***Energy Management***

**1.Introduction**:

1.1 Overview:

In general cases, people find it difficult to check manually very often. Instead, we come across this energy management system.

1.2 Purpose:

The energy management system helps us to get the readings of our ammeter, voltmeter, wattmeter in just a single touch in our energy management app in our mobile phones where it will easy for the workers in industries etc to monitor them at all times without going near the meters.

**2.Literature Survey**

2.1 Existing Problem:

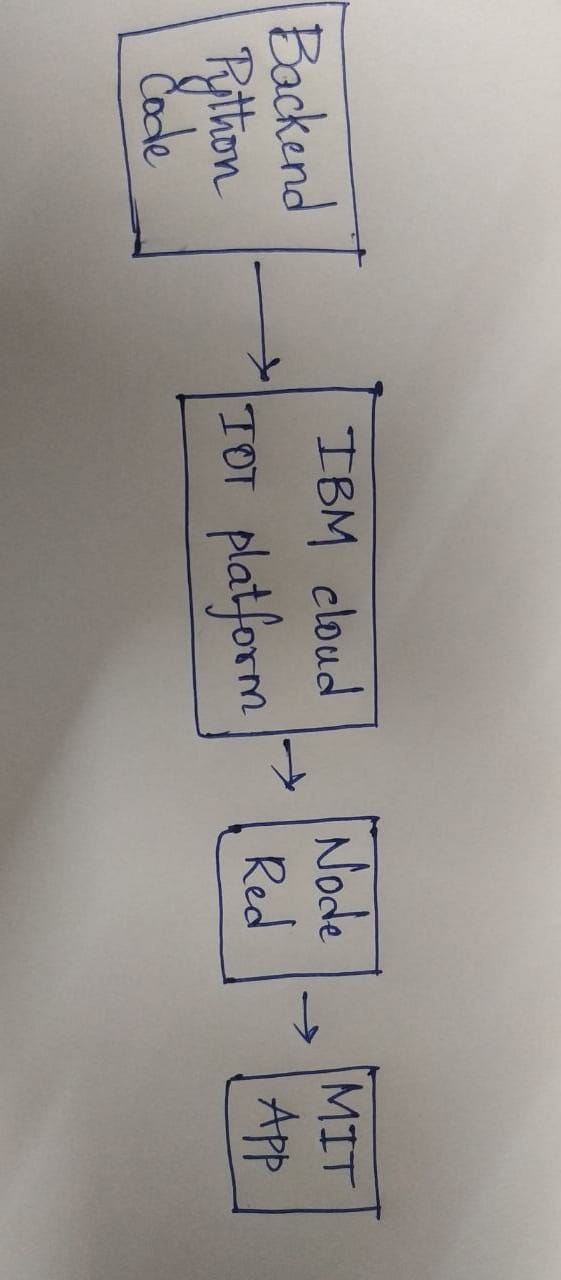
Sometimes it's difficult to monitor the meters in industries because there might be some short circuit, it might affect the workers to do that.

2.2 Suggested Solution:

Workers need not to get into the restricted area to monitor them. But instead, they can do it in their phone itself using this system. We get ammeter, voltmeter, wattmeter readings in the phone itself

**3.Theoretical Analysis**

3.1 Block diagram:

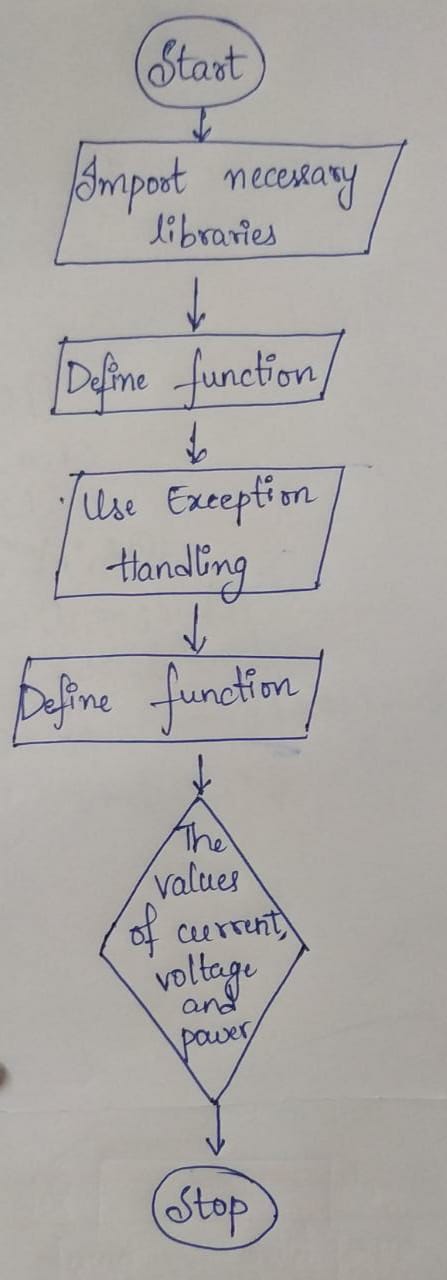


Using the cloud credentials, we write the backend code in python and the information is sent into IBM IoT platform,from there we send the information to nodered, by placing the nodered link in MIT app, we get the data in MIT app.

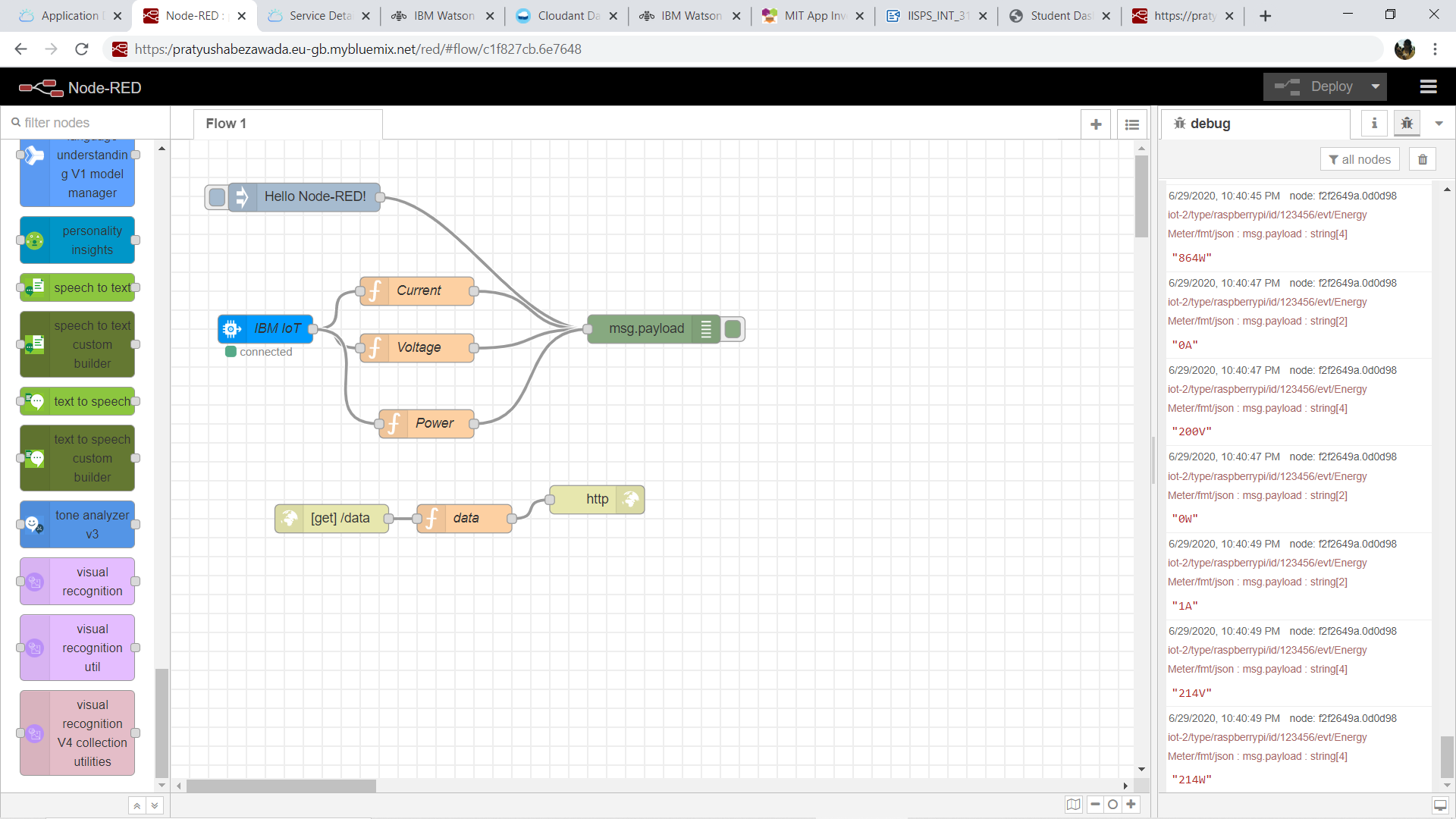
**4. Experimental Investigations**

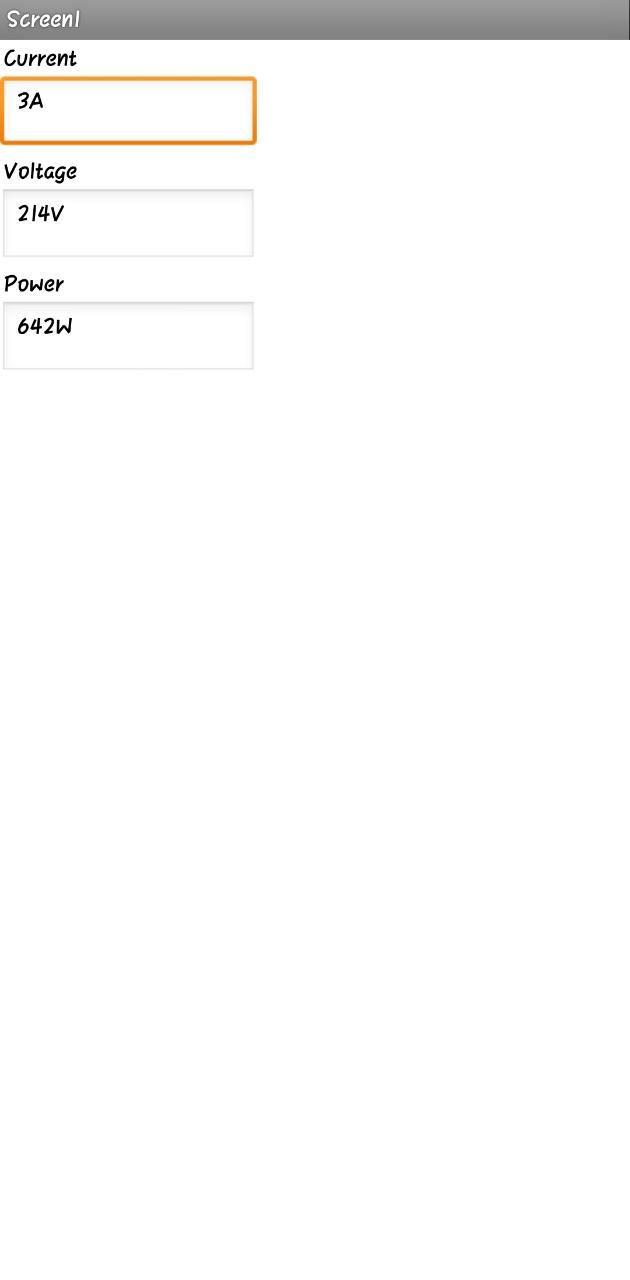
By using energy management system, we can monitor the current, voltage, power in the industry where we can see whether we have a good and satisfied readings of them.

**5.Flowchart**

****

**6. Result**



****

**7.Advantages and Disadvantages**

7.1 Advantages:

We can monitor the ammeter, voltmeter and wattmeter from anywhere round the globe.

**8. Applications:**

These systems can be used in industries, factories, gated communities etc to monitor the current, voltage and power readings.

**9. Conclusion:**

We can now proudly say that technology is not only used in IT sectors but also, in every sector. Thus, advancement of technology is always a boon.

**10.Future Scope**

It's not possible to monitor the energy meters all the time by sitting at the room provided for them, it might be too risky if there's any short circuit, sometimes the meters can blow off too. So, in order to avoid them we use energy management system, where we can monitor by sitting in any place the globe.

**Appendix**

import time

import sys

import ibmiotf.application

import ibmiotf.device

import random

#Provide your IBM Watson Device Credentials

organization = "dcwm2n"

deviceType = "raspberrypi"

deviceId = "123456"

authMethod = "token"

authToken = "12345678"

def myCommandCallback(cmd):

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

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:

i=random.randint(0,5)

v=random.randint(200,230)

p=v\*i

data = { 'Current' : str(i)+'A', 'Voltage': str(v)+'V', 'Power' : str(p)+'W'}

print (data)

def myOnPublishCallback():

print ("Published Current = %s A" % i, "Voltage = %s V" % v,"Power = %s W" %p, "to IBM Watson")

success = deviceCli.publishEvent("Energy Meter", "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()