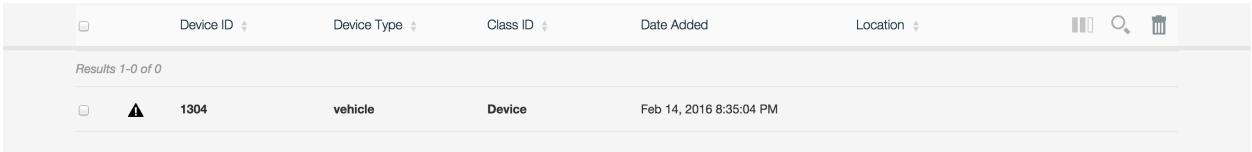
In the **"Security"** step select the Auto Generated Token option(default) and click next.



In the final step you can verify the data that has been given and finish the creation process of the device. Now that you have added the device you will be given a summary of your **Device Credentials**. **Make sure that you make a note of these**.



Now you can see that your device has been added to your organization but is not yet connected.

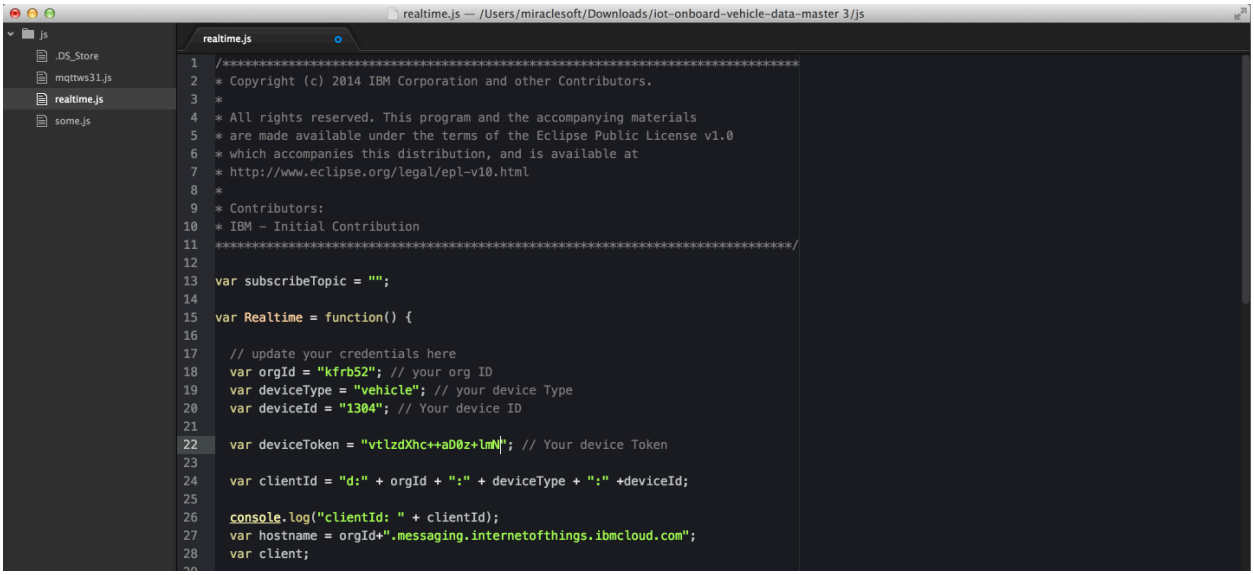


#6 | Configuring your Vehicle Simulator

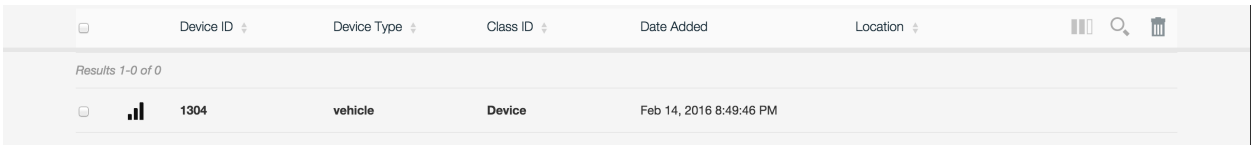
We have to now configure our Vehicle Simulator Application to ensure that it can connect with the IBM Watson IoT Foundation Service.

Navigate to **iot-onboard-vehicle-data-master/js/realtime.js** and open the file within a text editor like Sublime Text (or) Notepad++. Update the device credentials within the file(Check Comments for help) and save the file.

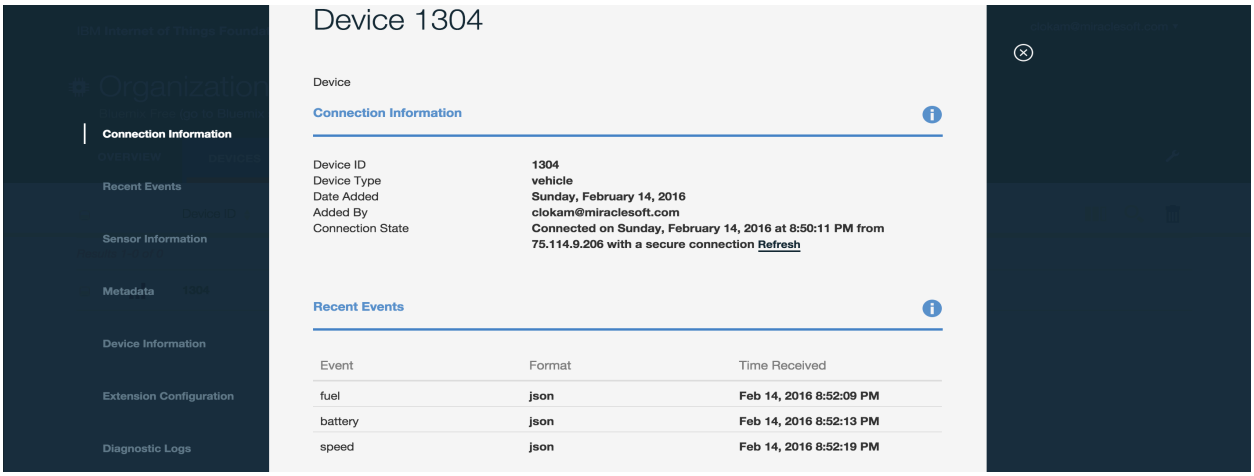
var orgId = "<your-Organization-ID>"
var deviceType = "<your-device-Type>"
var deviceId = "<your-device-ID>"
var deviceToken = "<your-device-Token>"



Once the file is saved navigate back to your index.html file and restart the simulator. Enter your name and start the simulator. You should now be able to see that the device is connected within Bluemix.

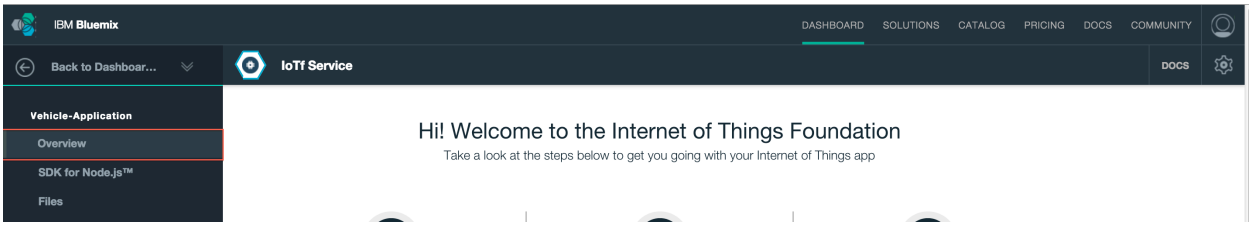


You can test the connection by sending sample events from the simulator. Then go back to the Dashboard and click on your device. You should be able to see a list of the most recent events from the simulator.

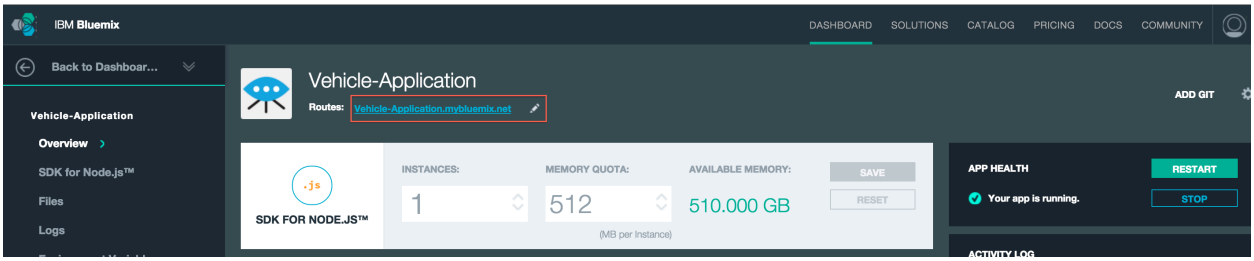


#7 | Creating your Node Red Flow

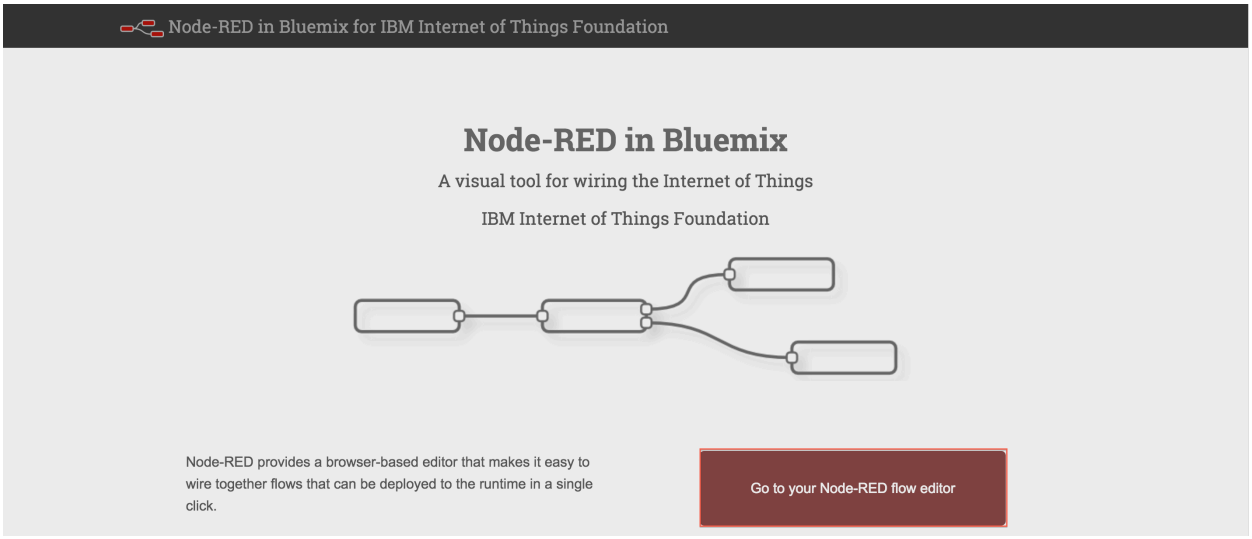
Next we will create a flow within NodeRed, which takes the simulator data, parses it and creates corresponding actions for the data. Go back to your Bluemix Application and click on overview.



Below the Application Name, click on the URL for the application.

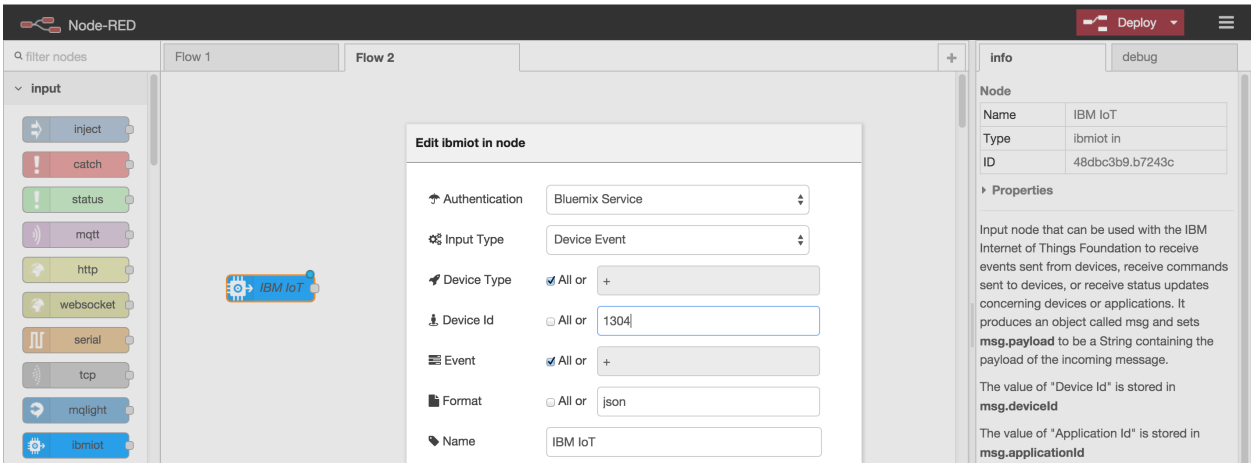


On the web page select the **"Go to your Node-RED flow editor"**.



Ignore the default flow that is given in Node Red and create your own new flow by clicking the "+" symbol in the right side corner. Rename it by right-clicking on the name. For example, **"Vehicle Cimulator Data Flow"**.

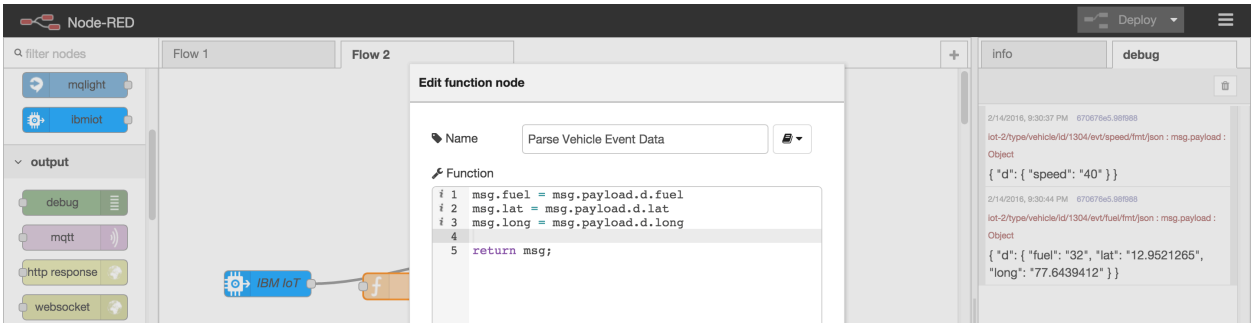
Select the **"ibmiot"** node from the **"input"** section in the left side nodes panel. Double Click on the node and give the **"Authentication"** as **"Bluemix Service"** and enter the **"Device ID"** as your Device ID.



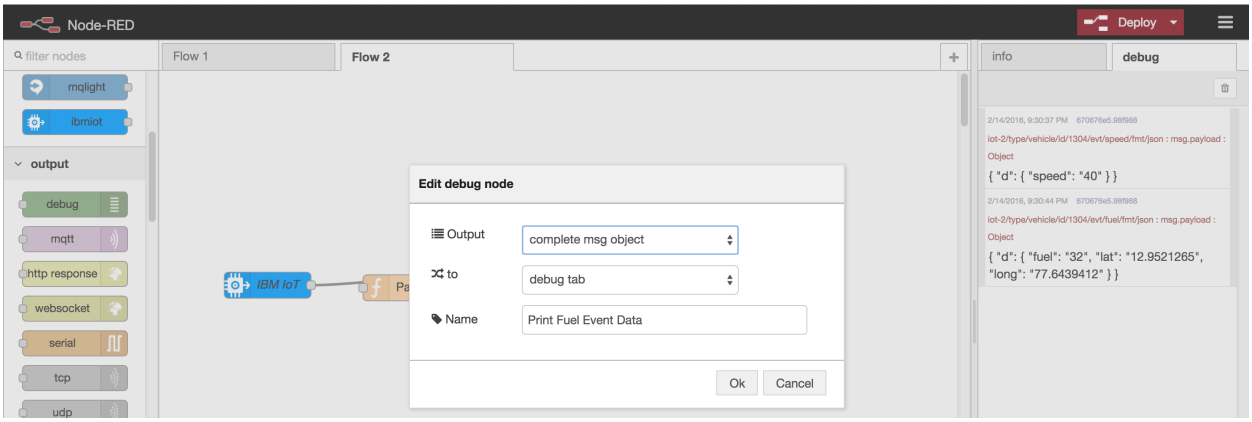
Add a Function Node to filter the event data JSON. Name the node as **"Parse Vehicle Event Data"** and add the below code to the node.

msg.fuel = msg.payload.d.fuel
msg.lat = msg.payload.d.lat
msg.long = msg.payload.d.long

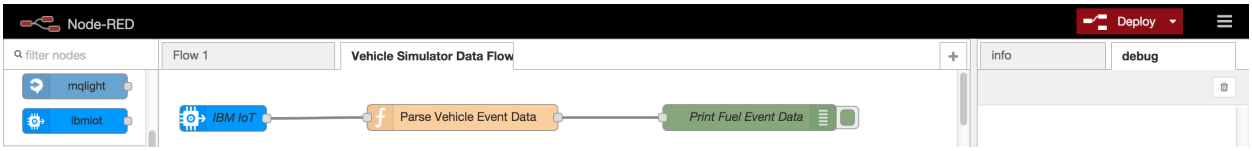
return msg;



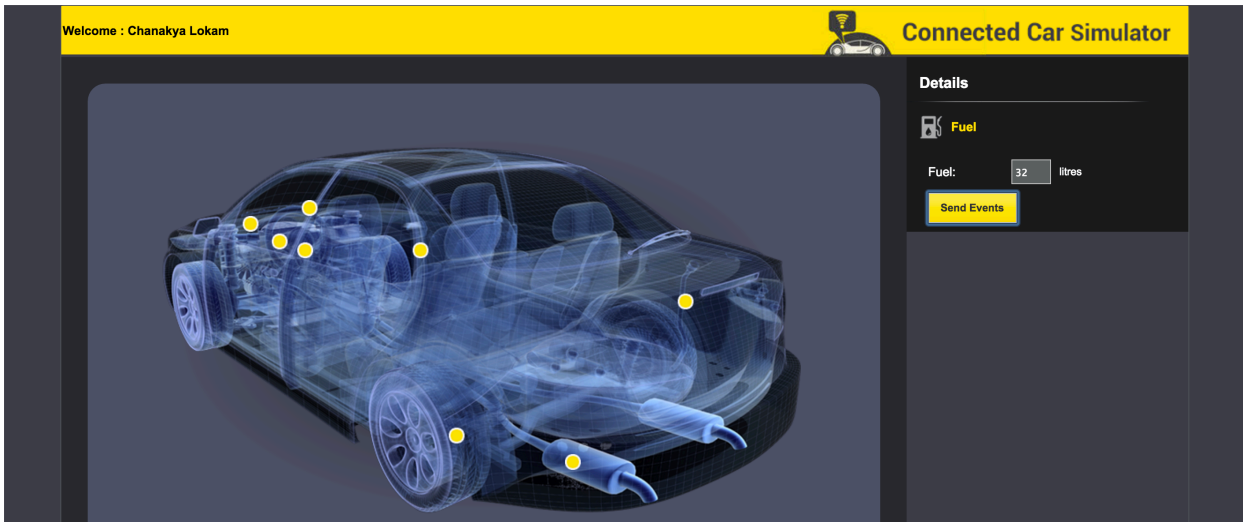
Now add a debug node and name it as **"Print Fuel Event Data".**



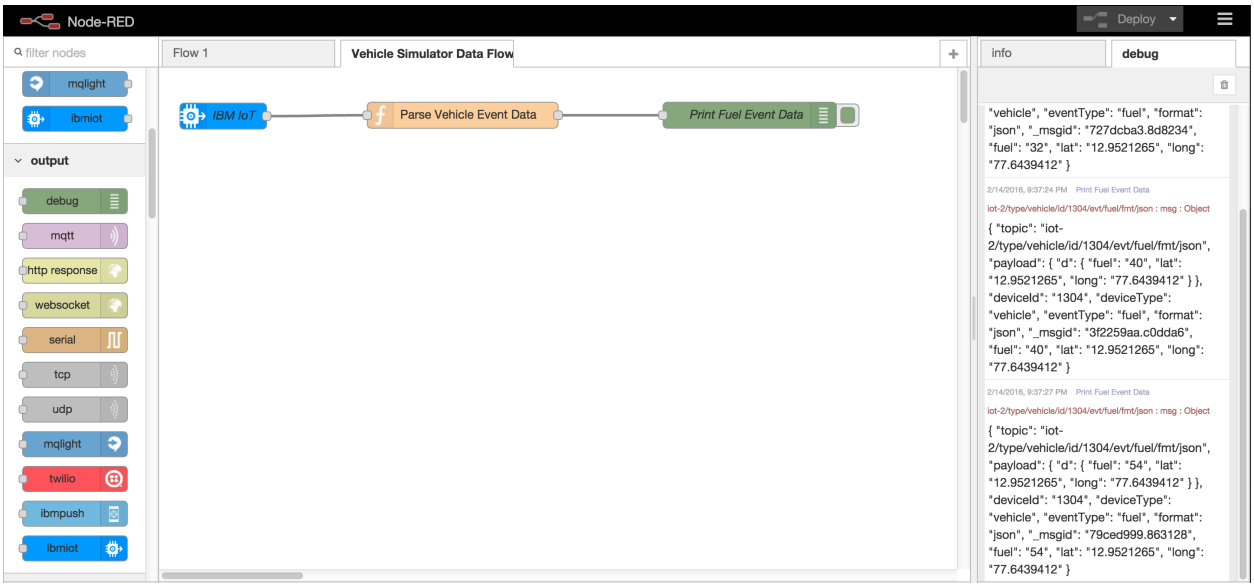
Connect the ibmiot node to the function node and the function node to the debug node and then deploy the entire flow.



Now go back to your simulator, and send 2-3 Fuel Events with varying Fuel values.



Head back to Node Red and check in the debug column to see the event data.

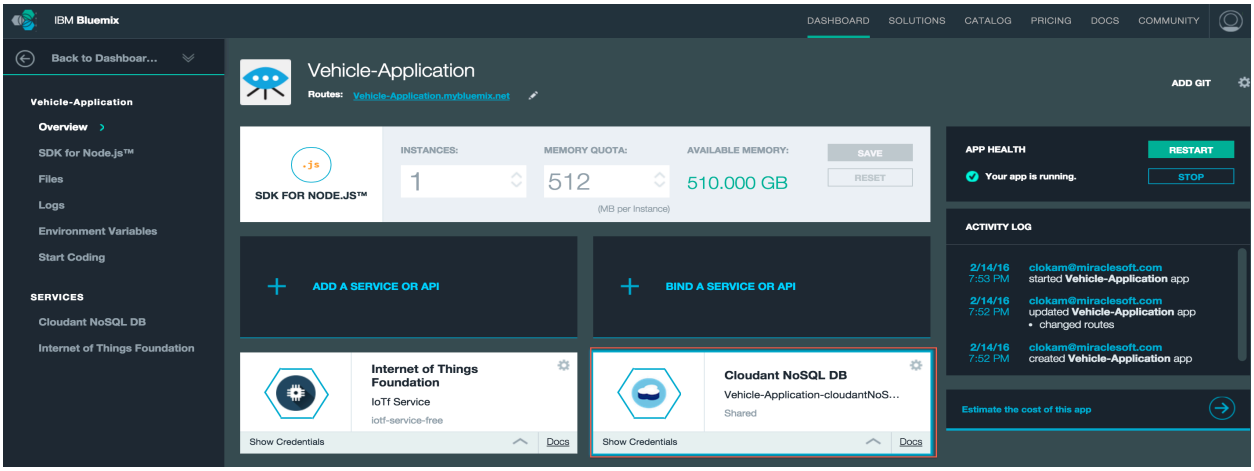


Note that we have built the flow specifically for Vehicle Fuel Events. You can customize the flow to work for more complex events and to even handle all the events that are coming from the Vehicle Simulator.

#8 | Storing into Cloudant DB

The final step will to ensure that the Vehicle Fuel data is being persisted into our Cloudant DB. For this we will first need to create our DB within Cloudant.

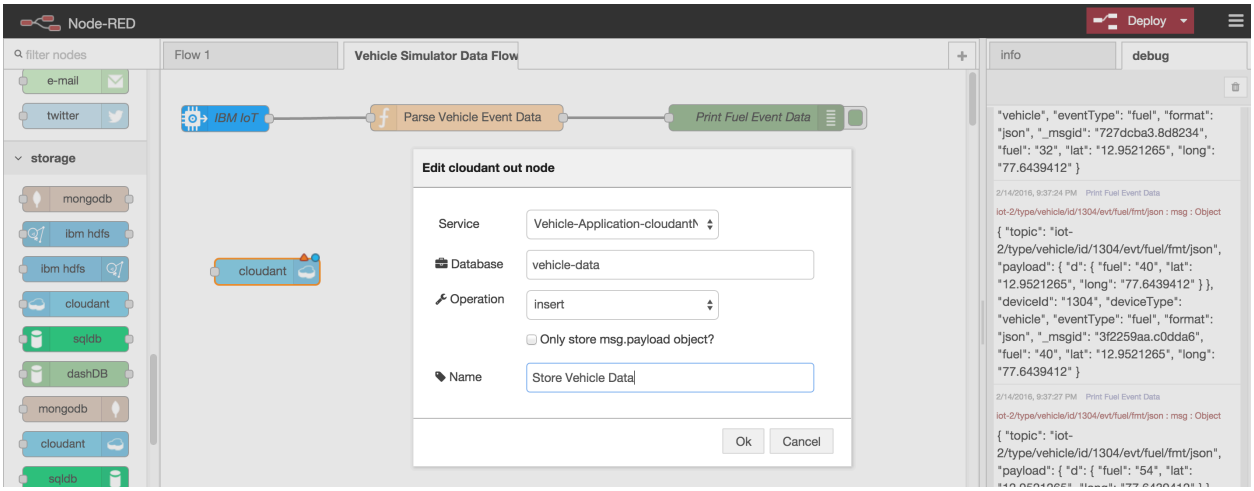
In Bluemix, on your application overview select and click the Cloudant NoSQL DB Service which is available by default.



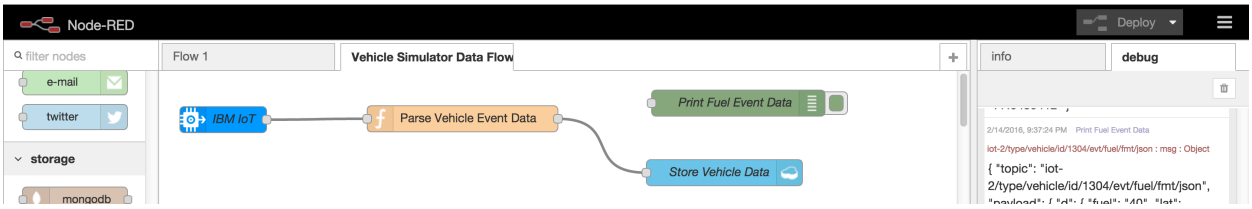
Click on the **“Launch”** button to take you to the Cloudant Dashboard.

On the dashboard click the “Create Database” button on the top right and create your new DB such as “vehicle-data”.

Now come back to your Node Red flow and add a new **Cloudant OUT Node** from the **"Storage"** Section. Name the node as **"Store Vehicle Data"** and select the Cloudant Service, which you have in your Bluemix account in the **"service"** field. In the Database field give your DB name, which in case of this document is **"vehicle-data"**. Leave the operation as **insert**.



Remove the link between your function node and debug node. Now connect your function node to your new Cloudant Out Node. Deploy your Node Red Flow once again.



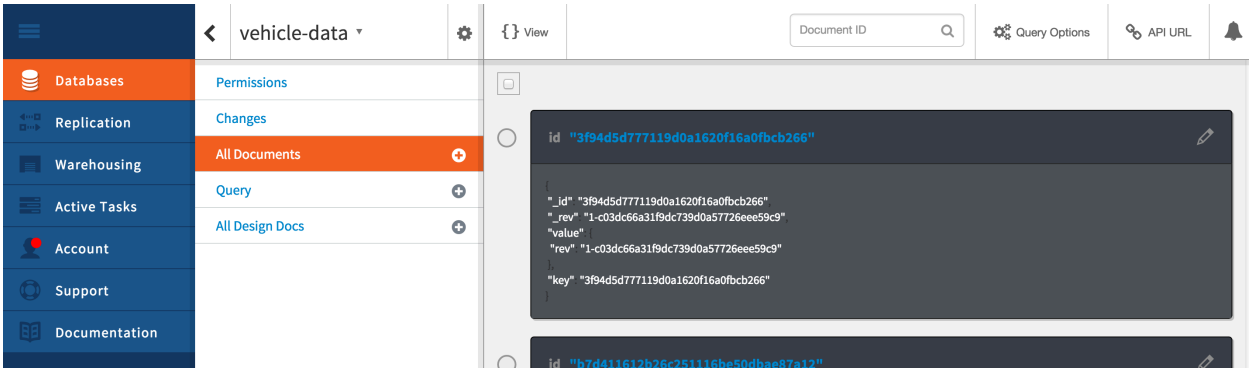
Note : By default the Cloudant Out Node stores the entire msg object into the Cloudant DB Documents

Go back to your Vehicle Simulator and send 2-3 fuel events with varying and random fuel values to check if they are stored in our DB (or) not.

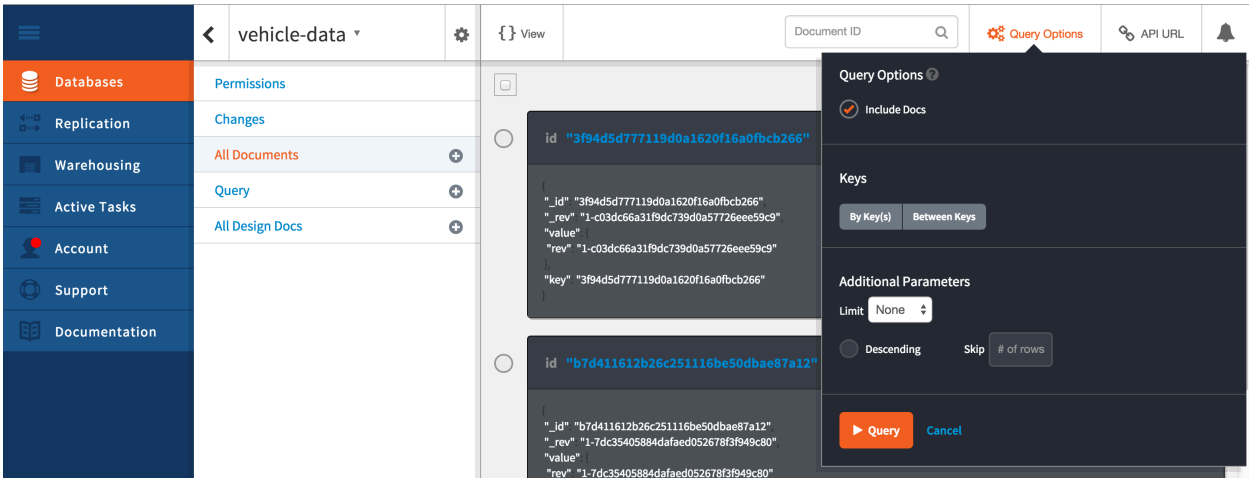
#9 | Viewing Vehicle Data in Cloudant

Go back to the Cloudant Dashboard that you opened earlier with your new DB, **"vehicle-data"**. If you have closed the window navigate through Bluemix to Cloudant and then to your newly created DB.

Click on **"All Documents"** to see the available documents within the Database.

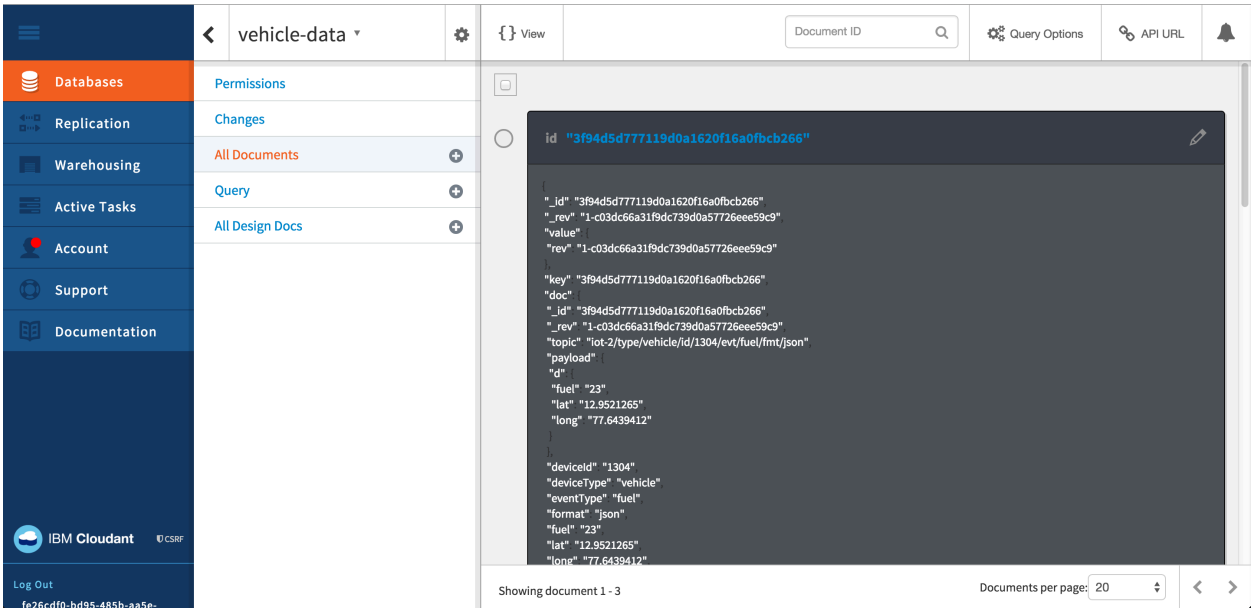


Click on **"Query Options"** in the top right and select the **"Include Docs"** option to enable Cloudant to show the entire object structure to us. Then click on "Query" to run the new query option.



This will show us the entire document object on the All Documents page directly without the need of having to go into each and every document.

Here you should now be able to see the various Fuel Events Data that you have sent form the simulator.



You have successfully connected the device to the cloud and created a flow to persist the fuel data into a NoSQL DB!