A Project Report

On

Predictive Maintenance of Industrial Motors

By

Gilla Nikitha

Vaishnavi Kulkarni

Nisha Mohd

S Arika

As an intern at smartinternz.com@risp2020

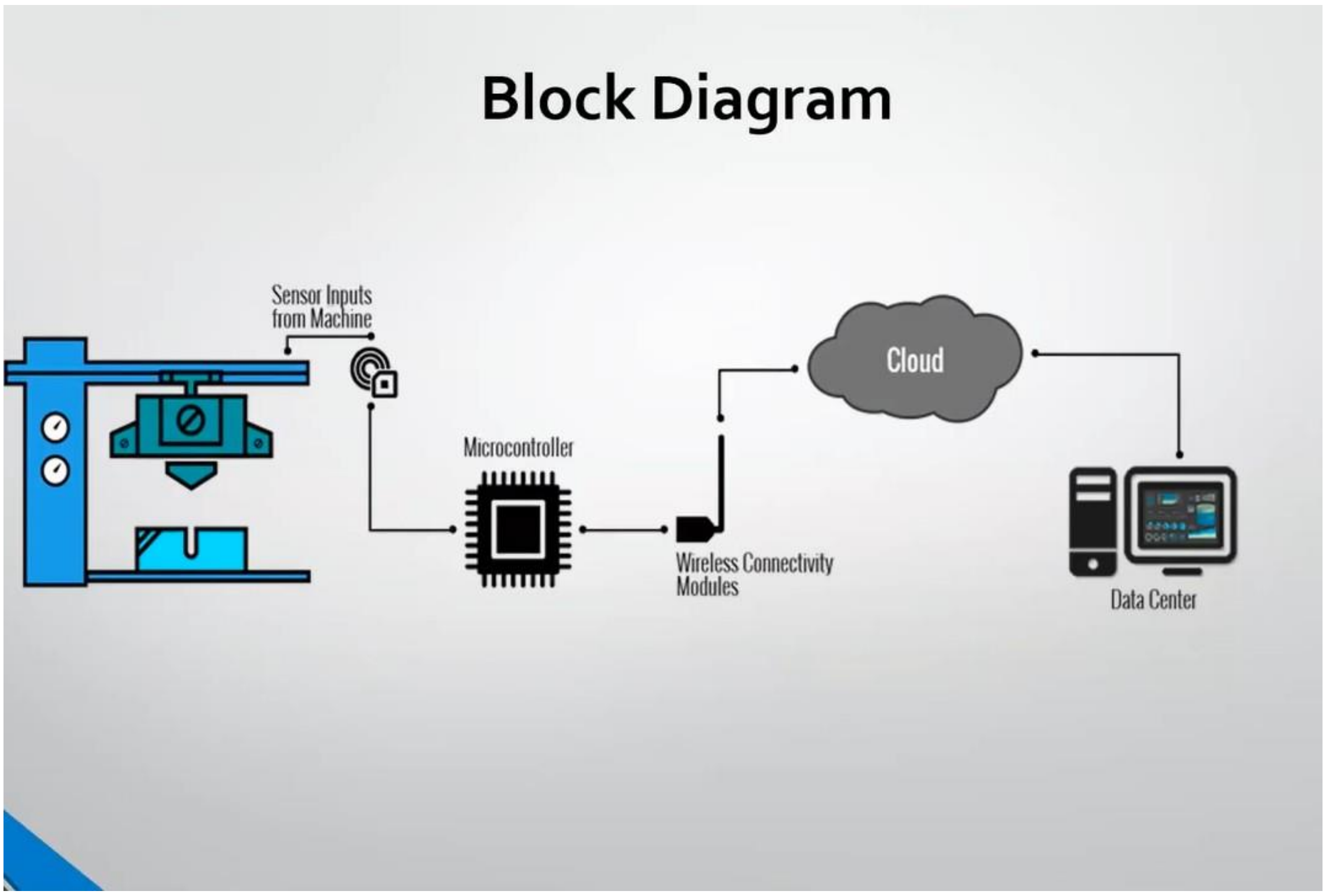
On

Internet of Things

INTRODUCTION

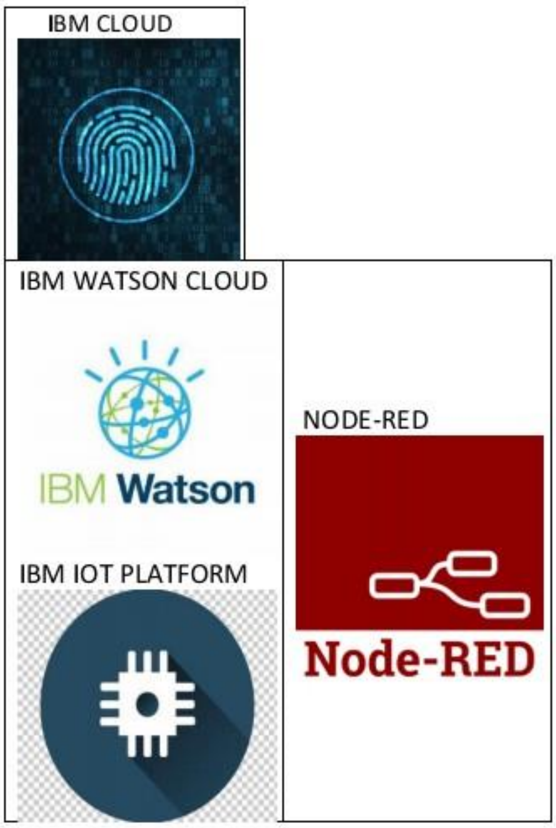
Industrial environmental conditions have been upgrading day by day with newly developing automation technology. And, as a result of getting rid of the conventional procedures of manufacturing , this leads to an increase in huge workloads. The next-gen industries will be more advanced and automated as company with existing ones. This brings a new terminology; “Smart Industries”. In this new era, Monitoring as well as controlling of various Industrial applications is challenging as ever. The Internet of Things (IoT), as an emerging technology that brought about rapid advancements in modern technologies, has attracted a lot of attention and is expected bring benefits to numerous applications. The newly introduced concepts is providing a helping hand to achieve Industrial automation through remote access.

The aim of the project is to monitor any manufacturing plant remotely for temperature, humidity, vibration, current etc. The program will set the parameters and if the results are not within the parameters, it will send a warning so that immediate action can be taken accordingly.



THEORITICAL ANALYSIS :

The block diagram for the IOT based Predictive Maintenance of Industrial Motors is shown below.



•

IBM CLOUD : It is a cloud platform where we have features og IBM IOT platform.

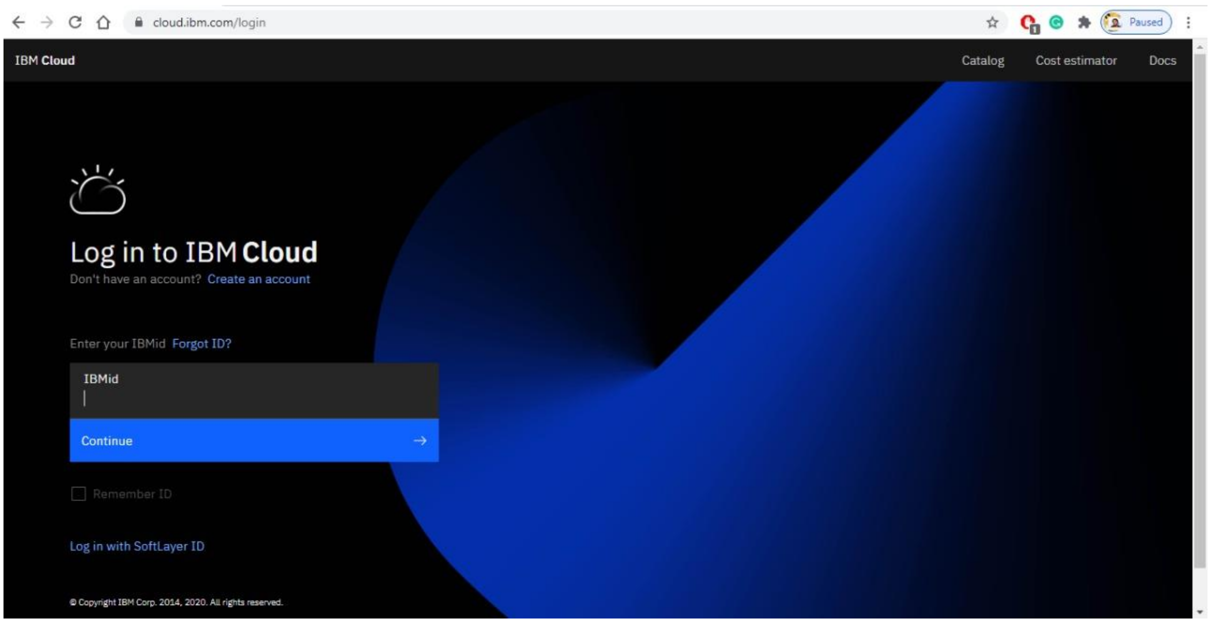
NODE-RED : It uses HTTP/MQTT protocol to connect devices to IBM IOT platform.

FAST2SMS : We use FAST 2SMS for sending message to mobile when the sensor value crosses the threshold.

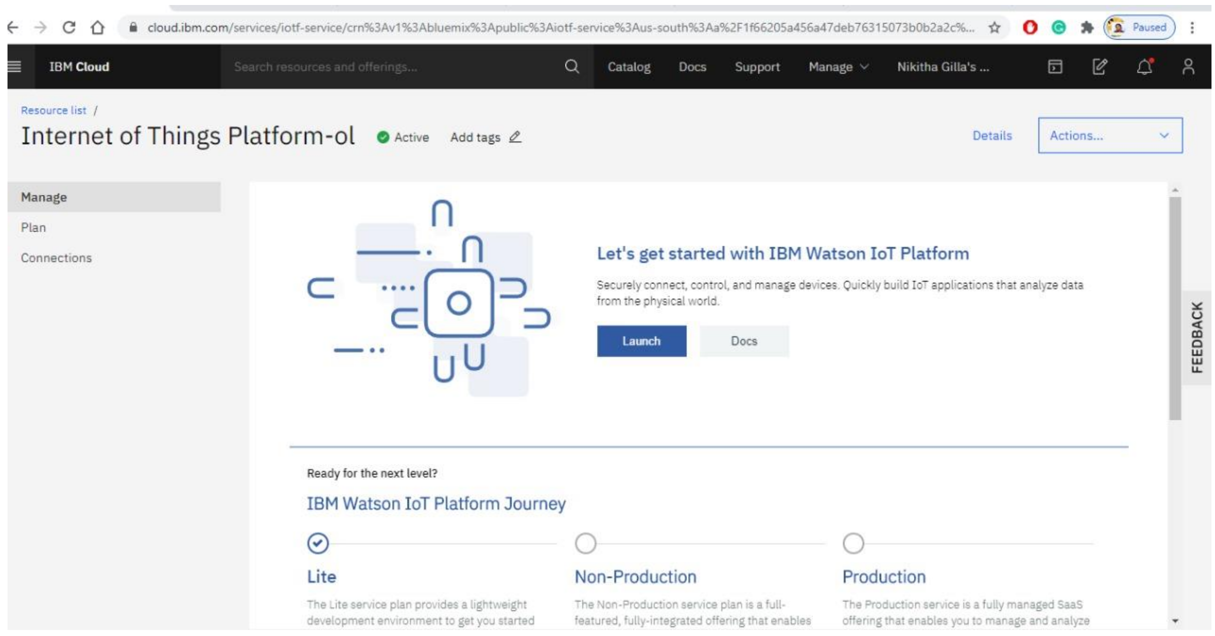
MIT APP INVENTOR : We use mit app inventor for displaying the sensor values in the mobile application.

Designing Procedure :

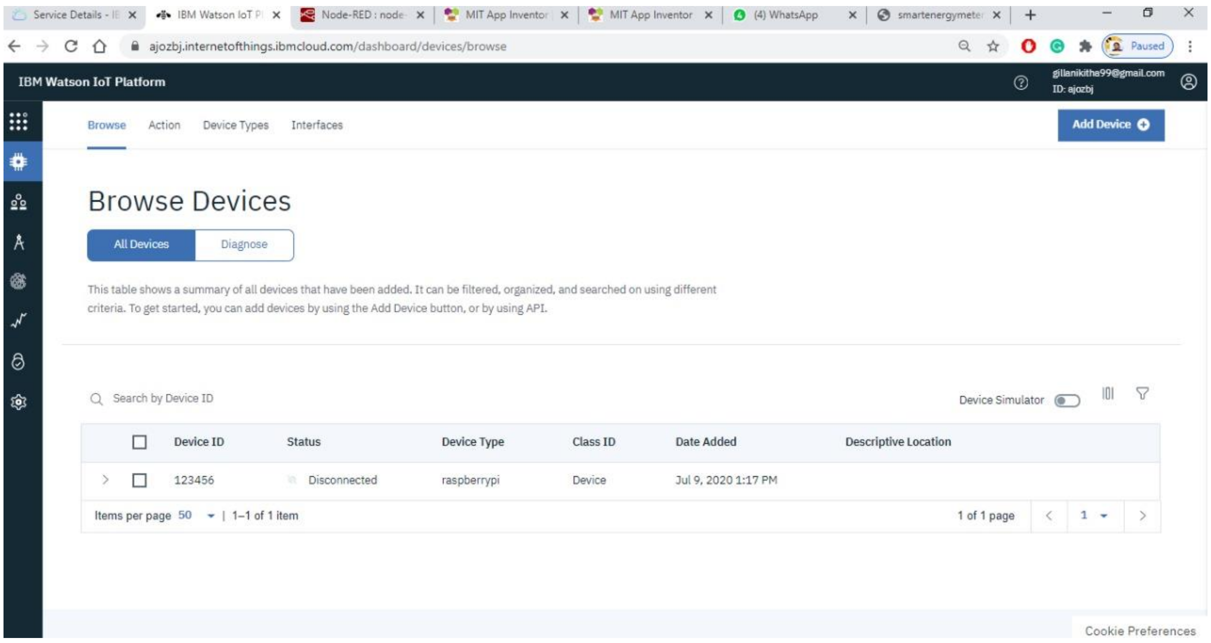
- Sign – in to your IBM account for the link https://cloud.ibm.com/login. Or else create your ibm account.


cloud.ibm.com/login
IBM Cloud
Catalog
Cost estimator
Docs
Log in to IBM Cloud
Don't have an account? Create an account
Enter your IBMid Forgot ID?
IBMid
Continue
Remember ID
Log in with SoftLayer ID
© Copyright IBM Corp. 2014, 2020. All rights reserved.

- Go to catalog and search for the IOT in search bar. Then select Internet of Things platform and subscribe for the desired plan and click create. Now in the menu, go to Resourcelist-→services---->Internet of Things Platform and then click Launch , as shown below.


IBM Cloud
Search resources and offerings...
Catalog
Docs
Support
Manage
Nikitha Gilla's ...
Resource list /
Internet of Things Platform-ol
Active
Add tags
Details
Actions...
Manage
Plan
Connections
Let's get started with IBM Watson IoT Platform
Securely connect, control, and manage devices. Quickly build IoT applications that analyze data from the physical world.
Launch
Docs
Ready for the next level?
IBM Watson IoT Platform Journey
Lite
Non-Production
Production
FEEDBACK

- Now in the Watson IOT platform, click on the Add Device button at the top right corner and register device in IBM cloud platform.



Browse Devices

This table shows a summary of all devices that have been added. It can be filtered, organized, and searched on using different criteria. To get started, you can add devices by using the Add Device button, or by using API.

Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
123456	Disconnected	raspberrypi	Device	Jul 9, 2020 1:17 PM	

Items per page 50 | 1–1 of 1 item   1 of 1 page   1

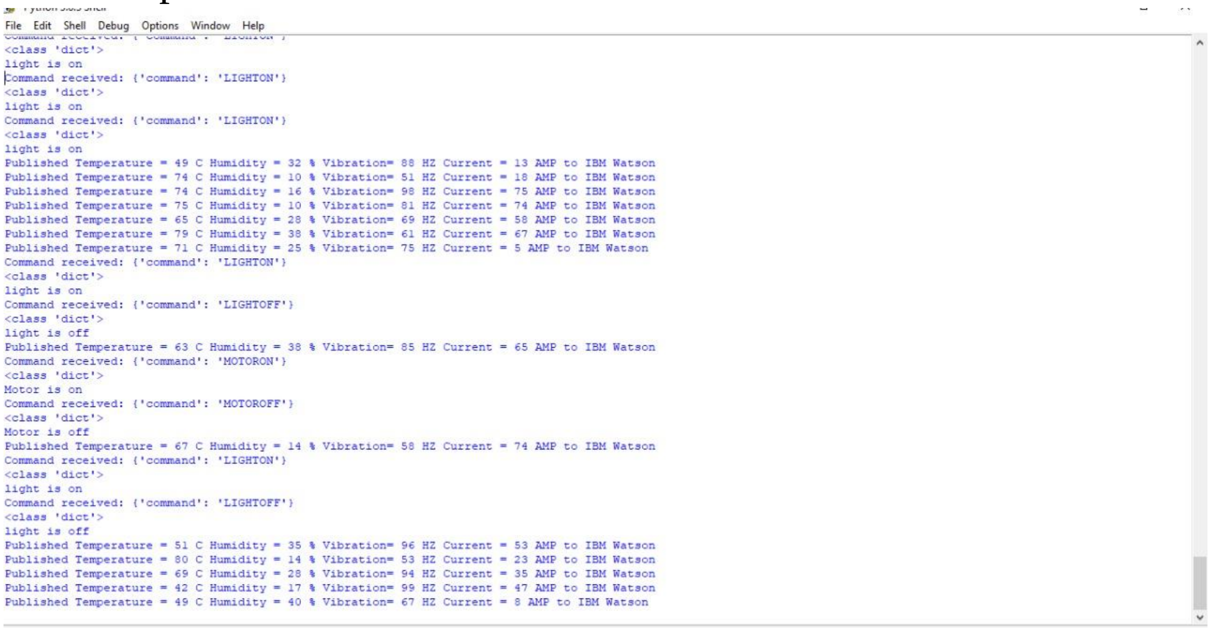
- Now modify the credentials in the ibm iot program and save and run the program.

PROGRAM :

```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random
#Provide your IBM Watson Device Credentials
organization = "ajozbj"
deviceType = "raspberrypi"
deviceId = "123456"
authMethod = "token"
authToken = "12345678"

# Initialize GPIO

def myCommandCallback(cmd):
        print("Command received: %s" % cmd.data)
        print(type(cmd.data))
        i=cmd.data['command']
        if i=='MOTORON':
                print("Motor is on")
        elif i=='MOTOROFF':
                print("Motor is off")
        elif i=='LIGHTON':
                print("light is on")
        elif i=='LIGHTOFF':
                print("light is off")


try:
        deviceOptions = {"org": organization, "type": deviceType,
"id": deviceId, "auth-method": authMethod, "auth-token":
authToken}
        deviceCli =
ibmiotf.device.Client(deviceOptions)#.........................................
...

except Exception as e:
         print("Caught exception connecting device: %s" % str(e))
         sys.exit()

# Connect and send a datapoint "hello" with value "world" into
the cloud as an event of type "greeting" 10 times
deviceCli.connect()

while True:

        hum = random.randint(10,40)
        #print(hum)
        temp = random.randint(30,80)
        vib = random.randint(50,100)
        curr = random.randint(5,80)
        #Send Temperature & Humidity to IBM Watson
        data = { 'Temperature' : temp, 'Humidity': hum,
'Vibration':vib, 'Current':curr }
        #print (data)
        def myOnPublishCallback():
            print ("Published Temperature = %s C" % temp,
"Humidity = %s %%" % hum, "Vibration= %s HZ" % vib
,"Current = %s AMP" % curr, "to IBM Watson")

        success = deviceCli.publishEvent("DHT11", "json", data,
qos=0, on_publish=myOnPublishCallback)
        if not success:
            print("Not connected to IoTF")
        time.sleep(2)

        deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud
deviceCli.disconnect()
```

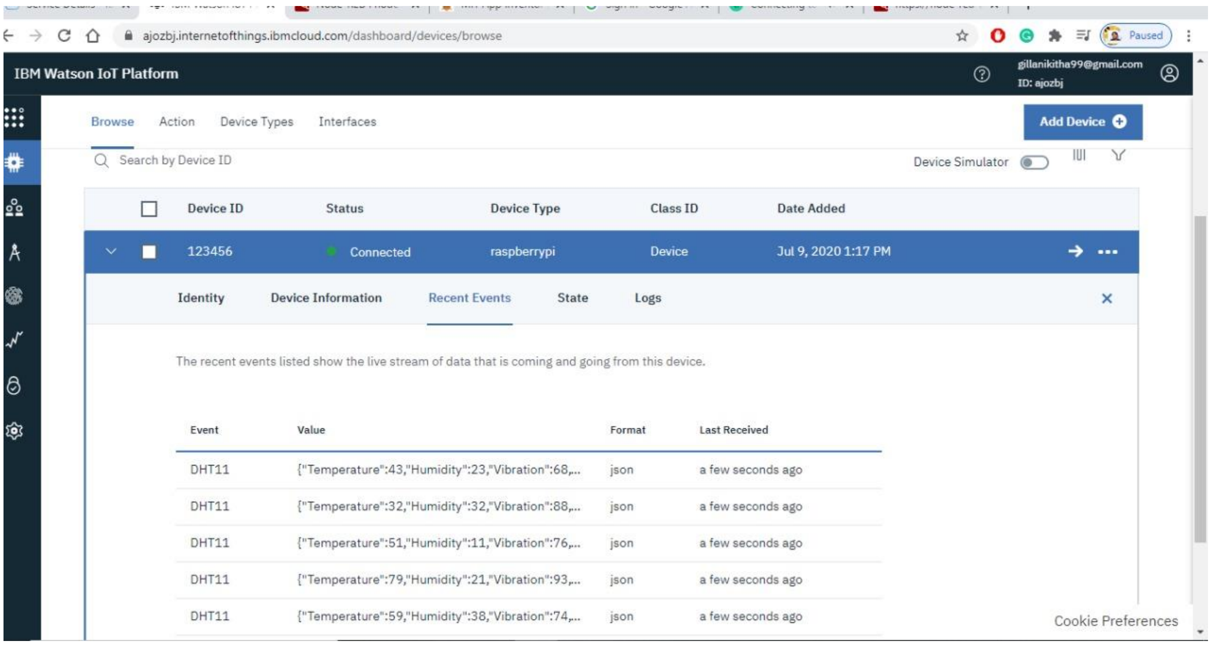
OUTPUT :

Code output :



```
<class 'dict'>
light is on
Command received: {'command': 'LIGHTON'}
<class 'dict'>
light is on
Command received: {'command': 'LIGHTON'}
<class 'dict'>
light is on
Published Temperature = 49 C Humidity = 32 % Vibration= 88 HZ Current = 13 AMP to IBM Watson
Published Temperature = 74 C Humidity = 10 % Vibration= 51 HZ Current = 18 AMP to IBM Watson
Published Temperature = 74 C Humidity = 16 % Vibration= 98 HZ Current = 75 AMP to IBM Watson
Published Temperature = 75 C Humidity = 10 % Vibration= 81 HZ Current = 74 AMP to IBM Watson
Published Temperature = 65 C Humidity = 28 % Vibration= 69 HZ Current = 58 AMP to IBM Watson
Published Temperature = 79 C Humidity = 38 % Vibration= 61 HZ Current = 67 AMP to IBM Watson
Published Temperature = 71 C Humidity = 25 % Vibration= 75 HZ Current = 5 AMP to IBM Watson
Command received: {'command': 'LIGHTON'}
<class 'dict'>
light is on
Command received: {'command': 'LIGHTOFF'}
<class 'dict'>
light is off
Published Temperature = 63 C Humidity = 38 % Vibration= 85 HZ Current = 65 AMP to IBM Watson
Command received: {'command': 'MOTORON'}
<class 'dict'>
Motor is on
Command received: {'command': 'MOTOROFF'}
<class 'dict'>
Motor is off
Published Temperature = 67 C Humidity = 14 % Vibration= 58 HZ Current = 74 AMP to IBM Watson
Command received: {'command': 'LIGHTON'}
<class 'dict'>
light is on
Command received: {'command': 'LIGHTOFF'}
<class 'dict'>
light is off
Published Temperature = 51 C Humidity = 35 % Vibration= 96 HZ Current = 53 AMP to IBM Watson
Published Temperature = 80 C Humidity = 14 % Vibration= 53 HZ Current = 23 AMP to IBM Watson
Published Temperature = 69 C Humidity = 28 % Vibration= 94 HZ Current = 35 AMP to IBM Watson
Published Temperature = 42 C Humidity = 17 % Vibration= 99 HZ Current = 47 AMP to IBM Watson
Published Temperature = 49 C Humidity = 40 % Vibration= 67 HZ Current = 8 AMP to IBM Watson
```

Output in IBM :



Device ID	Status	Device Type	Class ID	Date Added
123456	Connected	raspberrypi	Device	Jul 9, 2020 1:17 PM

The recent events listed show the live stream of data that is coming and going from this device.

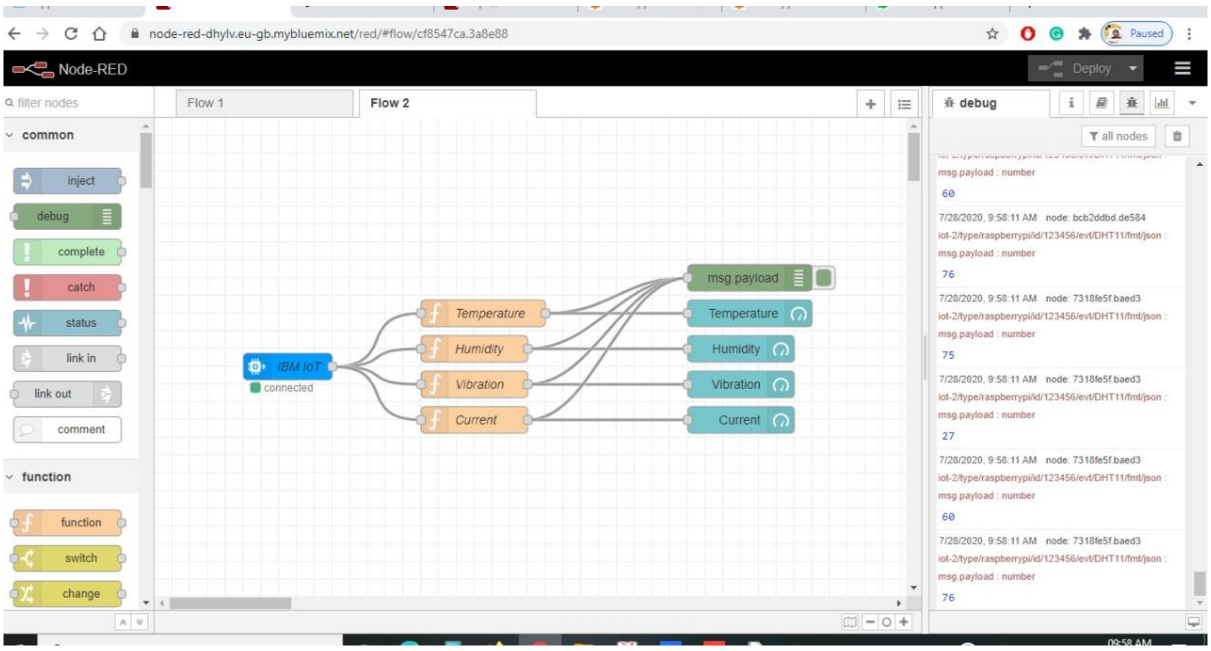
Event	Value	Format	Last Received
DHT11	{"Temperature":43,"Humidity":23,"Vibration":68,...	json	a few seconds ago
DHT11	{"Temperature":32,"Humidity":32,"Vibration":88,...	json	a few seconds ago
DHT11	{"Temperature":51,"Humidity":11,"Vibration":76,...	json	a few seconds ago
DHT11	{"Temperature":79,"Humidity":21,"Vibration":93,...	json	a few seconds ago
DHT11	{"Temperature":59,"Humidity":38,"Vibration":74,...	json	a few seconds ago

- Now we use special tool called Node-red , a low code programming tool for event- drive applications, to build a web-app.
- To install Node-red on windows, go to http://nodered.org/docs/getting-started/windows.

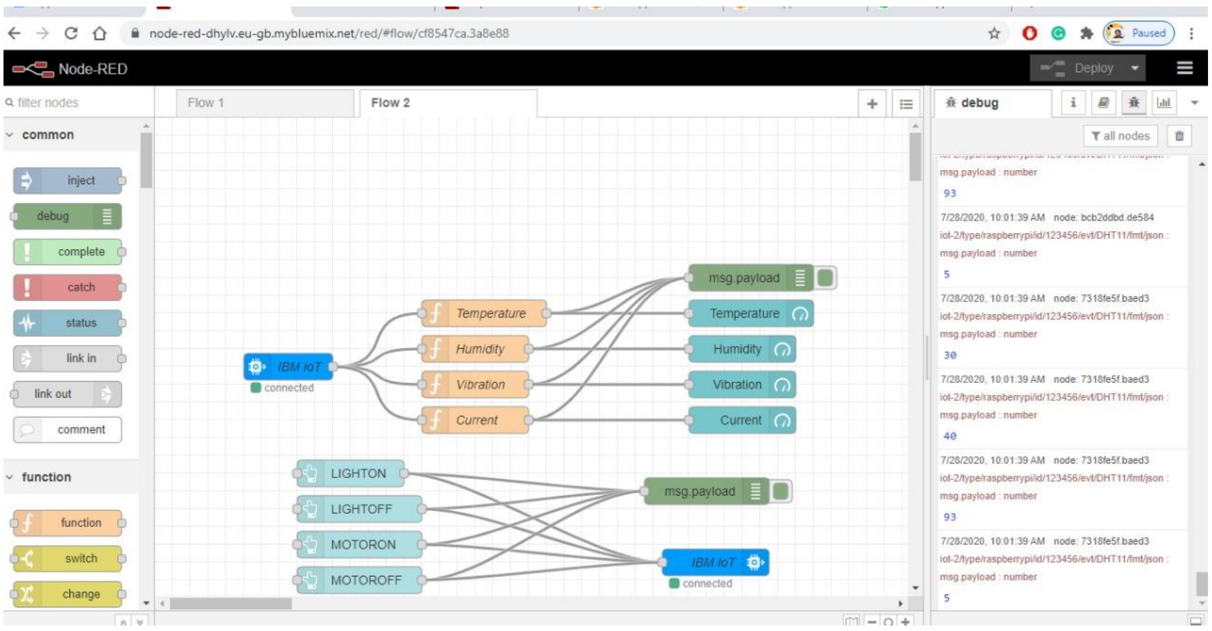
(For further details on how to use Node-red, visit https://nodered.org/docs/user-guide/)

Now to build a web app for Predictive Maintenance of Industrial Motors using node-red, a some flows would be required:

Flow1: Create a node red flow to send and retrieve data from the device.

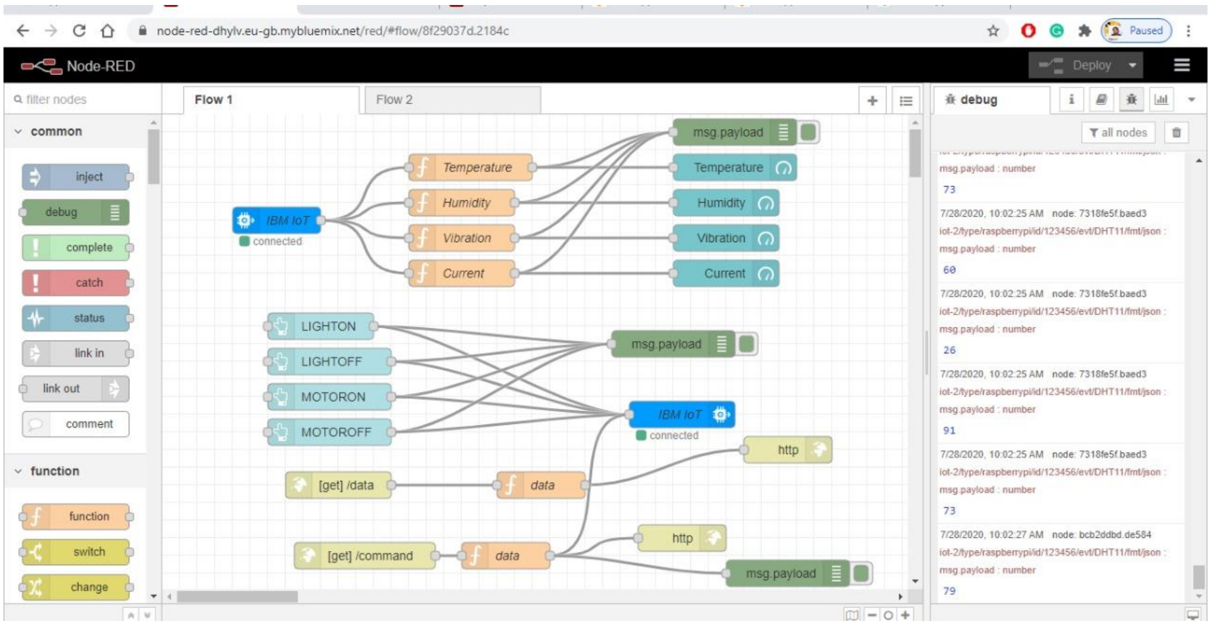


```
msg.payload : number
60
7/28/2020, 9:58:11 AM   node: bcb2ddbd.de584
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
76
7/28/2020, 9:58:11 AM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
75
7/28/2020, 9:58:11 AM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
27
7/28/2020, 9:58:11 AM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
60
7/28/2020, 9:58:11 AM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
76
```



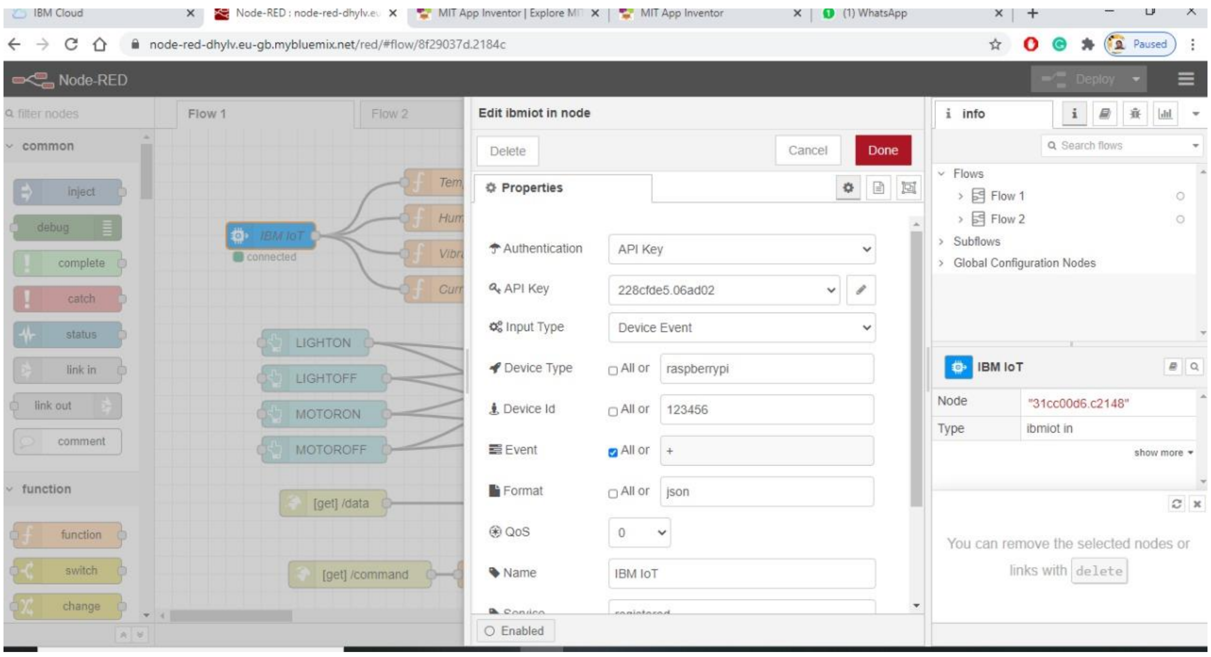
```
msg.payload : number
93
7/28/2020, 10:01:39 AM   node: bcb2ddbd.de584
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
5
7/28/2020, 10:01:39 AM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
30
7/28/2020, 10:01:39 AM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
40
7/28/2020, 10:01:39 AM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
93
7/28/2020, 10:01:39 AM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
5
```

Flow 2: Complete HTTP requests to communicate with the mobile.



```
msg.payload : number
73
7/28/2020, 10:02:25 AM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
60
7/28/2020, 10:02:25 AM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
26
7/28/2020, 10:02:25 AM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
91
7/28/2020, 10:02:25 AM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
73
7/28/2020, 10:02:27 AM   node: bcb2ddbd.de584
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
79
```

The configuration of the 'IBM IOT in' and the function nodes in the above flows are given as:



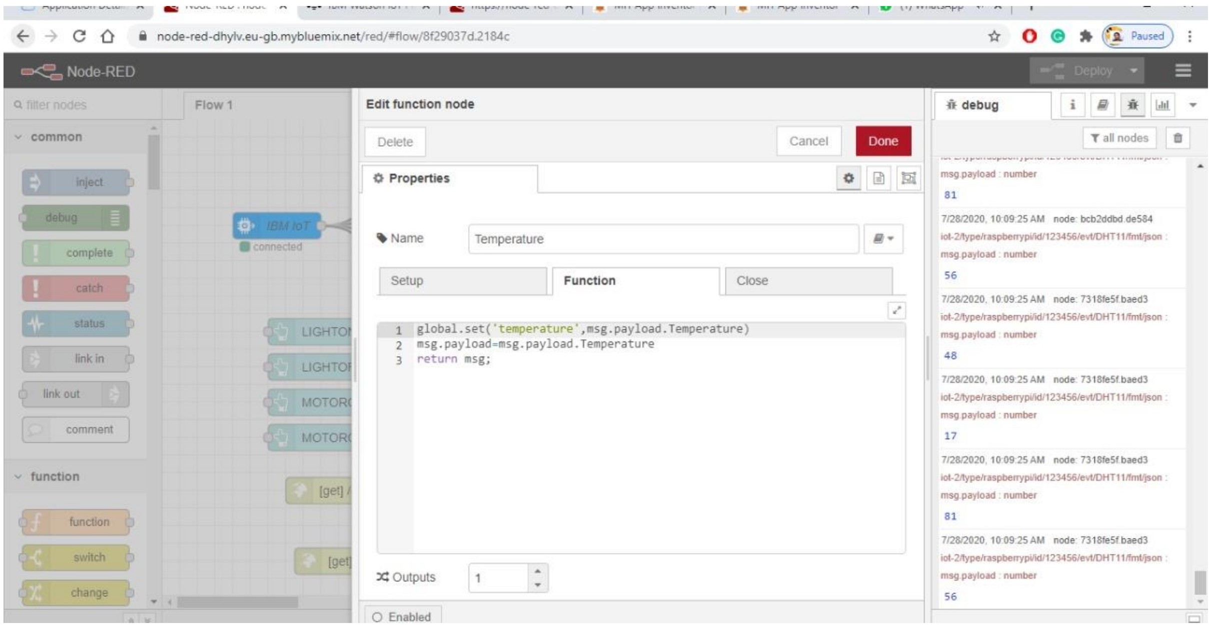
Edit ibmiot in node

Authentication		API Key
API Key		228cfde5.06ad02
Input Type		Device Event
Device Type	☐ All or	raspberrypi
Device Id	☐ All or	123456
Event	☑ All or	+
Format	☐ All or	json
QoS		0
Name		IBM IoT

IBM IoT

Node	"31cc00d6.c2148"
Type	ibmiot in

You can remove the selected nodes or links with delete


node-red-dhylv.eu-gb.mybluemix.net/red/#flow/8f29037d.2184c

Edit function node

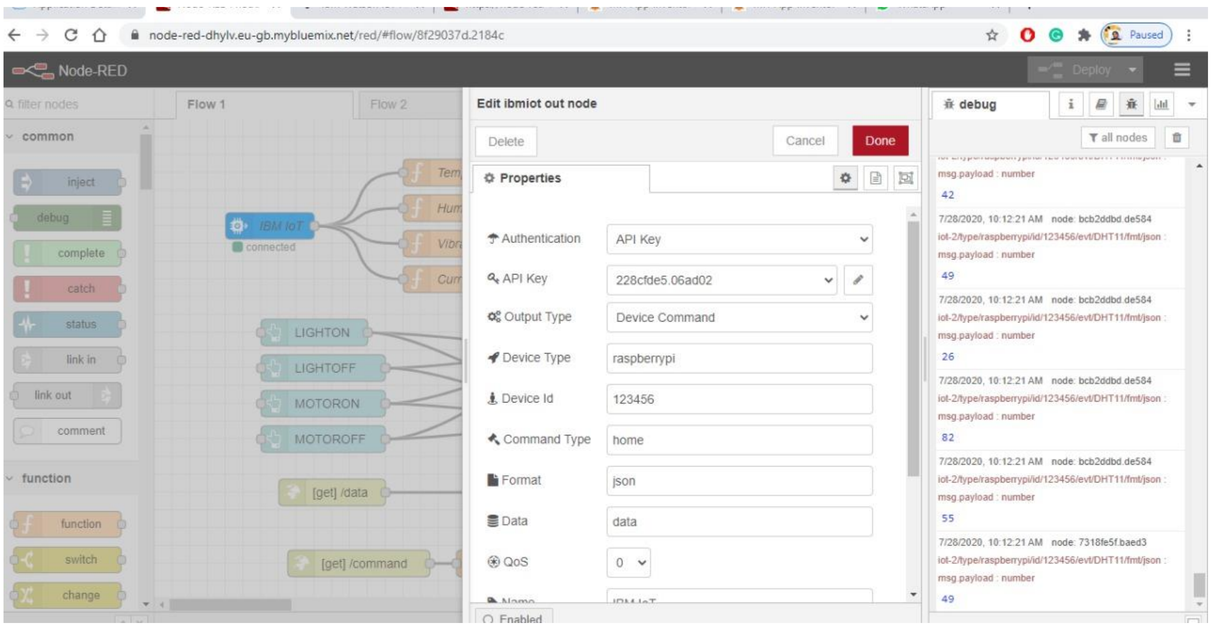
Name: Temperature

```
global.set('temperature',msg.payload.Temperature)
msg.payload=msg.payload.Temperature
return msg;
```

Outputs: 1

```
msg.payload : number
81
7/28/2020, 10:09:25 AM   node: bcb2ddbd.de584
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
56
7/28/2020, 10:09:25 AM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
48
7/28/2020, 10:09:25 AM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
17
7/28/2020, 10:09:25 AM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
81
7/28/2020, 10:09:25 AM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
56
```

The configuration of 'IBM iot out' node in the above figure of node-red is:

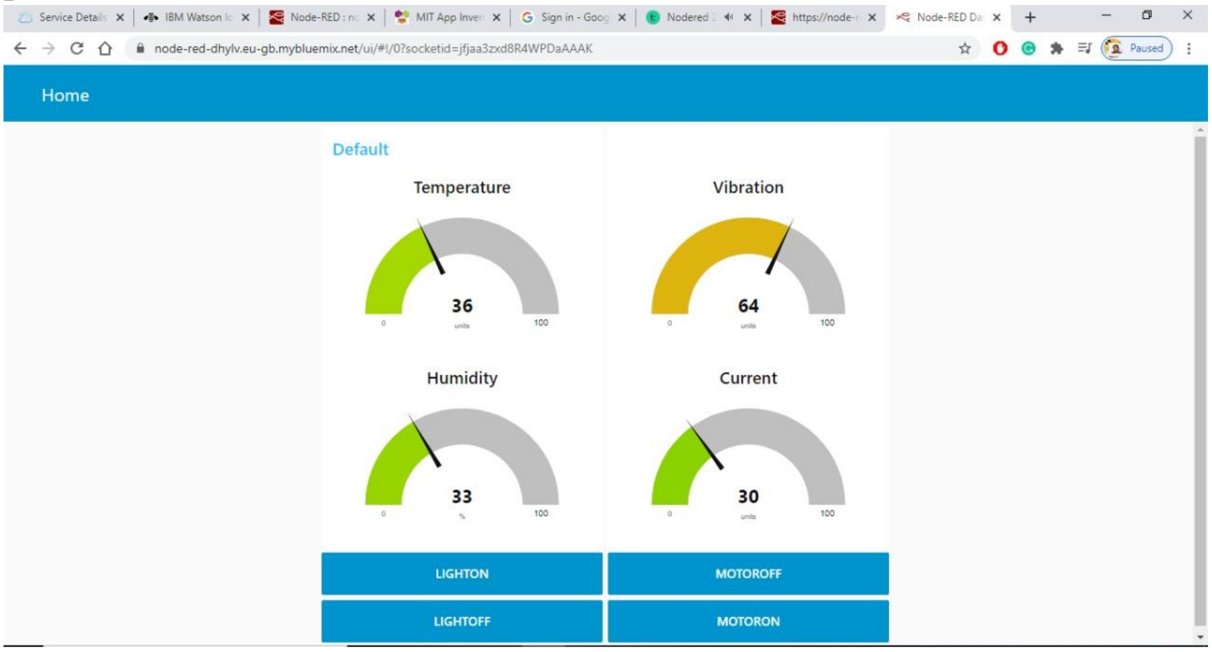

node-red-dhylv.eu-gb.mybluemix.net/red/#flow/8f29037d.2184c

Edit ibmiot out node

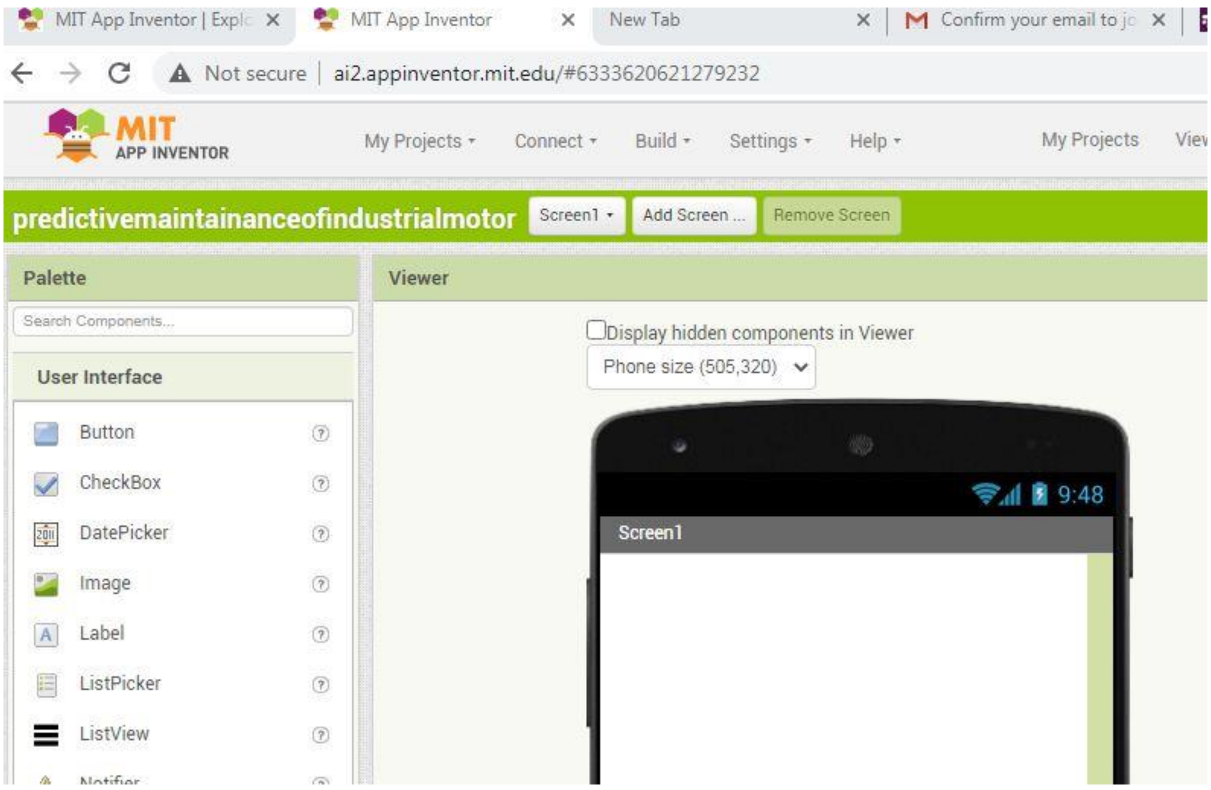
Authentication	API Key
API Key	228cfde5.06ad02
Output Type	Device Command
Device Type	raspberrypi
Device Id	123456
Command Type	home
Format	json
Data	data
QoS	0
Name	IBM IoT

```
msg.payload : number
42
7/28/2020, 10:12:21 AM   node: bcb2ddbd.de584
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
49
7/28/2020, 10:12:21 AM   node: bcb2ddbd.de584
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
26
7/28/2020, 10:12:21 AM   node: bcb2ddbd.de584
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
82
7/28/2020, 10:12:21 AM   node: bcb2ddbd.de584
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
55
7/28/2020, 10:12:21 AM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
49
```

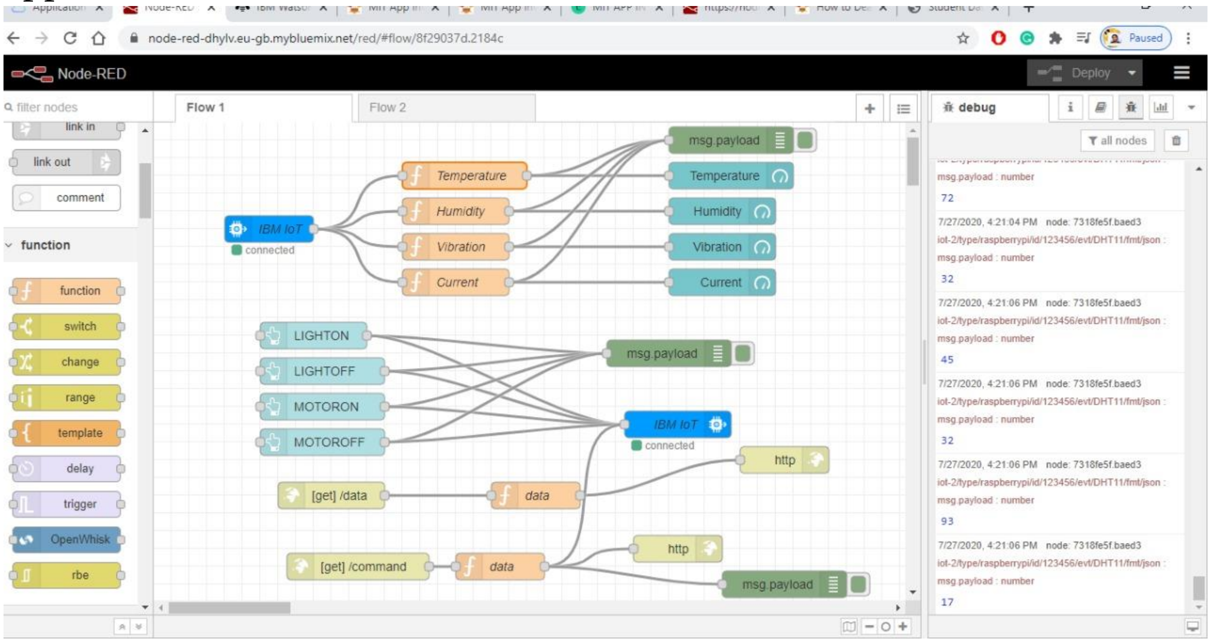
RESULT : The web application generated by the above designing procedure is as follows:



- Now to display the sensor values in the mobile application search for mit app inventor and click on MIT APP INVENTER and click to create app and login.

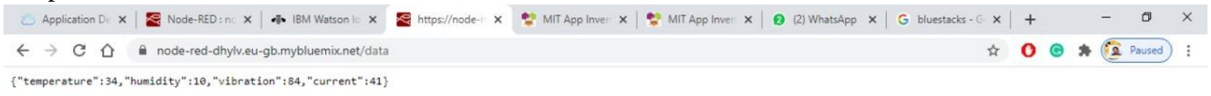


- Now we get data into node-red and to send this data into mobile app we need some interface HTTP.



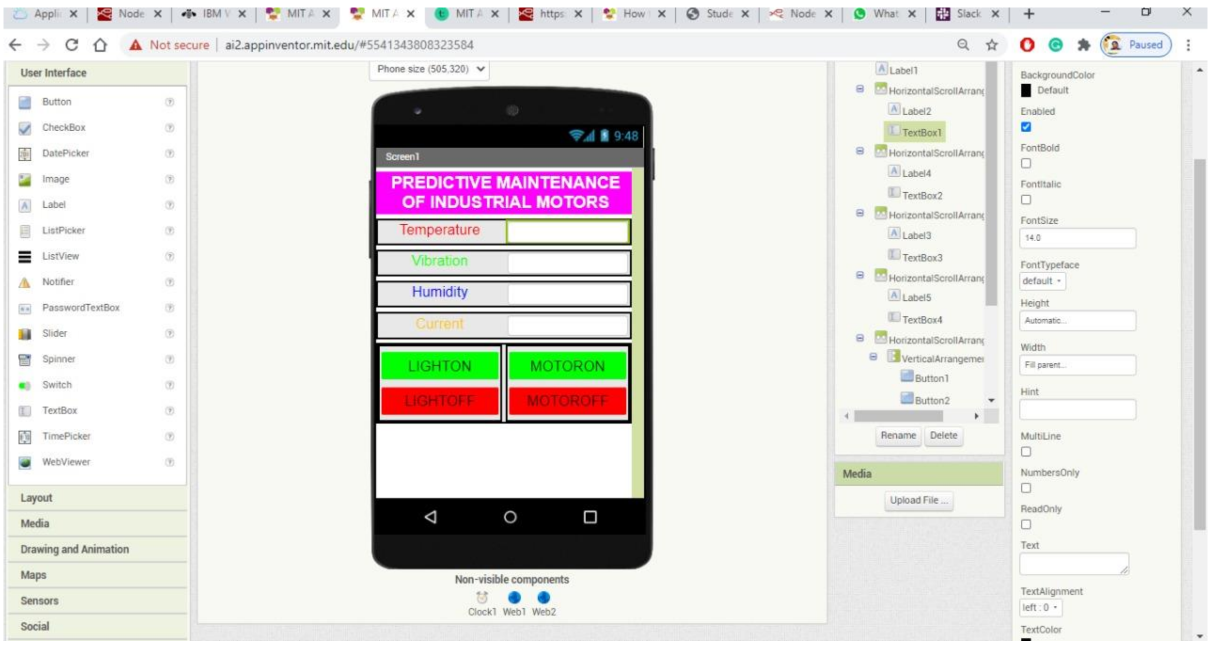
```
msg.payload : number
72
7/27/2020, 4:21:04 PM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
32
7/27/2020, 4:21:06 PM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
45
7/27/2020, 4:21:06 PM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
32
7/27/2020, 4:21:06 PM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
93
7/27/2020, 4:21:06 PM   node: 7318fe5f.baed3
iot-2/type/raspberrypi/id/123456/evt/DHT11/fmt/json :
msg.payload : number
17
```

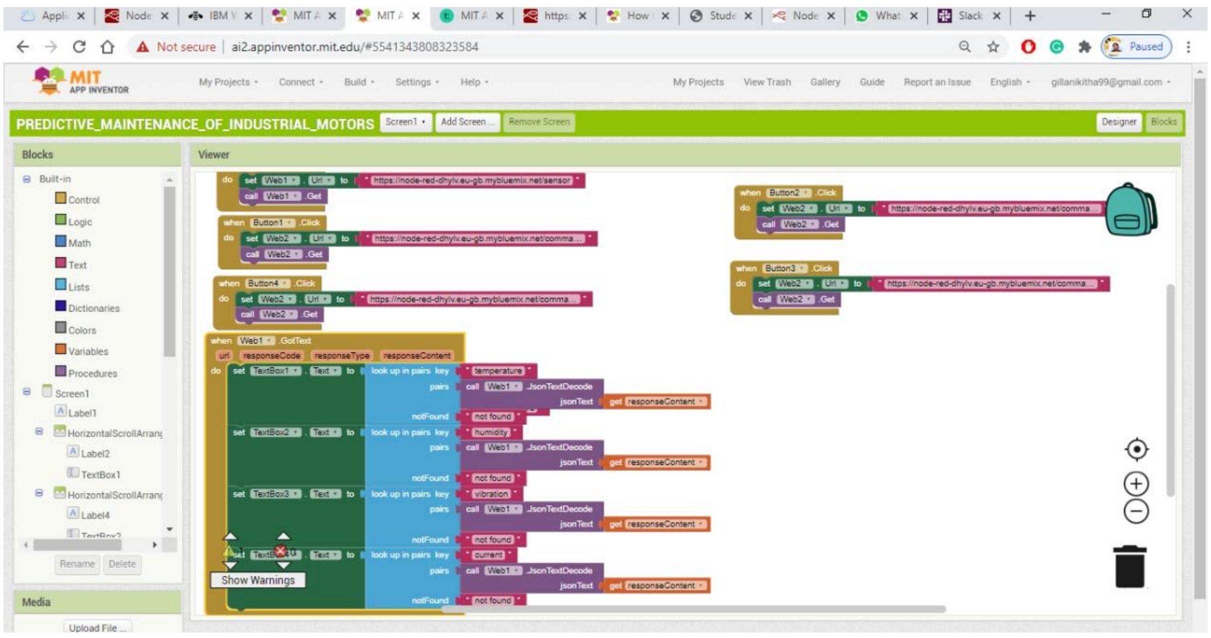
Output :



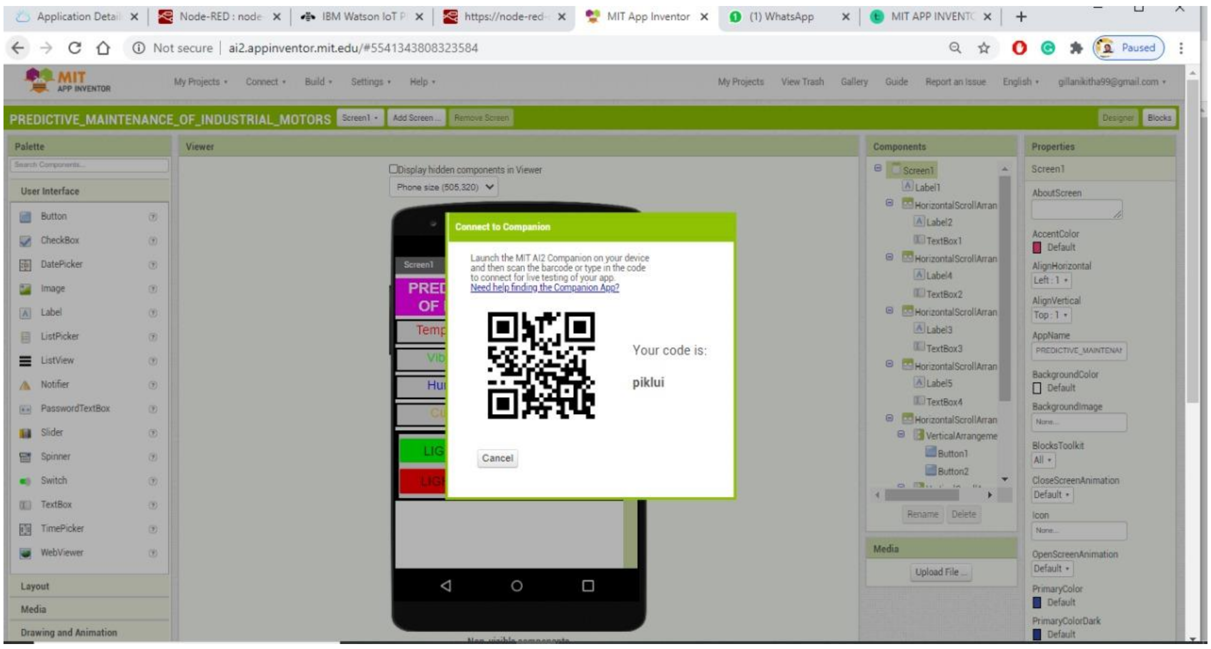
```
{"temperature":34,"humidity":10,"vibration":84,"current":41}
```

- Controlling the appliances using mobile app:





- Now execute the task in mobile app MIT AI2 COMPANION , Connect to AI COMPANION . Scan the QR code or connect with code (6 character code)

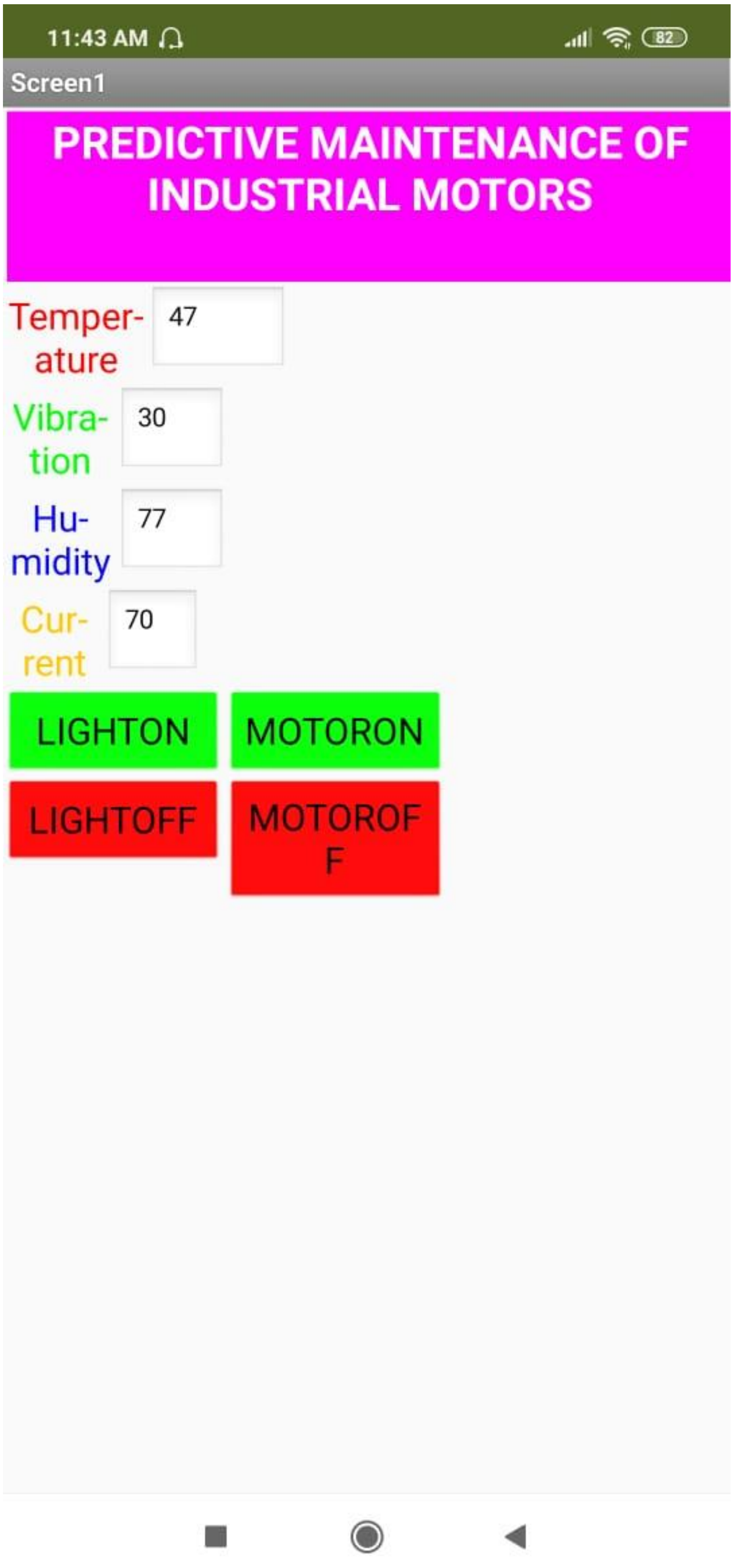

ai2.appinventor.mit.edu/#5541343808323584
PREDICTIVE_MAINTENANCE_OF_INDUSTRIAL_MOTORS
Palette
Viewer
Components
Properties
Screen1
Connect to Companion
Cancel

Launch the MIT AI2 Companion on your device and then scan the barcode or type in the code to connect for live testing of your app.
Need help finding the Companion App?

Your code is:

piklui

- After scanning you will get the output.



PREDICTIVE MAINTENANCE OF INDUSTRIAL MOTORS

Temper-ature	47
Vibra-tion	30
Hu-midity	77
Cur-rent	70

ADVANTAGES

- Reduction in maintenance costs.
- Reduction in machine failures.
- Reduced downtime for repairs.
- Reduced stock of spare parts.
- Increased service life of parts.
- Increased production
- Improved operator safety.
- Verification of repairs.

DISADVANTAGES

- Increases investment in diagnostic equipment.
- Increases investment in staff training.
- Savings potential is readily seen by management.

CONCLUSION

This paper focused on the problem of carrying out predictive maintenance in a industrial motors and presented the results of the preliminary data analysis and feature selection that were performed on a sample of the collected data .The derived data from IoT device gives the status of industrial motors about temperature, humidity, Vibration, Current to and from the equipment is continuously monitored to avoid any short circuits the line breakage. So Predictive Maintenance of Industrial Motors plays a major roll in maintaining industrial production.

It is possible to have a successful preventive maintenance program. From a cost reduction viewpoint is essential , but it does entail risk . In order to minimize risk, preventive maintenance has to be carefully planned and carried out by well – trained and motivated workers.