



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

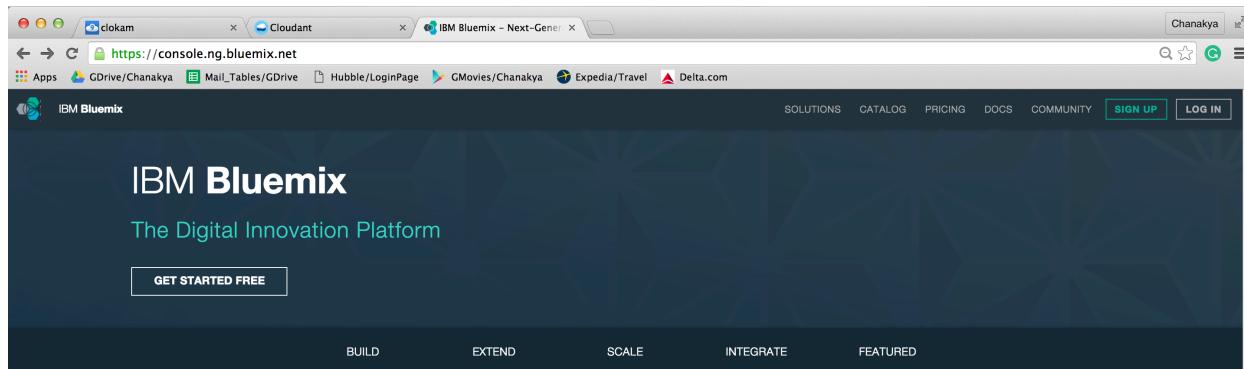
Download the tutorial at <https://github.com/MiracleLabs/IotApis101Lab>

Lab Steps

So, let us get started with the lab!

#1 | Get Access to IBM Bluemix

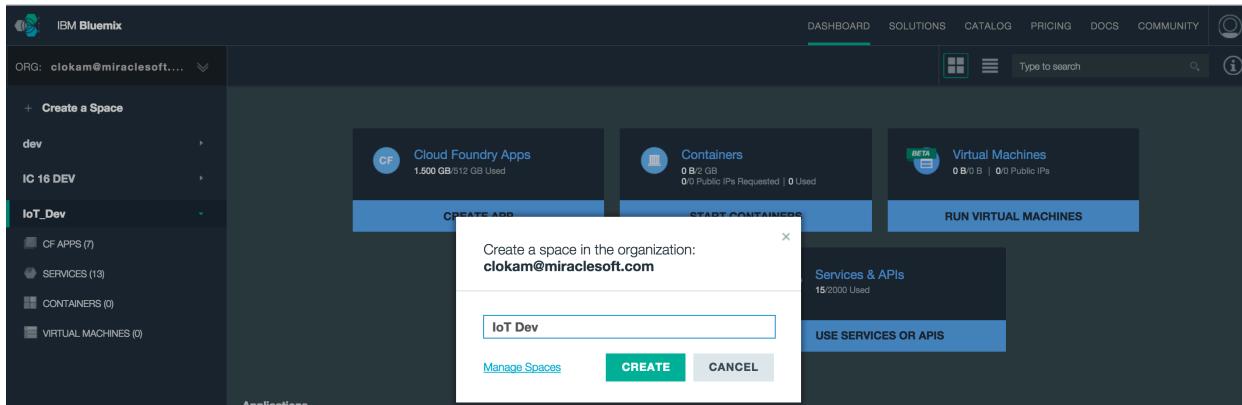
The first step of the lab will be to gain access to IBM Bluemix. If you already have an account then proceed by logging in from <http://bluemix.net> (or) sign up today.



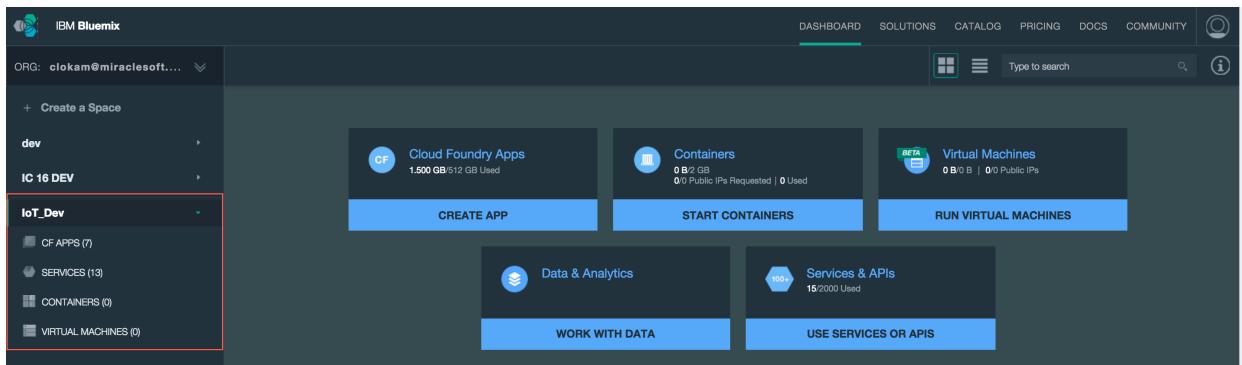
Once you have logged in you can see your dashboard and catalog screens. Go to your profile screen at the top and change the **Cloud Availability Region** to "**US South**". Note that the IoT Foundation Service is only available in that region.

A screenshot of the IBM Bluemix dashboard. On the left, there's a sidebar with organization details (ORG: clokam@miraclesoft....), a "+ Create a Space" button, and a list of spaces: dev, IC 16 DEV, and IoT_Dev. Under IoT_Dev, there are links for CF APPS (7), SERVICES (13), CONTAINERS (0), and VIRTUAL MACHINES (0). The main dashboard area shows Cloud Foundry Apps (1.500 GB/512 GB Used), Containers (0 B/0 GB, 0/0 Public IPs Requested | 0 Used), and Virtual Machines (0 B/0 B). There are buttons for CREATE APP, START CONTAINERS, and RUN VMS. Below these are sections for Data & Analytics (WORK WITH DATA) and Services & APIs (USE SERVICES OR APIs). On the right side, there's a sidebar for Chanakya Lokam (clokam@miraclesoft.com) with options for Account, Status, and Log Out. It also shows the current Region set to US South, with options for Sydney and United Kingdom. The sidebar includes sections for Notifications (You have no new notifications), Support (Get help, Submit an Idea), and Manage Organizations.

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**".

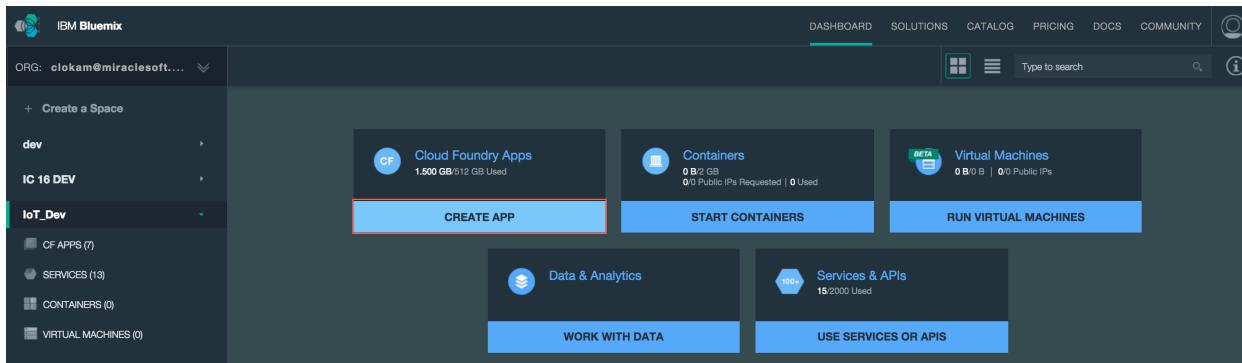


Note that the space creation will take a few minutes. Once the space has been created you will be able to see the new space on the left side panel.

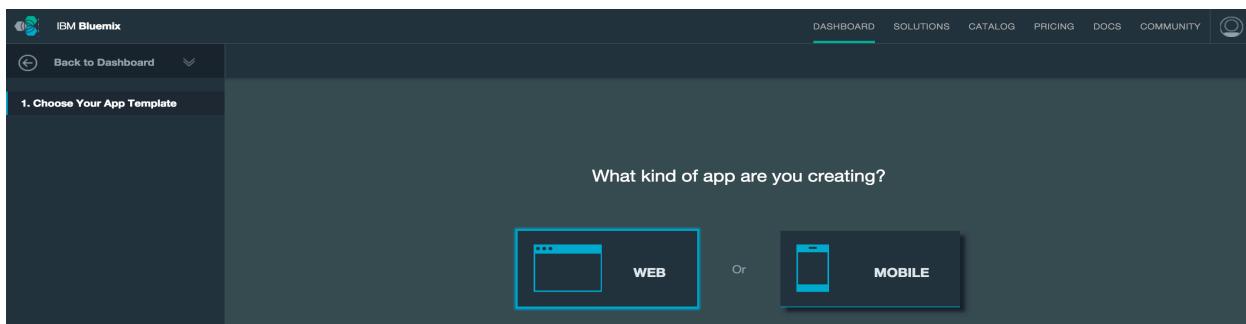


#2 | Create the IoT Starter Application and Service

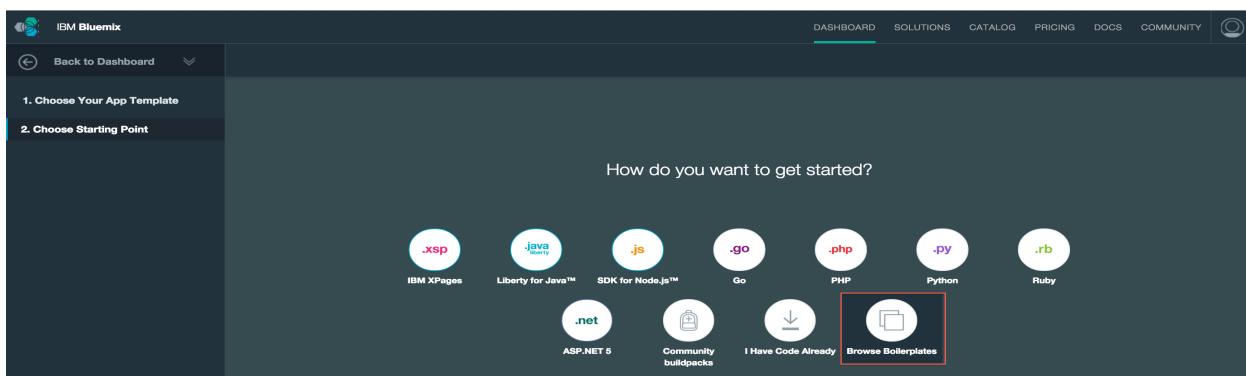
The next step will be to create the IoT Starter Application. Select the **Cloud Foundry Apps** option and click on "**Create App**".



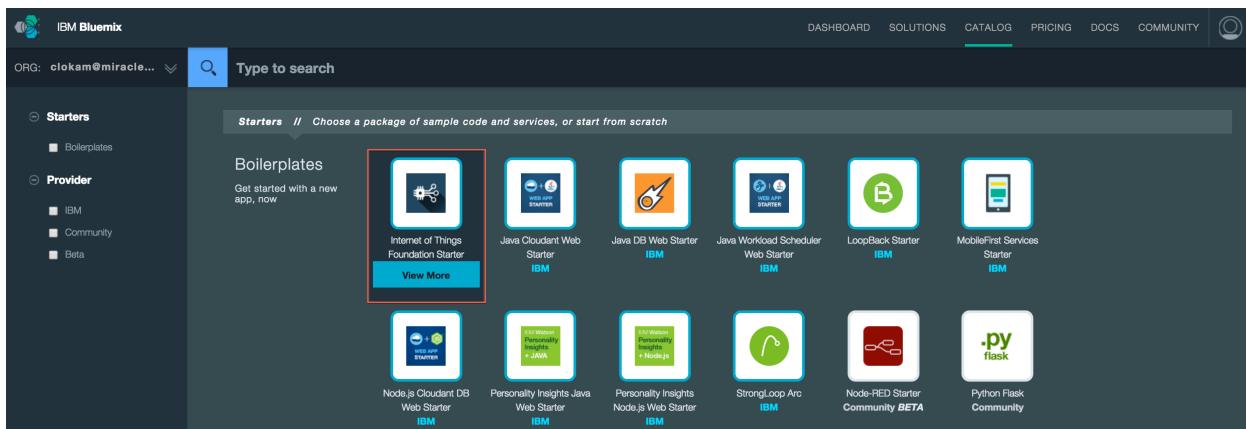
When prompted select “**Web Application**”.



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.

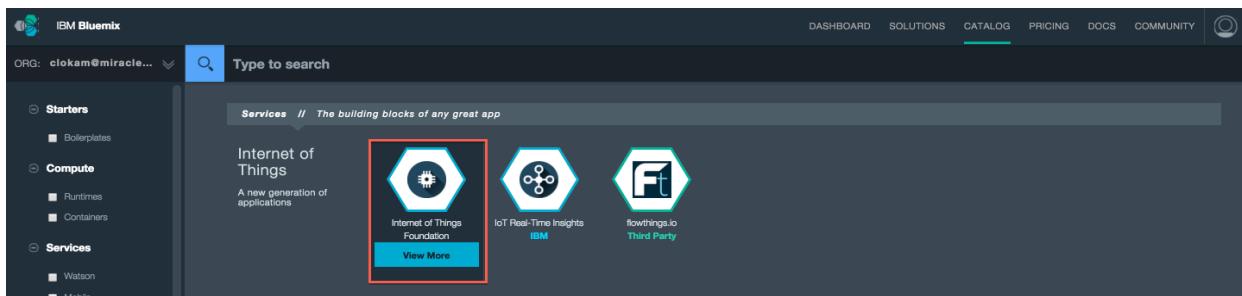
The screenshot shows the IBM Bluemix Catalog interface. On the left, there's a sidebar with a search bar and a list of categories like 'SOLUTIONS', 'CATALOG', 'PRICING', 'DOCS', and 'COMMUNITY'. Below the search bar, there's a link to 'Back to Boilerplates'. The main area displays a card for the 'Internet of Things Foundation Starter' boilerplate. The card includes a thumbnail icon, the name, a brief description, version (0.4.18), type (Boilerplate), and a 'VIEW DOCS' button. To the right of the card, there are two service icons: 'SDK for Node.js™' (with a JS logo) and 'Cloudant NoSQL DB'. A callout box points from the 'SDK for Node.js™' icon to a text block: 'Develop, deploy, and scale server-side JavaScript® apps with ease. The IBM SDK for Node.js™ provides enhanced performance, security, and serviceability.' Below this, there's a 'VIEW DOCS' button. Further down, there's a section titled 'Pick a plan' with a table showing a 'Default' plan: 'Run one or more apps free for 30 days (375 GB-hours free.)' at '\$0.07 USD/GB-Hour'. A note below the table says, 'This is a service plan for the IBM Bluemix Platform runtime.' To the right of the plan table, there's a form for creating an app. It includes fields for 'Space' (set to 'IoT_Dev'), 'Name' (set to 'Vehicle-Application'), 'Host' (set to 'Vehicle-Application'), 'Domain' (set to 'mybluemix.net'), and 'Selected Plan' (set to 'SDK for Node.js™ - Default'). There are dropdown menus for 'Cloudant NoSQL DB' (set to 'Shared') and a 'CREATE' button. The overall background is dark with light-colored cards and forms.

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

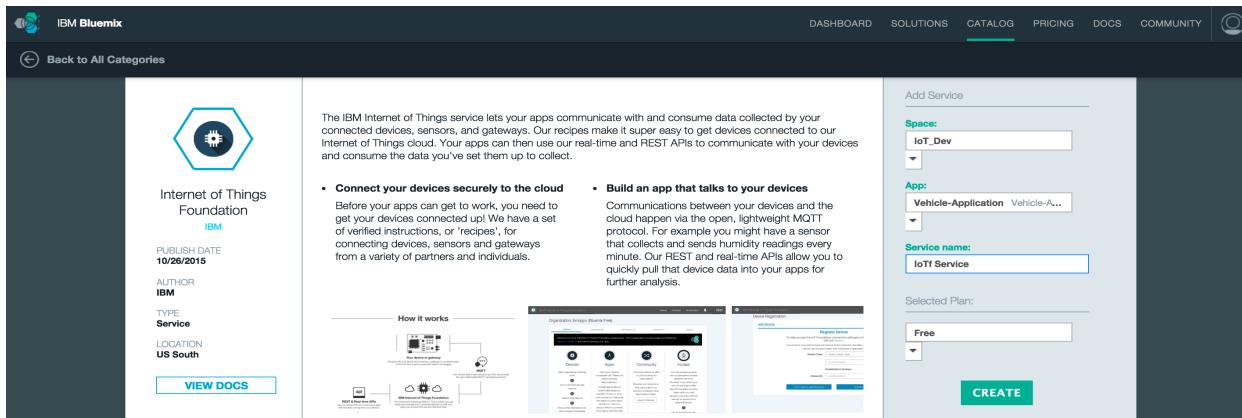
The screenshot shows the IBM Bluemix Dashboard for the 'Vehicle-Application'. The left sidebar has a 'Back to Dashboard...' link and a list of sections: 'Vehicle-Application', 'Overview', 'SDK for Node.js™', 'Files', 'Logs', 'Environment Variables', 'Start Coding >', and 'SERVICES' (with 'Cloudant NoSQL DB' listed). The main area has a header 'Your application is staging. <http://Vehicle-Application.mybluemix.net>'. Below it, there's a 'Getting Started with:' section for 'Internet of Things' (with a gear icon) and a 'Start coding with Internet of Things' section. The 'Start coding' section contains numbered steps: 1. After your application has started, click on the **Routes URL** or enter the following URL in a browser: <http://<yourhost>.mybluemix.net>. Step 1 has a small icon with a 1. 2. Click [Go to your Node-RED flow editor](#). You will see a ready-made flow that can process temperature readings from a simulated device. Step 2 has a small icon with a 2. Below these steps, there's a 'Customizing your Node-RED instance' section with a note about installing the Cloud Foundry command line interface. The overall background is dark with light-colored cards and forms.

#3 | Adding IoT Service to your Application

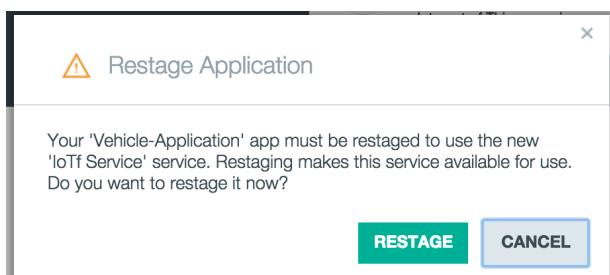
Go to the “Catalog” on the top right menu and select “Internet of Things” in the left side menu. Within the given options select the “**Internet of Things Foundation Service**”.



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.

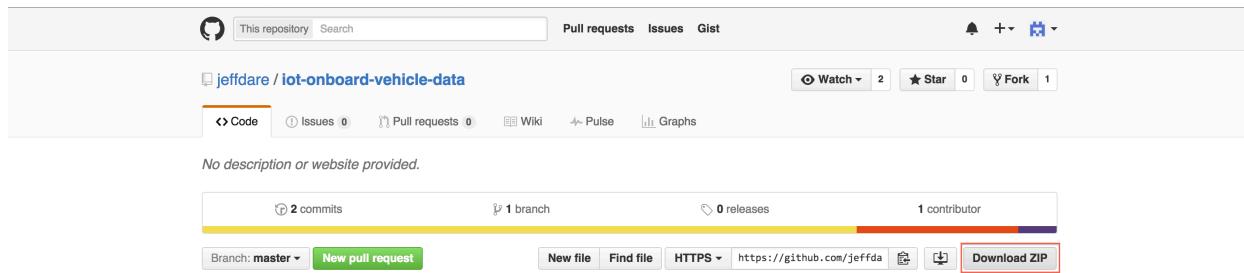


Note : This process will once again take a few minutes

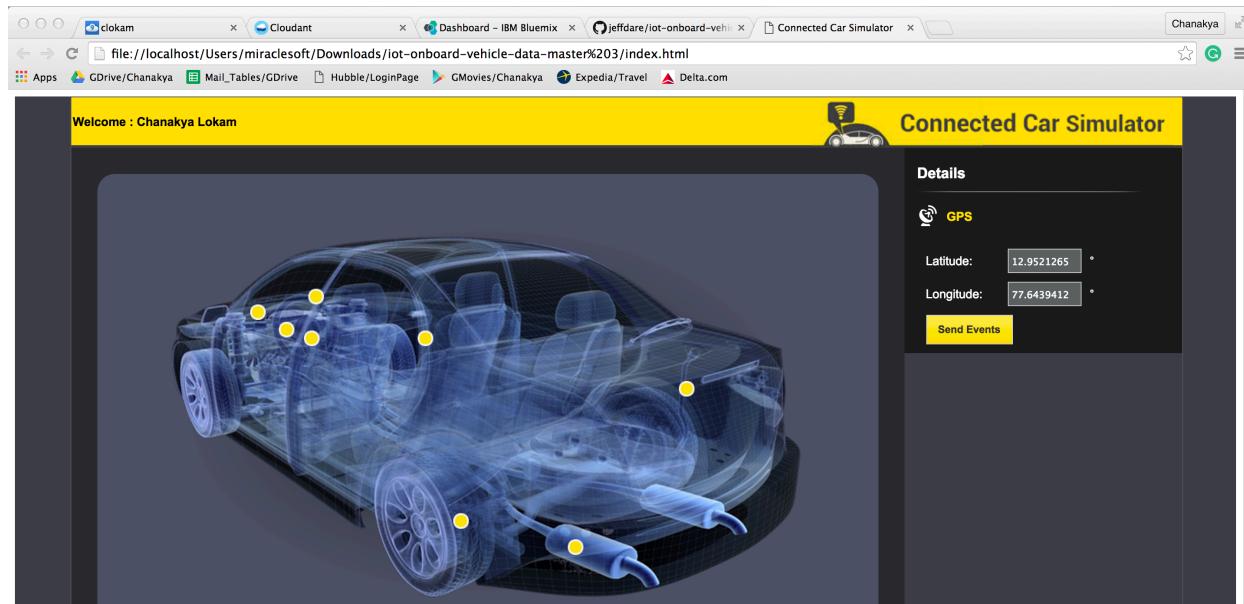
#4 | Get the OBD II Vehicle Simulator

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>

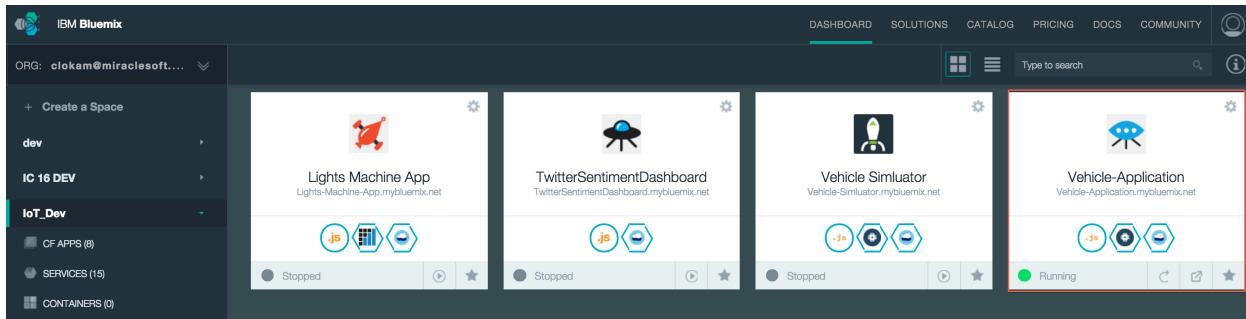


Once you have downloaded the files navigate to the **iot-onboard-vehicle-data-master** folder and open the **index.html** file with a browser. When prompted give your name, and then you can see the car simulators with data such as Fuel, GPS and more. (See Below)

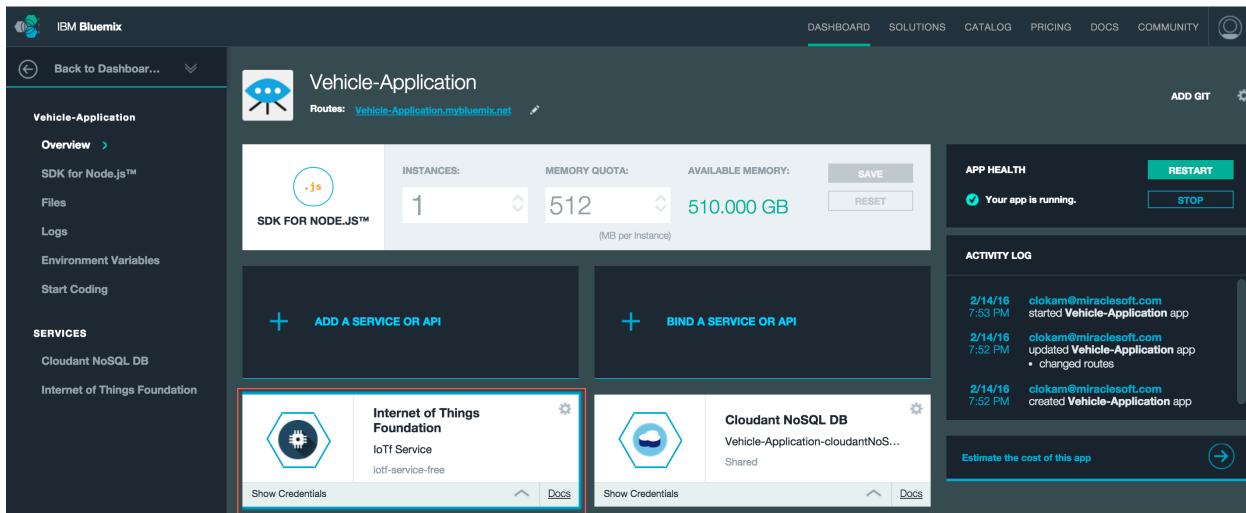


#5 | Register your Vehicle with the IoTF Service

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.

IBM Bluemix

Back to Dashboard... IoT Service

Vehicle-Application

- Overview
- SDK for Node.js™
- Files
- Logs
- Environment Variables
- Start Coding

SERVICES

- Cloudant NoSQL DB
- Internet of Things Foundation >

Hi! Welcome to the Internet of Things Foundation

Take a look at the steps below to get you going with your Internet of Things app.

Connect your devices

Use our [recipes](#) to find out how to add your devices. We work with partners and have sample connection recipes for many devices.

Launch the Internet of Things Foundation dashboard and add your devices by clicking the 'Add Device' button under the 'Devices' tab.

Learn how to build your app

When you have added your devices, you can come back to Bluemix to start building your app using your real-time and historical device data.

Read the docs to find out how to make the most out of your app.

[Go to docs](#)

Learn how to extend your app

Use other Bluemix services to extend your app to start creating a great Internet of Things app.

Here are some of the services you could use:

- Twilio Third Party
- Cloudant NoSQL DB IBM
- Dash DB IBM
- Watson Assistant
- Watson Tone Analyzer
- Watson Visual Recognition

Click on “+Add a device”.

IBM Internet of Things Foundation Quickstart Service Status Documentation Blog clokam@miraclesoft.com ▾

Organization ID: kfrb52

Bluemix Free ([go to Bluemix service](#))

OVERVIEW DEVICES ACCESS USAGE

Devices

DEVICES

Devices registered

+ Add a device

There are no devices in your organization.
[Click here to add new devices](#)

Usage

THIS MONTH
0.00 Average devices connected

PREVIOUS MONTH
0.00 Average devices connected

You will be prompted to create a new device type.

IBM Internet of Things Foundation

Organization

Bluemix Free ([go to Bluemix service](#))

Choose Device Type

OVERVIEW DEVICES Device Info

Metadata

Security

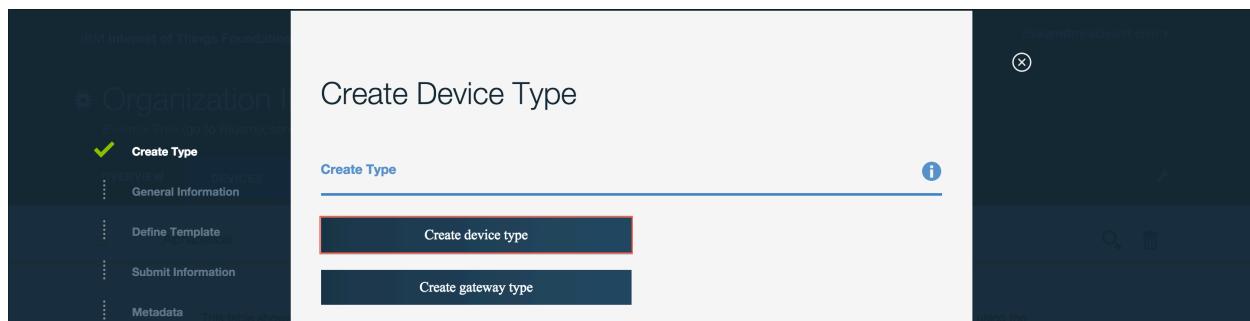
Summary

Add Device

Choose Device Type

Create device type

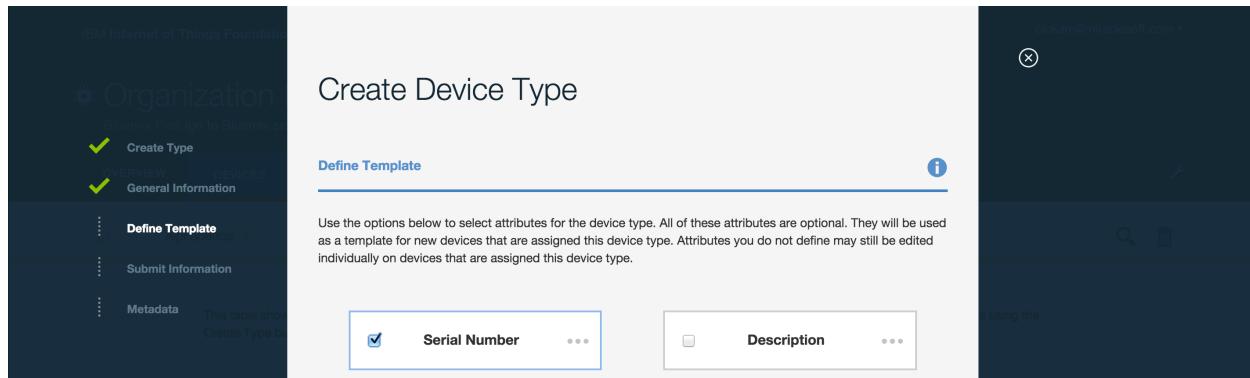
In the next dialog select “Create Device Type” as we want to add a device.



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.

The screenshot shows the 'Add Device' process in the IBM IoT Foundation. The left sidebar has a 'Security' section selected. The main area shows the 'Security' step with two options: 'Auto-generated authentication token' (selected) and 'Custom authentication token'. A note explains that the service generates a 18-character alphanumeric token.

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.**

The screenshot shows the 'Your Device Credentials' page for device ID 1304. The left sidebar has a 'Device Information' section selected. The main area displays the device credentials, including the Organization ID (kfrb52), Device Type (vehicle), Device ID (1304), Authentication Method (token), and Authentication Token (vtlzdXhc++aD0z+lmN).

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

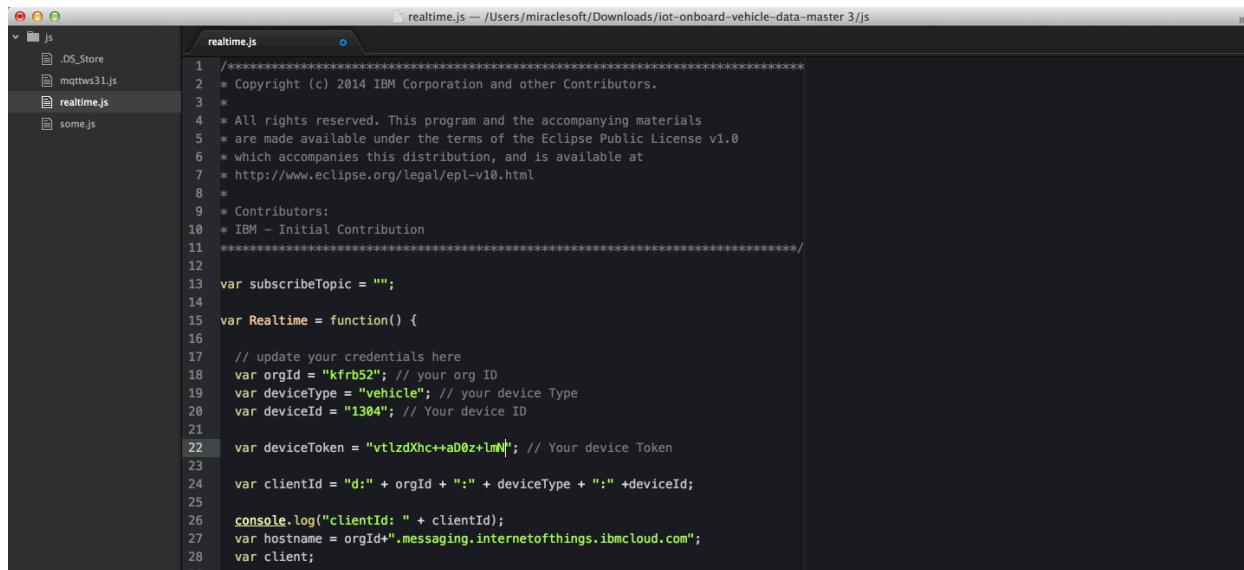
The screenshot shows a table of devices. The first row is highlighted with a yellow background. It contains columns for Device ID (1304), Device Type (vehicle), Class ID (Device), Date Added (Feb 14, 2016 8:35:04 PM), and Location (dropdown menu). There are also icons for edit, delete, and search.

#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>"
```



```
realtime.js — /Users/miraclesoft/Downloads/iot-onboard-vehicle-data-master 3 /js  
realtime.js  
1 /*****  
2 * Copyright (c) 2014 IBM Corporation and other Contributors.  
3 *  
4 * All rights reserved. This program and the accompanying materials  
5 * are made available under the terms of the Eclipse Public License v1.0  
6 * which accompanies this distribution, and is available at  
7 * http://www.eclipse.org/legal/epl-v10.html  
8 *  
9 * Contributors:  
10 * IBM - Initial Contribution  
*****/  
11  
12 var subscribeTopic = "";  
13  
14 var Realtime = function() {  
15  
16 // update your credentials here  
17 var orgId = "kfrb52"; // your org ID  
18 var deviceType = "vehicle"; // your device Type  
19 var deviceId = "1304"; // Your device ID  
20  
21 var deviceToken = "vtlzdXhc++aD0z+lmN"; // Your device Token  
22  
23 var clientId = "d:" + orgId + ":" + deviceType + ":" + deviceId;  
24  
25 console.log("clientId: " + clientId);  
26 var hostname = orgId+".messaging.internetofthings.ibmcloud.com";  
27 var client;
```

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.

Device ID	Device Type	Class ID	Date Added	Location	Actions
Results 1-0 of 0					
1304	vehicle	Device	Feb 14, 2016 8:49:46 PM		

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.

The screenshot shows the IBM Internet of Things Foundation dashboard. On the left, there's a sidebar with links like 'Organization', 'Recent Events', 'Device ID', 'Sensor Information', 'Metadata', 'Device Information', 'Extension Configuration', and 'Diagnostic Logs'. The main area is titled 'Device 1304' and contains two sections: 'Connection Information' and 'Recent Events'. Under 'Connection Information', it shows the Device ID (1304), Device Type (vehicle), Date Added (Sunday, February 14, 2016), Added By (clkam@miraclesoft.com), and Connection State (Connected on Sunday, February 14, 2016 at 8:50:11 PM from 75.114.9.206 with a secure connection). Under 'Recent Events', there are three entries: 'fuel' (json, Feb 14, 2016 8:52:09 PM), 'battery' (json, Feb 14, 2016 8:52:13 PM), and 'speed' (json, Feb 14, 2016 8:52:19 PM).

#7 | Creating your Node Red Flow

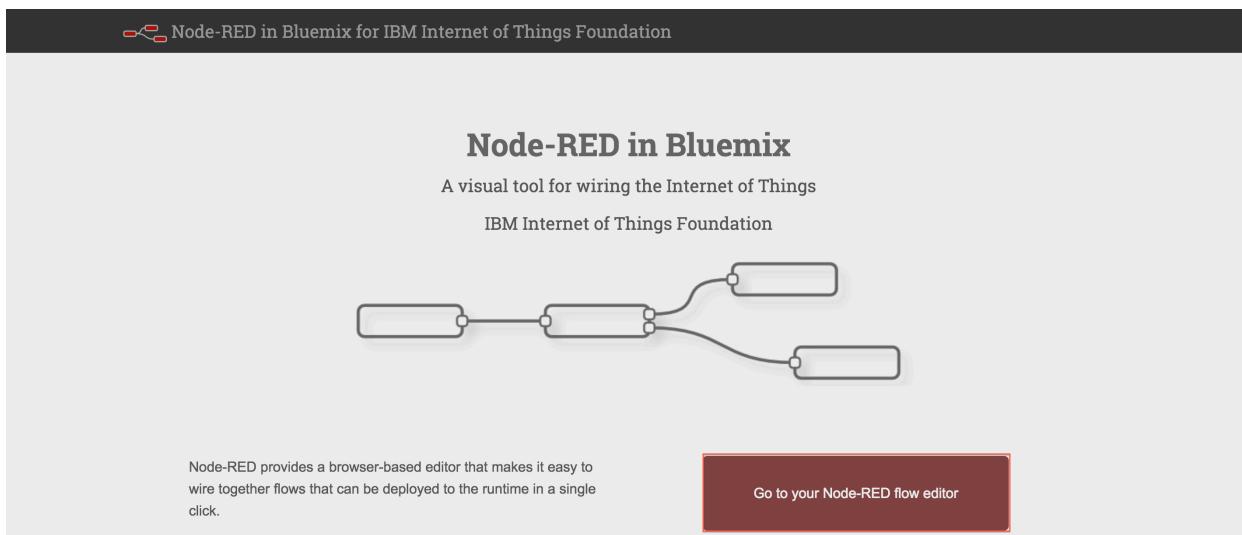
Next we will create a flow within NodeRed, which takes the simulator data, parses it and creates corresponding actions for the data. Go back to your Bluemix Application and click on overview.

The screenshot shows the IBM Bluemix IoT Service application overview page. The sidebar on the left has 'Vehicle-Application' selected, with 'Overview' highlighted. The main content area says 'Hi! Welcome to the Internet of Things Foundation' and 'Take a look at the steps below to get you going with your Internet of Things app'. There are several tabs at the top: DASHBOARD, SOLUTIONS, CATALOG, PRICING, DOCS, and COMMUNITY.

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

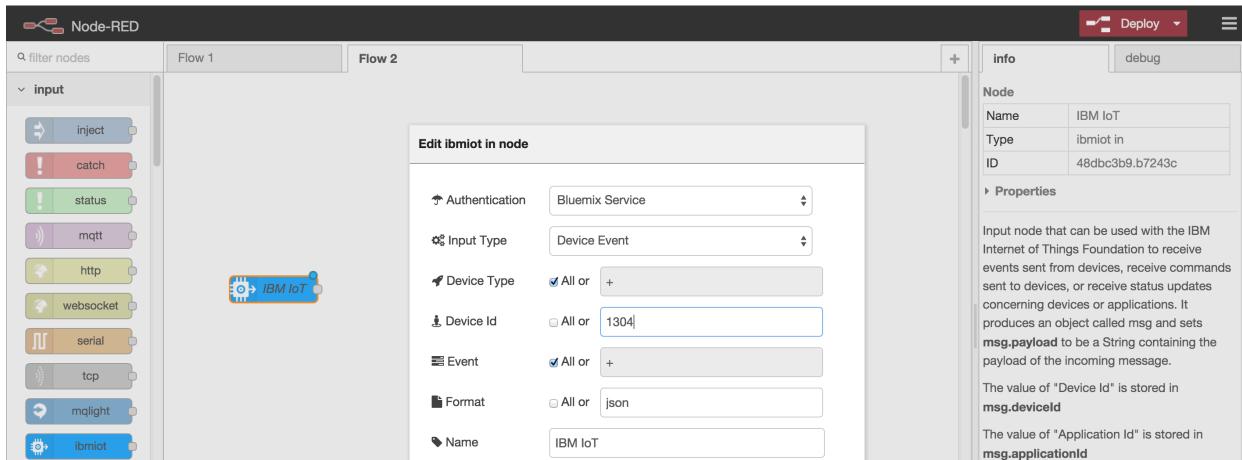
The screenshot shows the IBM Bluemix application overview page for 'Vehicle-Application'. The sidebar on the left has 'Vehicle-Application' selected, with 'Overview' highlighted. The main content area shows the application route as 'Vehicle_Application.mybluemix.net'. It has fields for 'INSTANCES' (set to 1), 'MEMORY QUOTA' (512 MB), and 'AVAILABLE MEMORY' (510.000 GB). A 'SAVE' button is present. On the right, there's an 'APP HEALTH' section stating 'Your app is running.' and a 'STOP' button. At the bottom, there's an 'ACTIVITY LOG' section.

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 Simulator Data Flow**”.

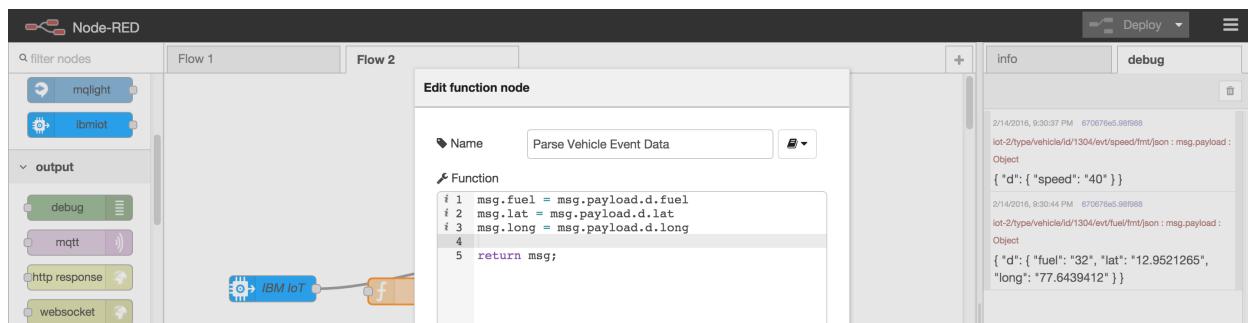
Select the “**ibmiot**” node from the “**input**” section in the left side nodes panel. Double Click on the node and give the “**Authentication**” as “**Bluemix Service**” and enter the “**Device ID**” as your Device ID.



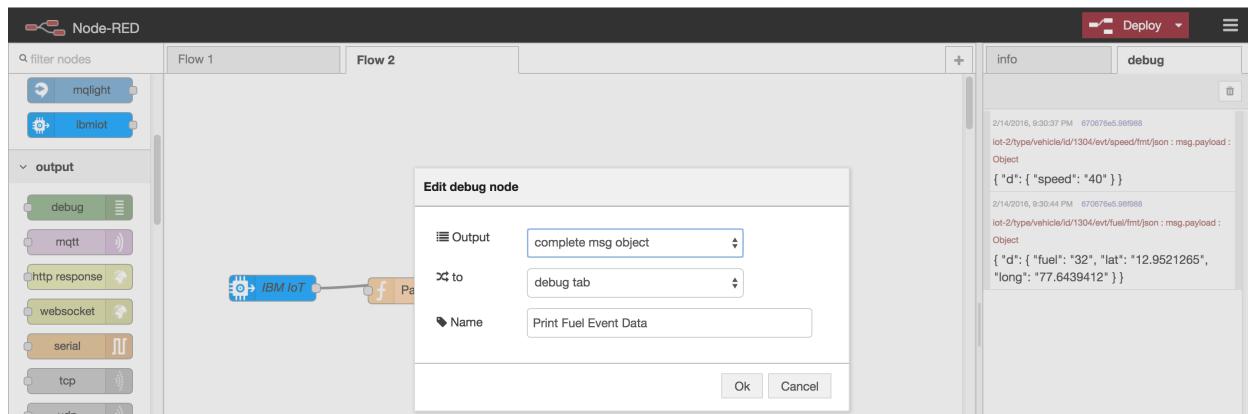
Add a Function Node to filter the event data JSON. Name the node as “**Parse Vehicle Event Data**” and add the below code to the node.

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

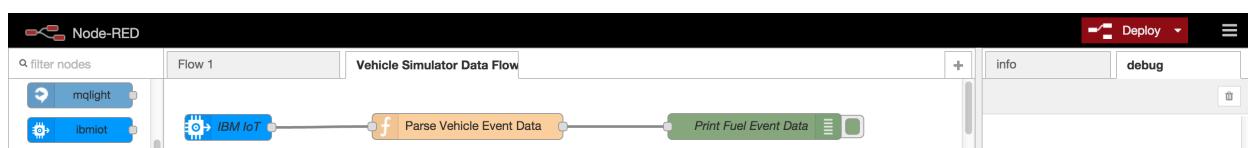
```
return msg;
```



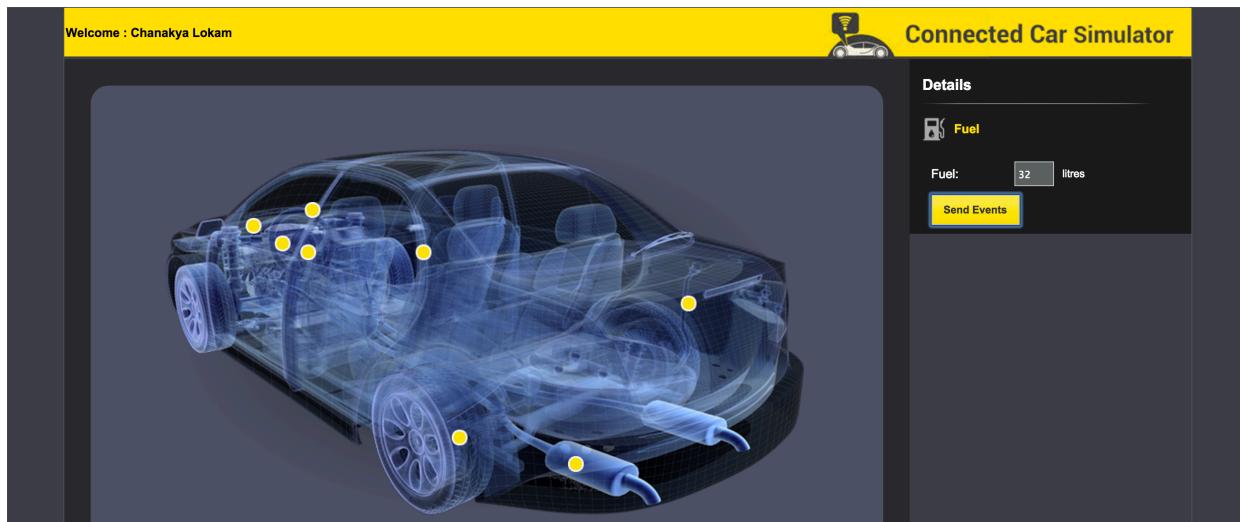
Now add a debug node and name it as “**Print Fuel Event Data**”.



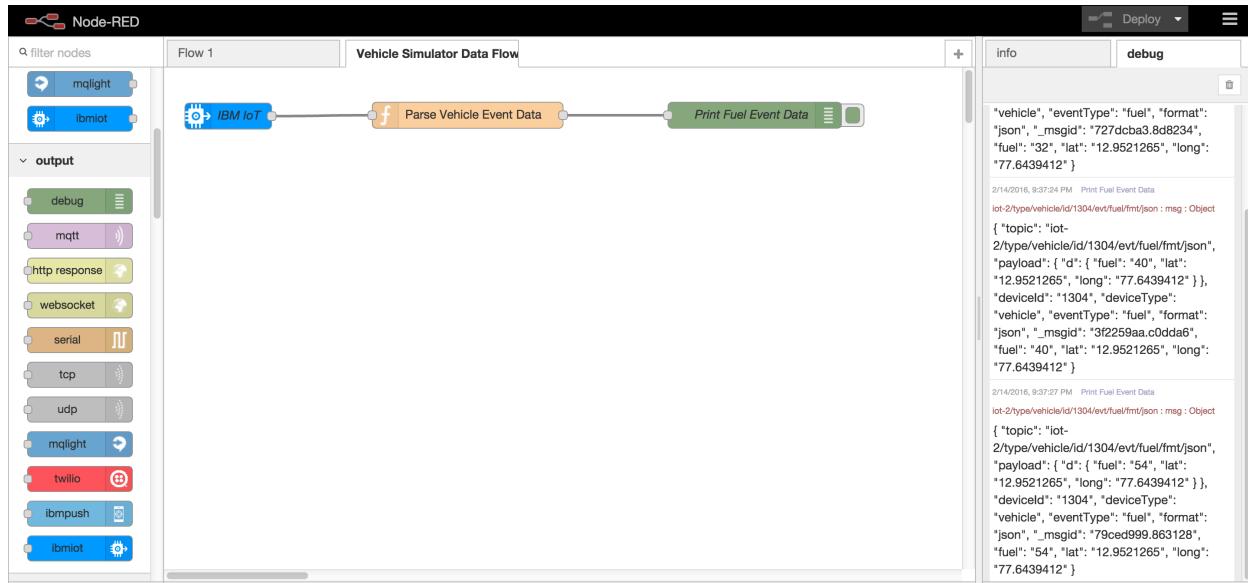
Connect the ibmiot node to the function node and the function node to the debug node and then deploy the entire flow.



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 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.

The screenshot shows the IBM Bluemix Application Overview dashboard for the 'Vehicle-Application' service. The service summary indicates 1 instance, 512 MB memory quota, and 51.000 GB available memory per instance. Below the summary are two cards: 'ADD A SERVICE OR API' and 'BIND A SERVICE OR API'. Under the 'SERVICES' section, the 'Cloudant NoSQL DB' service is listed and highlighted with a red box. To the right, the 'APP HEALTH' section shows the app is running, and the 'ACTIVITY LOG' section lists recent events including the creation of the app and changes to routes. A 'LAUNCH' button is located at the bottom right of the main service card.

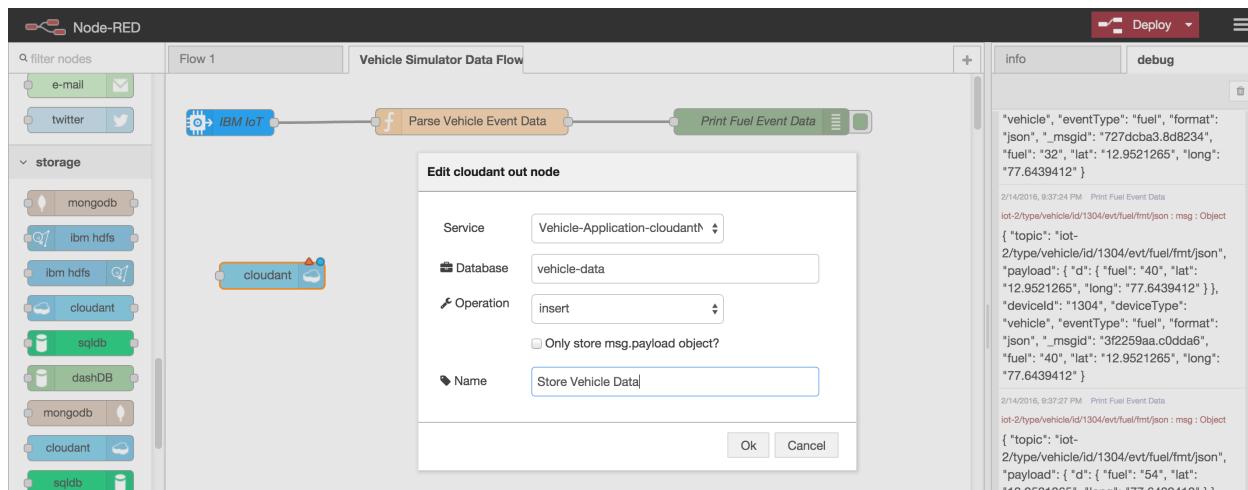
Click on the “Launch” button to take you to the Cloudant Dashboard.

The screenshot shows the IBM Bluemix Application Overview dashboard for the 'Vehicle-Application-cloudantNoSQLDB' service. The service summary indicates the service is running. Below the summary is a card for the 'Cloudant NoSQL DB' service, which is highlighted with a red box. A 'LAUNCH' button is located at the bottom right of the main service card.

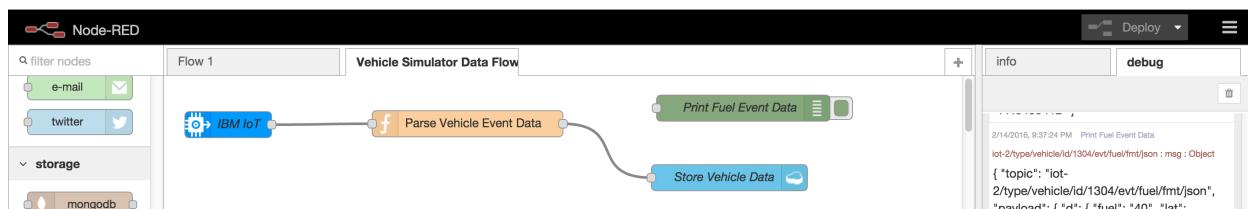
On the dashboard click the “Create Database” button on the top right and create your new DB such as “vehicle-data”.

The screenshot shows the Cloudant NoSQL DB Dashboard. On the left is a sidebar with navigation links: Databases (highlighted in orange), Replication, Warehousing, Active Tasks, and Account. The main area is titled 'Databases' and shows a table of existing databases. A 'Create Database' button is located at the top right of the database list, highlighted with a red box. A modal window is open, showing the 'Create Database' form with the name 'vehicle-data' entered. Other buttons in the modal include 'Create' and a cancel icon.

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.

The screenshot shows the Cloudant Dashboard interface. On the left, a sidebar lists various database management options like Databases, Replication, Warehousing, Active Tasks, Account, Support, and Documentation. The 'Databases' section is highlighted. In the center, the 'vehicle-data' database is selected. The top navigation bar includes 'View', 'Document ID', 'Query Options', 'API URL', and a bell icon. Below the navigation, there are tabs for 'Permissions', 'Changes', 'All Documents', 'Query', and 'All Design Docs'. The 'All Documents' tab is active and highlighted in orange. The main content area displays two document entries. The first document's details are expanded, showing its ID, revision, value, and key. The second document's ID is also visible below it.

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 screenshot shows the same Cloudant dashboard as above, but with the 'Query Options' dialog box overlaid on the right side. The 'Include Docs' checkbox is checked, indicating that the query will return the full document structure. The 'Query' button at the bottom of the dialog is highlighted in orange, ready to be clicked. The rest of the interface remains the same, showing the 'vehicle-data' database and the two document entries.

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 from the simulator.

The screenshot shows the IBM Cloudant interface. On the left, a sidebar lists databases, replication, warehousing, active tasks, account, support, and documentation. The 'vehicle-data' database is selected. The main area shows a list of documents under 'All Documents'. One document is selected, showing its details. The document ID is "3f94d5d777119d0a1620f16a0fbcb266". The document content is as follows:

```
{  
  "_id": "3f94d5d777119d0a1620f16a0fbcb266",  
  "_rev": "1-c03dc66a31f9dc739d0a57726eee59c9",  
  "value": {  
    "rev": "1-c03dc66a31f9dc739d0a57726eee59c9"  
  },  
  "key": "3f94d5d777119d0a1620f16a0fbcb266",  
  "doc": {  
    "_id": "3f94d5d777119d0a1620f16a0fbcb266",  
    "_rev": "1-c03dc66a31f9dc739d0a57726eee59c9",  
    "topic": "iot-2/type/vehicle/id/1304/evt/fuel/mt/json",  
    "payload": {  
      "d"  
      "fuel": "23"  
      "lat": "12.9521265",  
      "long": "77.6439412"  
    },  
    "deviceID": "1304",  
    "deviceType": "vehicle",  
    "eventType": "fuel",  
    "format": "json",  
    "fuel": "23",  
    "lat": "12.9521265",  
    "long": "77.6439412"  
  }  
}
```

At the bottom, it says "Showing document 1 - 3" and "Documents per page: 20".

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