

A Project Report
on
Smart Energy Meter

By
S.Prathyusha
(prathyusharsingieddy123@gmail.com)

as an intern at
smartinternz.com@rsip2020

On
Internet of Things

Introduction:

Overview:

In this method we are using Node-red services which designs websites. and it uses node js language .The configuration of ibm iot node which connects iot platform. And also we are using MIT APP inventor to display the sensor values in the mobile application . Energy meters which is already installed at our houses are not replaced, but a small modification on the already installed meters can change the existing meters into smart meters. The use of FAST 2SMS provides a feature of notification through SMS. One can easily access the meter working through connecting our mobile app by installing MIT APP ai2 companion. Current reading with cost can be seen on web page. Automatic ON & OFF of meter is possible. Threshold value setting and sending of notification is the additional task that we are performing.

Purpose:

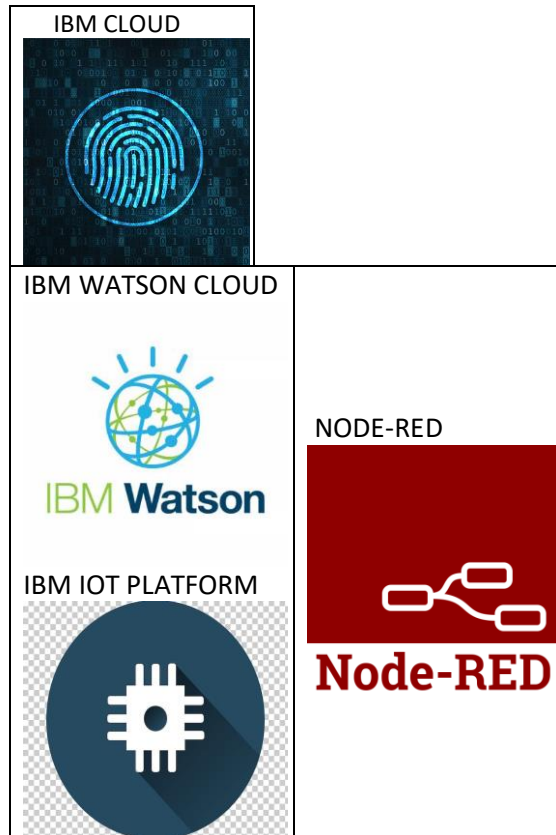
The present project “IoT Based Smart Energy Meter” addresses the problems faced by both the consumers and the distribution companies. It mainly deals with smart energy meter, which utilizes the features of embedded systems i.e. combination of hardware and software in order to implement desired functionality. It discusses the application of FAST 2sms and Wi-Fi modems to introduce ‘Smart’ concept. With the use of FAST 2sms modem the consumer as well as service provider will get the used energy reading with the respective amount, Consumers will even get notification in the form text through Fast2sms when they are about to reach their threshold value, that they have set. Also with the help of Wi-Fi modem the consumer can monitor hises consumed reading and can set the threshold value through webpages

LITERATURE SURVEY:

A new concept of energy meter will be discussed, where maximum demand of energy of a consumer will be indicated in the live url. After exceeding the maximum demand, the meter and hence the connection will automatically be disconnected . Fast 2sms used to produce communication between circuit and utility side. We actually have Fast 2sms to send notification to mobile. .We have NODE_RED services which designs websites. we can built websites by using nodes instead of php. It is a flow based application. Here we use API key to integrate iot platform to Node-red

THEORITICAL ANALYSIS:

The block diagram for the IOT based Smart Energy Meter is shown below



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

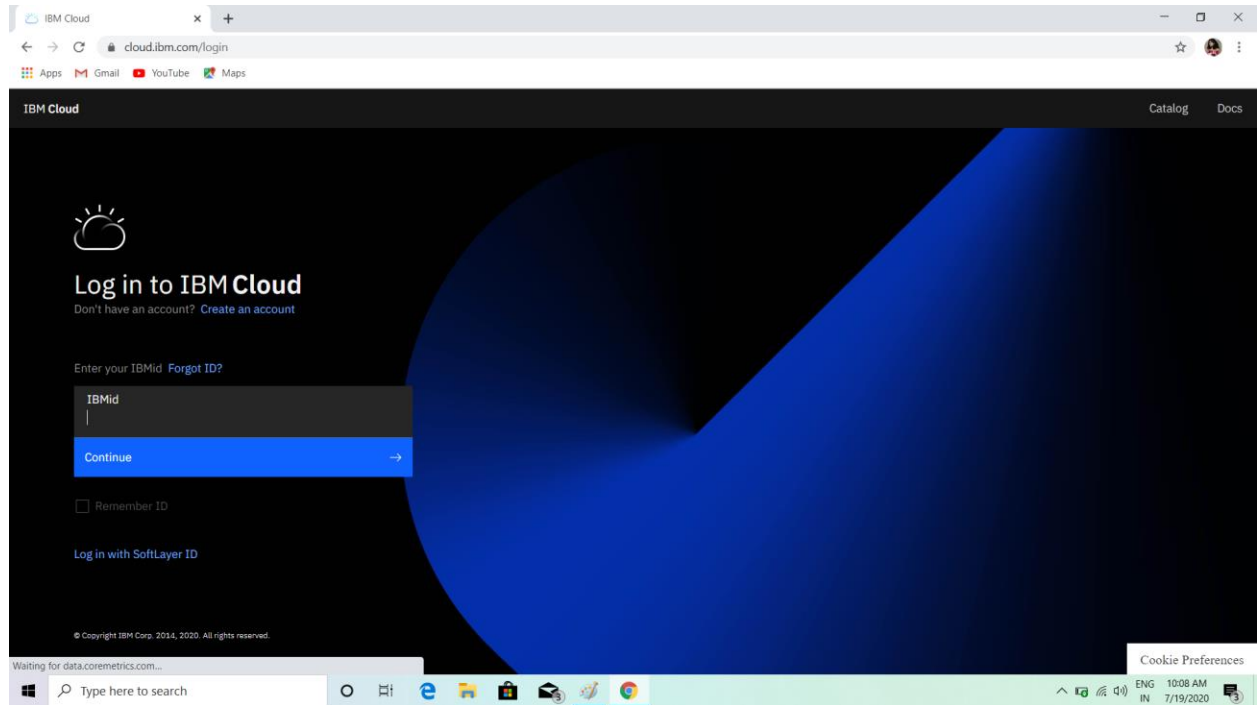
NODE-RED: It uses HTTP/MQTT protocols to connect devices to IBM IOT platform

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



- Go to catalog and search for 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:

Internet of Things Platform-sv 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**
The Lite service plan provides a lightweight development environment to get you started with the connectivity capabilities of Watson IoT Platform.
- Non-Production**
The Non-Production service plan is a full-featured, fully-integrated offering that enables you to explore Watson IoT Platform to see how the service can fit into your IoT environment.
- Production**
The Production service is a fully managed SaaS offering that enables you to manage and analyze enterprise IoT data.

• Free • Starts at \$500 per month • Includes IBM Service & Support

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

IBM Watson IoT Platform

[Browse](#) [Action](#) [Device Types](#) [Interfaces](#)

Browse Devices

[All Devices](#) [Diagnose](#)

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.

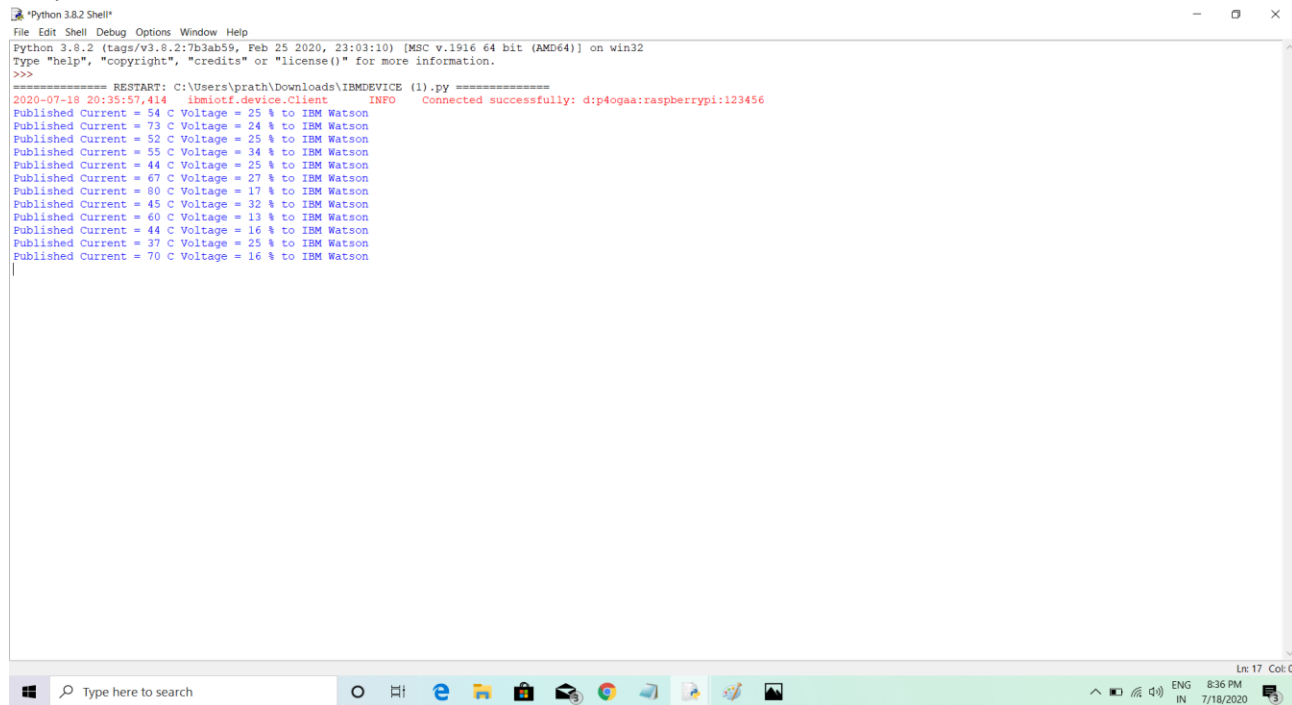
Search by Device ID Device Simulator 101 Filter

Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
123456	Disconnected	raspberrypi	Device	16 Jul 2020 10:12	

Items per page 50 | 1-1 of 1 item 1 of 1 page [1](#) [2](#)

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

Output:



```
Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 23:03:10) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\prath\Downloads\IBMDEVICE (1).py =====
2020-07-18 20:35:57,414 ibmiotf.device.Client INFO Connected successfully: d:p4ogaa:raspberrypi:123456
Published Current = 54 C Voltage = 25 % to IBM Watson
Published Current = 73 C Voltage = 24 % to IBM Watson
Published Current = 52 C Voltage = 25 % to IBM Watson
Published Current = 55 C Voltage = 34 % to IBM Watson
Published Current = 44 C Voltage = 25 % to IBM Watson
Published Current = 67 C Voltage = 27 % to IBM Watson
Published Current = 80 C Voltage = 17 % to IBM Watson
Published Current = 45 C Voltage = 32 % to IBM Watson
Published Current = 60 C Voltage = 13 % to IBM Watson
Published Current = 44 C Voltage = 16 % to IBM Watson
Published Current = 37 C Voltage = 25 % to IBM Watson
Published Current = 70 C Voltage = 16 % to IBM Watson
```

- Now we use a 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 Smart Energy Meter using node-red, a some flows would be required:

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

Application Details x Node-RED : node- x Node-RED on IBM x IBM Cloud x IBM Watson IoT Pl x IoT Sensor x Service Details - IE x IBM Watson IoT Pl x

node-red-mythj.eu-gb.mybluemix.net/red/#flow/b66a1657.c88ab8

Apps Gmail YouTube Maps

Node-RED

Flow 1

inject debug complete catch status link in link out comment

function

- function
- switch
- change
- range
- template
- delay

Prathyusha Singireddy

msg.payload

IBM IoT

Current

Voltage

debug

all nodes

34

7/18/2020, 5:59:08 PM node: e55784aa1f67068
iot-2/type/rasberryplid/123456/evl/smart energy
meter/rmtl/json : msg.payload : number

77

7/18/2020, 5:59:08 PM node: e55784aa1f67068
iot-2/type/rasberryplid/123456/evl/smart energy
meter/rmtl/json : msg.payload : number

10

7/18/2020, 5:59:08 PM node: e55784aa1f67068
iot-2/type/rasberryplid/123456/evl/smart energy
meter/rmtl/json : msg.payload : number

56

7/18/2020, 5:59:08 PM node: e55784aa1f67068
iot-2/type/rasberryplid/123456/evl/smart energy
meter/rmtl/json : msg.payload : number

40

7/18/2020, 5:59:10 PM node: e55784aa1f67068
iot-2/type/rasberryplid/123456/evl/smart energy
meter/rmtl/json : msg.payload : number

73

7/18/2020, 5:59:10 PM node: e55784aa1f67068
iot-2/type/rasberryplid/123456/evl/smart energy
meter/rmtl/json : msg.payload : number

33

Type here to search

ENG 5:59 PM 7/18/2020

The screenshot displays the Node-RED web interface in a browser. The main workspace shows a flow named 'Flow 1' with the following components and connections:

- IBM IoT Node:** A blue node labeled 'connected' on the left.
- Function Nodes:** Two orange nodes labeled 'Current' and 'Voltage' in the center.
- Output Nodes:** Two teal nodes labeled 'Current' and 'Voltage' on the right.
- msg.payload Node:** A green node in the center-right.
- HTTP Node:** A yellow node labeled 'http' at the bottom.
- SWITCHON and SWITCHOFF Nodes:** Two teal nodes at the bottom left.
- IBM IoT Node (Bottom):** A blue node labeled 'connected' at the bottom center.

Connections include:

- From the top 'IBM IoT' node to both 'Current' and 'Voltage' function nodes.
- From both 'Current' and 'Voltage' function nodes to their respective 'Current' and 'Voltage' output nodes.
- From both 'Current' and 'Voltage' function nodes to the 'msg.payload' node.
- From the 'msg.payload' node to the 'http' node.
- From the 'SWITCHON' and 'SWITCHOFF' nodes to the bottom 'IBM IoT' node.
- From the bottom 'IBM IoT' node to the 'msg.payload' node.

The right sidebar shows the 'debug' console with the following log entries:

```
7/18/2020, 10:41:05 PM node: e55784aa.f67068  
iot-2/type/rasberrypi/id/123456/evt/smart energy meter/fmt/json : msg.payload : Object  
  { Current: 46, Voltage: 18 }  
7/18/2020, 10:41:06 PM node: e55784aa.f67068  
iot-2/type/rasberrypi/id/123456/evt/smart energy meter/fmt/json : msg.payload : number  
46  
7/18/2020, 10:41:07 PM node: e55784aa.f67068  
iot-2/type/rasberrypi/id/123456/evt/smart energy meter/fmt/json : msg.payload : number  
18  
7/18/2020, 10:41:08 PM node: e55784aa.f67068  
iot-2/type/rasberrypi/id/123456/evt/smart energy meter/fmt/json : msg.payload : Object  
  { Current: 67, Voltage: 24 }  
7/18/2020, 10:41:09 PM node: e55784aa.f67068  
iot-2/type/rasberrypi/id/123456/evt/smart energy meter/fmt/json : msg.payload : number  
67  
7/18/2020, 10:41:10 PM node:
```

Flow 2: Complete HTTP requests to communicate with the mobile

Sign in - Google x #AlaVaikunt x Search x www.amazon x LITERATURE x Application x Node-RED x NODE-RED x Service Det x IBM Watson x + - x

node-red-myhj.eu-gb.mybluemix.net/red/#flow/b66a1657.c88ab8

Apps Gmail YouTube Maps

Node-RED

Filter nodes

Flow 1

common

- inject
- debug
- complete
- catch
- status
- link in
- link out
- comment

function

- function
- switch
- change
- range
- template

IBM IoT in

Current

Voltage

msg.payload

switch

http request

[get] /sensor

data

http

SWITCHON

SWITCHOFF

IBM IoT out

debug

Debug messages (ctrl-g d)

```
7/21/2020, 10:42:49 AM node: 594b7aa5.820694
iot-2/type/rasberrypld/123456/evl/smart energy
meter/rmt/json : msg.payload : Object
{ Current: 45, Voltage: 16 }

7/21/2020, 10:42:49 AM node: 594b7aa5.820694
iot-2/type/rasberrypld/123456/evl/smart energy
meter/rmt/json : msg.payload : number
45

7/21/2020, 10:42:50 AM node: 594b7aa5.820694
iot-2/type/rasberrypld/123456/evl/smart energy
meter/rmt/json : msg.payload : number
16

7/21/2020, 10:42:52 AM node: 594b7aa5.820694
iot-2/type/rasberrypld/123456/evl/smart energy
meter/rmt/json : msg.payload : Object
{ Current: 40, Voltage: 11 }

7/21/2020, 10:42:53 AM node: 594b7aa5.820694
iot-2/type/rasberrypld/123456/evl/smart energy
meter/rmt/json : msg.payload : number
40

7/21/2020, 10:42:54 AM node: 594b7aa5.820694
iot-2/type/rasberrypld/123456/evl/smart energy
meter/rmt/json : msg.payload : number
11

7/21/2020, 10:42:55 AM node: 594b7aa5.820694
```

https://node-red-myhj.eu-gb.mybluemix.net/red/#debug

Type here to search

ENG 10:43 AM 7/21/2020

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

The screenshot shows the Node-RED web interface in a browser. The main workspace displays a flow with an 'IBM IoT' node connected to 'Current' and 'Voltage' function nodes. The 'Edit IBM IoT in node' dialog is open, showing the following configuration:

- Authentication: API Key
- API Key: c6480528.738438
- Input Type: Device Event
- Device Type: All or raspberry
- Device Id: All or 123456
- Event: All or +
- Format: All or json
- QoS: 0
- Name: IBM IoT
- Service: registered

A yellow banner at the bottom of the dialog reads: "Use the Input Type property to configure this node to receive Events". The right sidebar shows a debug console with messages from the IoT node, including payloads like `{ Current: 32, Voltage: 37 }`.

The screenshot shows the Node-RED web interface with the 'Edit function node' dialog open for the 'Current' node. The configuration is as follows:

- Name: Current
- Function:

```
1 global.set('current',msg.payload.Current)
2 msg.payload=msg.payload.Current
3 return msg;
```
- Outputs: 1

The right sidebar shows the debug console with messages from the function node, including payloads like `{ Current: 43, Voltage: 36 }`.

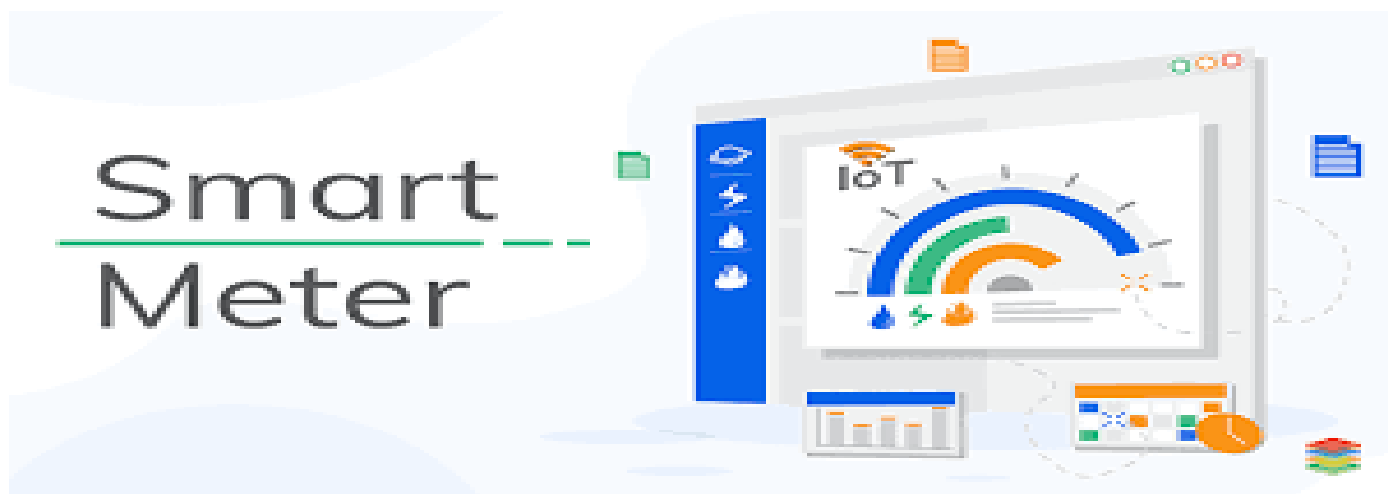
The configuration of 'IBM iot out' node in the above figure of nod-red is:

The screenshot displays the Node-RED web interface in a browser. The main workspace shows a flowchart with a 'data' node connected to a 'switch' node, which then branches into 'SWITCHON' and 'SWITCHOFF' nodes. The 'Edit ibmiot out node' panel is open on the right, showing configuration for an IBM IoT device. The configuration includes:

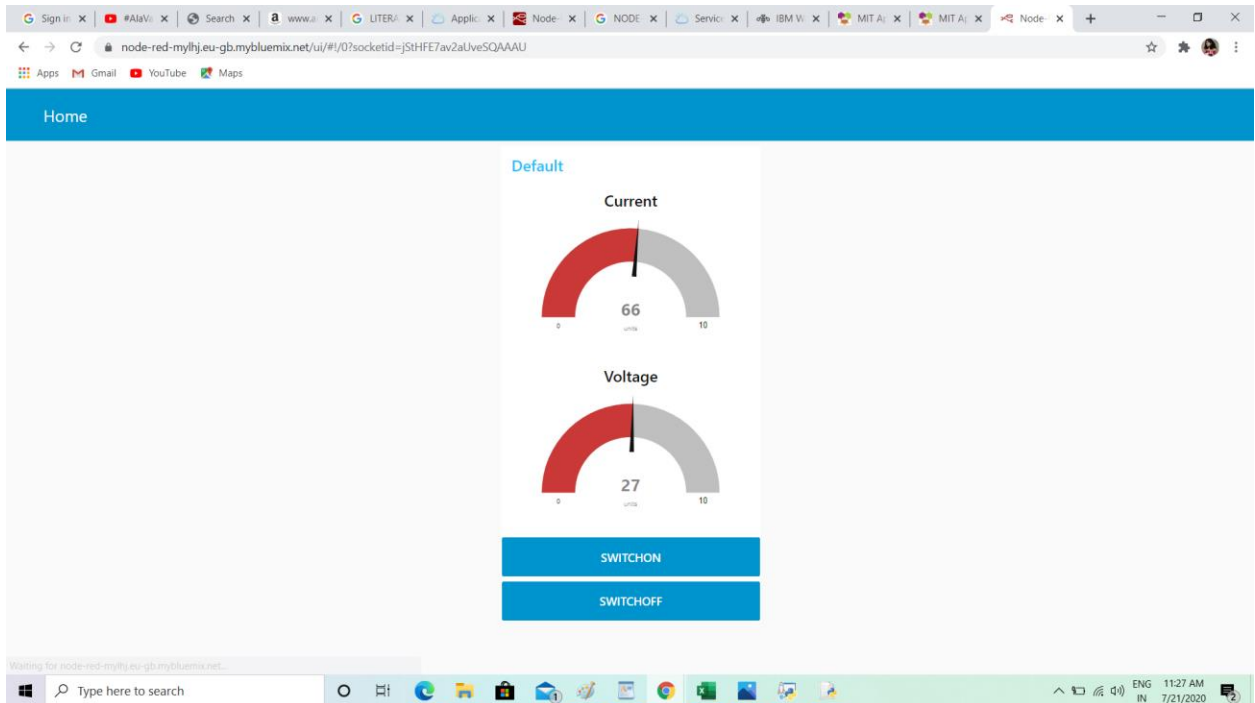
- Authentication: API Key
- API Key: c6480528.738438
- Output Type: Device Command
- Device Type: raspberrypi
- Device Id: 123456
- Command Type: home
- Format: json
- Data: data
- QoS: 0
- Name: IBM IoT
- Service: registered

The debug console on the far right shows a series of messages from the device, including current and voltage readings.

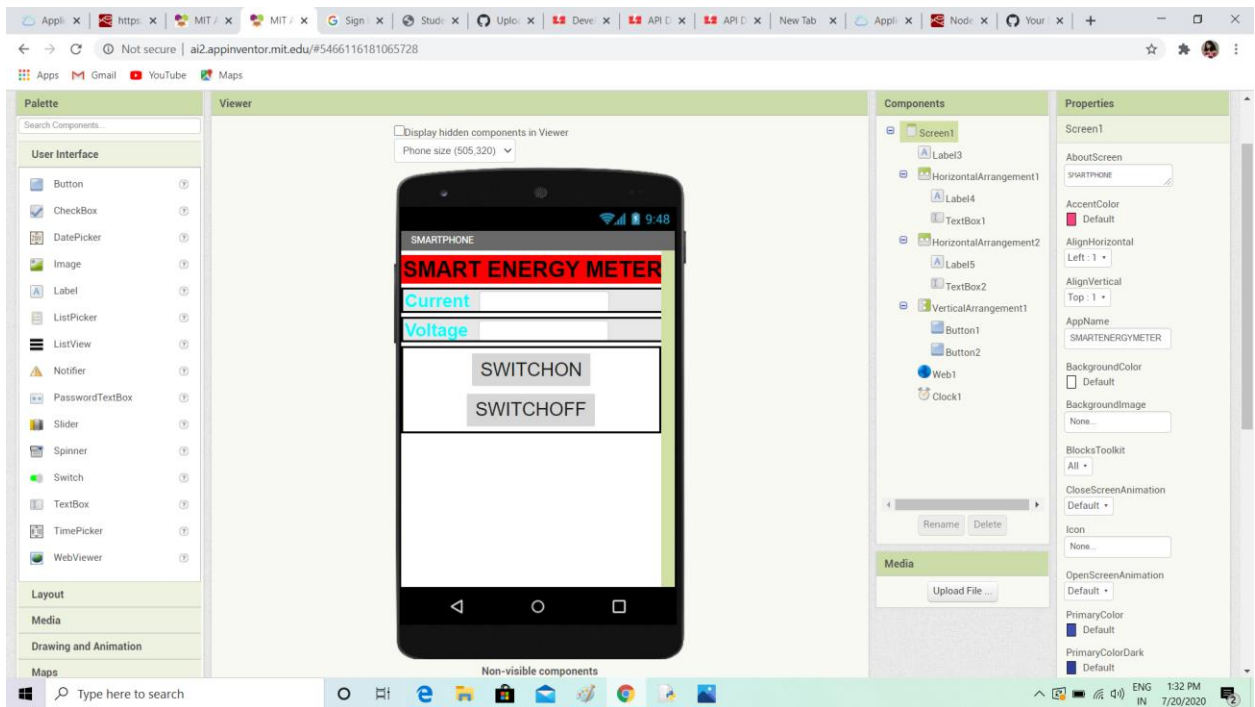
FLOW CHART:



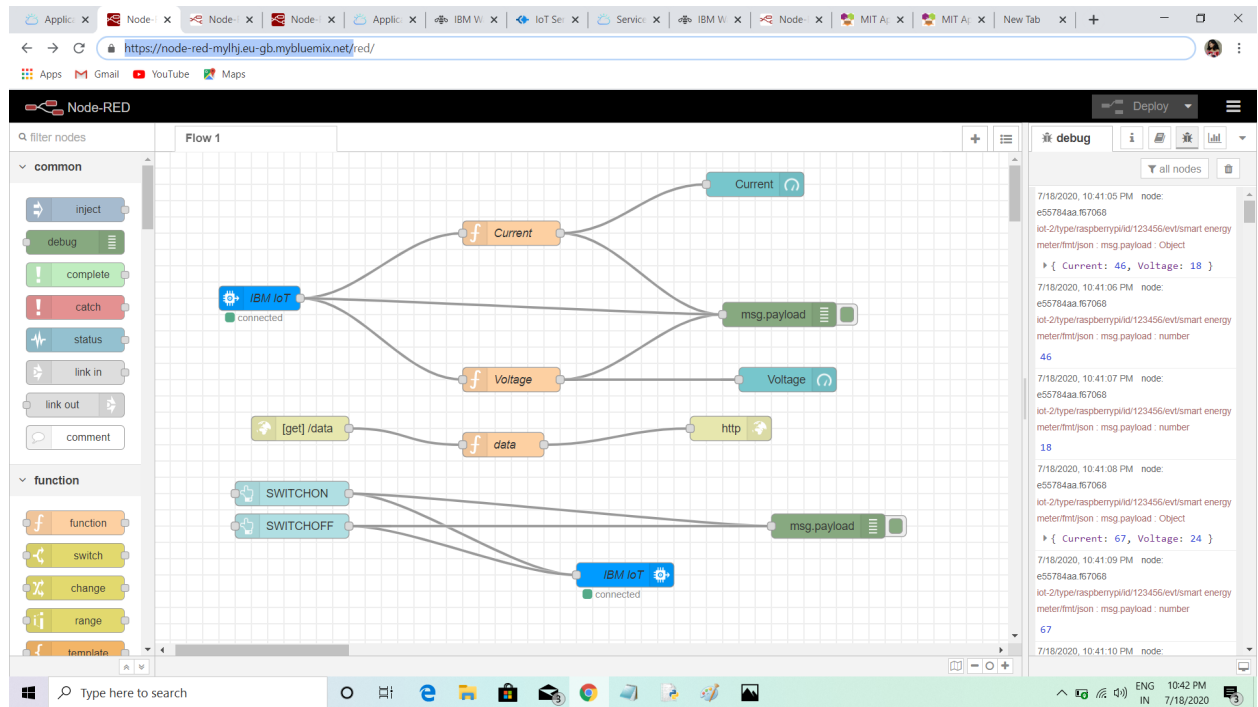
Result: The web application generated by the above designing procedure is as follows:



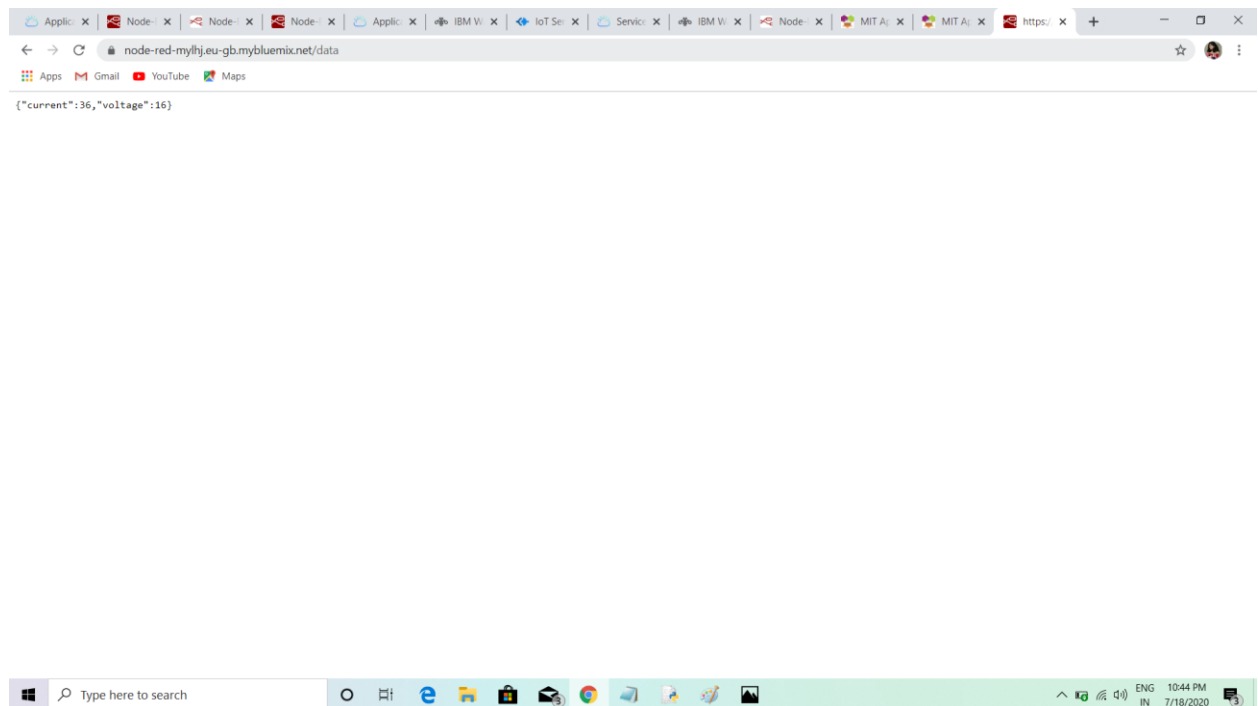
- Now to display the sensor values in the mobile application search for mit app inventor and click on MIT APP INVENTOR and click to create app and login



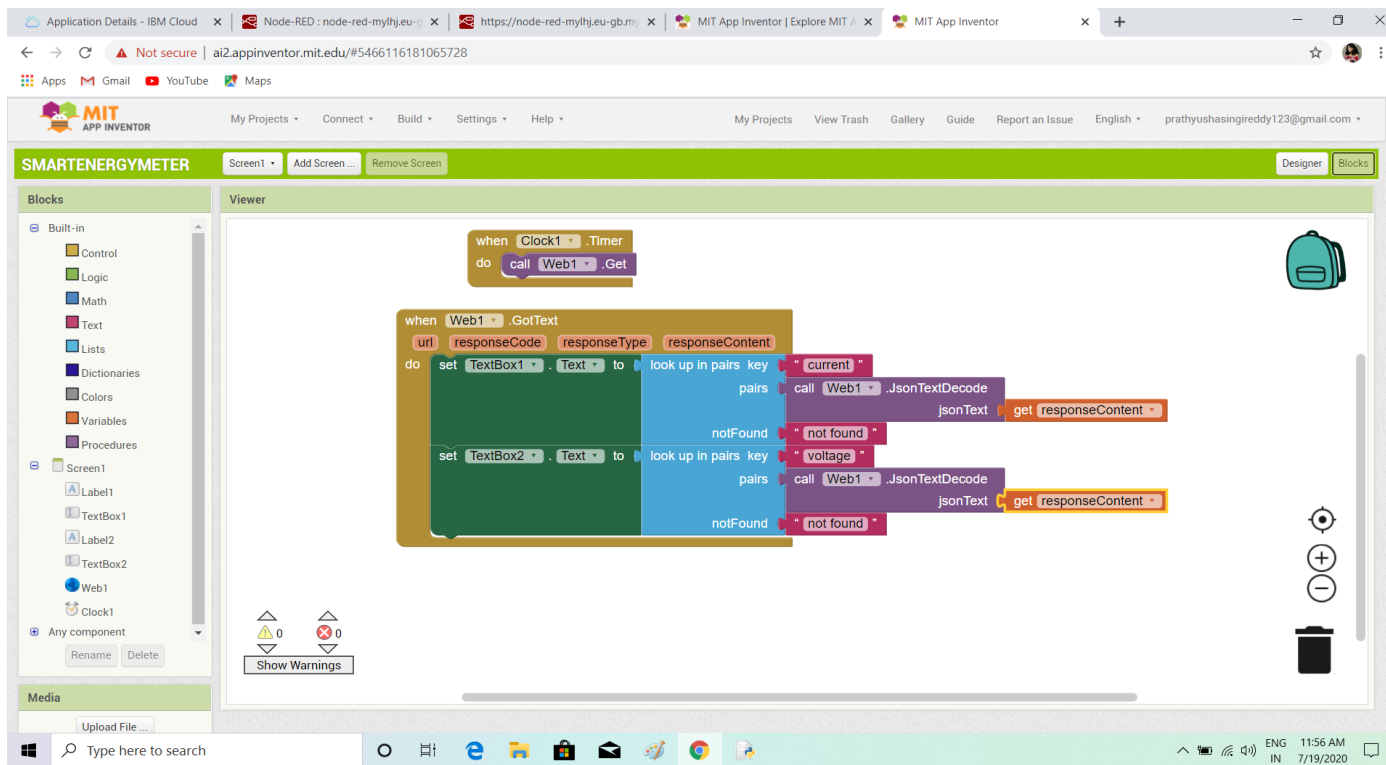
- Now we get data into node-red and to send this data into mobile app we need some interface HTTP



Output:



- Controlling the appliances using mobile app:



- Notification to mobile when the sensor value crosses the threshold

9:25



Fast2SMS

VK-FSTSMS



1

9182518240:your current consumption is high please reduce your consumption

1

9182518240:your current consumption is high please reduce your consumption

1

9182518240:your current consumption is high please reduce your consumption

1

9182518240:your current consumption is high please reduce your consumption

1

9182518240:your current consumption is high please reduce your consumption

1

9182518240:your current consumption is high please reduce your consumption

1

9182518240:your current consumption is high please reduce your consumption

1

9182518240:your current consumption is high please reduce your consumption



Send message



Python program for current and voltage sensor:

```
import time

import sys

import ibmiotf.application

import ibmiotf.device

import random

#Provide your IBM Watson Device Credentials

organization = "p4ogaa"

deviceType = "raspberrypi"

deviceId = "123456"

authMethod = "token"

authToken = "12345678"


def myCommandCallback(cmd):

    print("Command received: %s" % cmd.data)#Commands

    print(type(cmd.data))

    i=cmd.data['command']

    if i=='switchon':

        print("switch is on")

    elif i=='switchoff':

        print("switch 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:

    volt = random.randint(10, 40)

    #print(volt)

    cur = random.randint(30, 80)

    #Send Current & Voltage to IBM Watson

    data = { 'Current': cur, 'Voltage': volt }

    #print (data)

    def myOnPublishCallback():

        print ("Published Current = %s C" % cur, "Voltage = %s %%" % volt, "to IBM Watson")

    success = deviceCli.publishEvent("smart energy meter", "json", data, qos=0,
on_publish=myOnPublishCallback)

    if not success:

        print("Not connected to IoT")

        time.sleep(2)

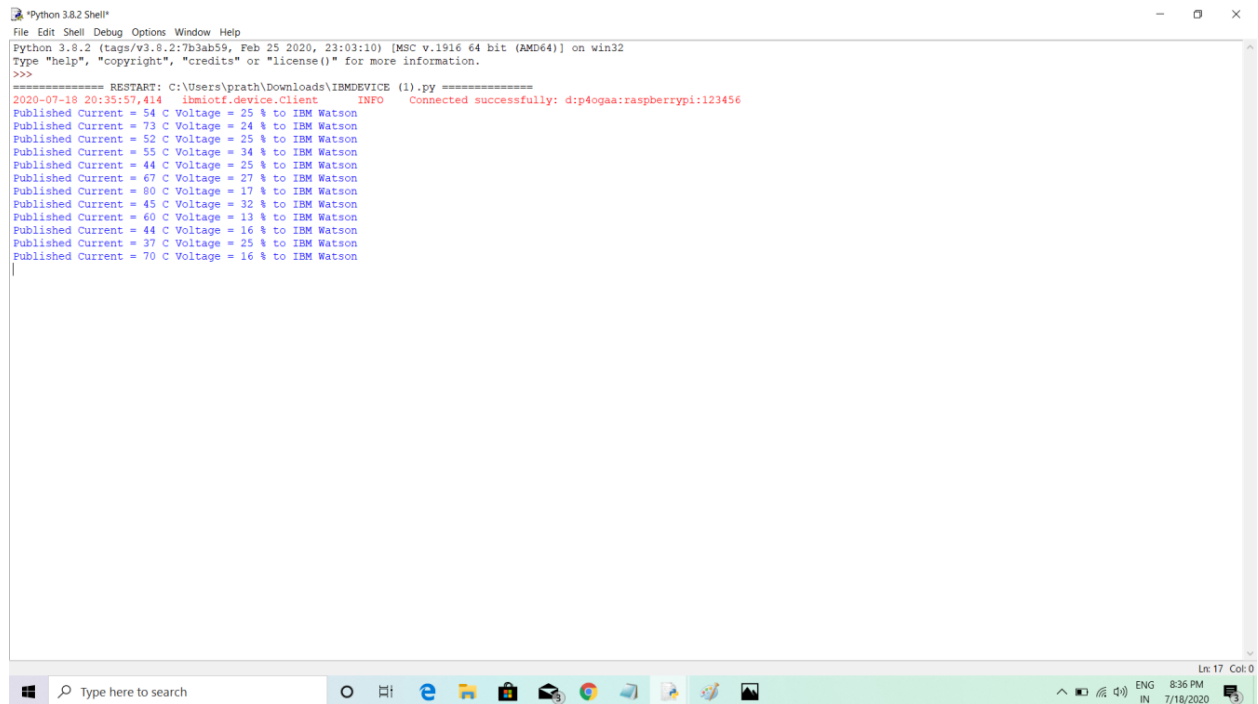
    deviceCli.commandCallback = myCommandCallback

```

Disconnect the device and application from the cloud

```
deviceCli.disconnect()
```

Output:



```
Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 23:03:10) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\prath\Downloads\IBMDEVICE (1).py =====
2020-07-18 20:35:57,414  lbmiotf.device.Client  INFO  Connected successfully: d:p4ogaa:raspberrypi:123456
Published Current = 54 C Voltage = 25 % to IBM Watson
Published Current = 73 C Voltage = 24 % to IBM Watson
Published Current = 52 C Voltage = 25 % to IBM Watson
Published Current = 55 C Voltage = 34 % to IBM Watson
Published Current = 44 C Voltage = 25 % to IBM Watson
Published Current = 67 C Voltage = 27 % to IBM Watson
Published Current = 80 C Voltage = 17 % to IBM Watson
Published Current = 45 C Voltage = 32 % to IBM Watson
Published Current = 60 C Voltage = 13 % to IBM Watson
Published Current = 44 C Voltage = 16 % to IBM Watson
Published Current = 37 C Voltage = 25 % to IBM Watson
Published Current = 70 C Voltage = 16 % to IBM Watson
|
```

Conclusion: An attempt has been made to make a model of 'IOT Based Smart Energy Meter'. The propagated model is used to calculate the energy consumption of the household, and even make energy unit reading easy and accurate. Hence it reduces the wastage of energy and brings awareness among all.

Future Scope: Smart energy meters have been introduced as a means to modernize the grids and to bring about operational changes such as reduce nontechnical losses, introduce remote reading and switching or simplify the billing procedures