**IoT Analytics in Energy Management**

**By Team-02**

1) Mohammad Asgar Ali Ansari (17481A04C9)

2) P.S.N.Alekhya (17481A04F2)

3) P.Bhuvan Chand (17481A04F8)

4) M.Vasu Devarao (17481A04C2)

1  **INTRODUCTION**

1.1 **Overview**

1. Purpose of the Project
2. Existing Problem
3. Proposed Solution
4. Block Diagram
5. Hardware/Software Designing
6. Experimental Investigations
7. Flowchart
8. Result
9. Advantages & Disadvantages
10. Applications
11. Conclusions
12. Future Scope
13. Bibliography

1.2 **Purpose**

1. Energy management is done by this smart IoT analytic device. It calculates the load current that has been used.
2. The smart device calculates the available energy at both the power distribution Grid and Solar.
3. The power details from both the power sources are uploaded to the cloud.
4. Using the machine learning model we will predict the cost of the current for the next hours.
5. The predicted cost will be sent to the device, at the device level, there we will decide which power source is to be used, either grid or solar power and it will be connected according to that.
6. At maximum peak hours if both power sources are able to drive the load then the solar power source will get the higher priority, thus minimizing the grid usage we can save renewable energy.

**2 LITERATURE SURVEY**

2.1  **Existing Problem**

The problem with the present electricity bill system is that, the bill is not in basic terminology, for the users to understand. It involves several terms and calculations in the bill.

Also, during the peak hours, both the power sources (i.e.,) grid power source and solar power source are able to drive the load, thus a larger use of grid power source can be occurred.

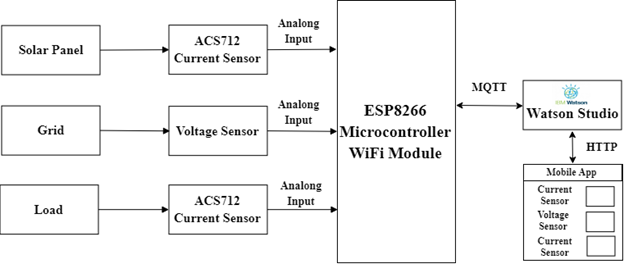
2.2 **Proposed Solution**

We can make use of IoT Analytics in Energy Management, such that during peak hours, when both the grid power source and solar power source are able to drive load, then the usage of grid power source is minimized and solar power source is used, thus saving renewable energy.

Also, with the usage of IoT, a automated electricity bill may be delivered to users device which is in simple and basic language consisting of only voltage, current, and total bill to be paid. This helps the user to understand the energy consumption.

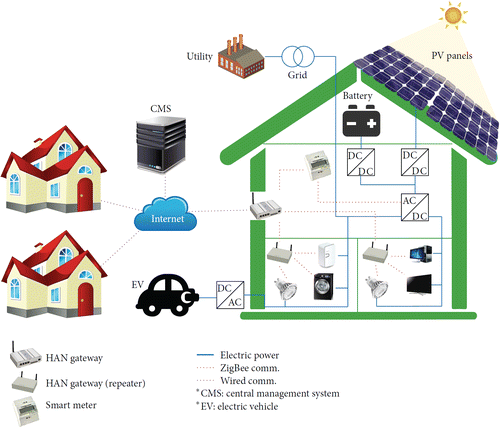
**3 Theoretical Analysis**

**3.1 Block Diagram**



**3.2 Hardware/Software Designing**

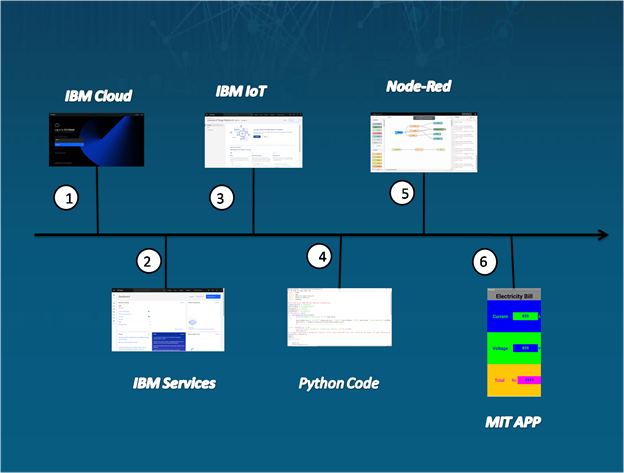
The Software designing involves generating a sequence of values for current and voltage. With the help of current & voltage values, we can calculate the current bill.IBM Cloud Service is used to create Internet of Things platform. In IoT platform, a virtual Raspberry Pi device is created. After creating the design we will get the device credentials. We use these credentials in Python program to integrate the Node-Red platform with IoT. With the help of MIT APP Inverter an app is designed & integrated with the Node-Red to observe the values.



**4 Experiment Investigation**

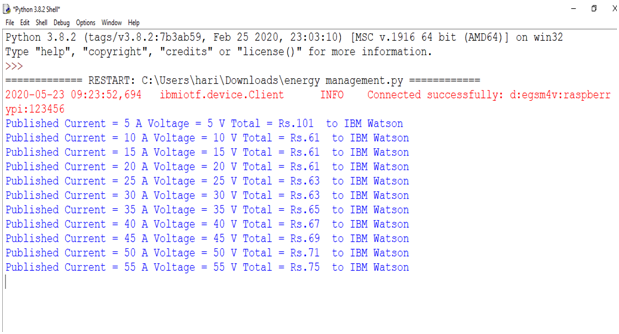
In order to complete our project,we collected the required data from Google & research papers. After getting the complete knowledge, we worked according to our roles in the project. At first we created the IBM Cloud account then, we created the Internet of Things Platform after we wrote a python code in IDLE to connect IBM IoT Platform. Next we created the Node-Red Services. This service helps us to show virtual flow graphs. We connected Node-Red to IBM IoT to get the values of current, voltage and calculated bills. From Node-Red we have sent values to the MIT APP. From app we can view the Current, Voltage & Bill details clearly.

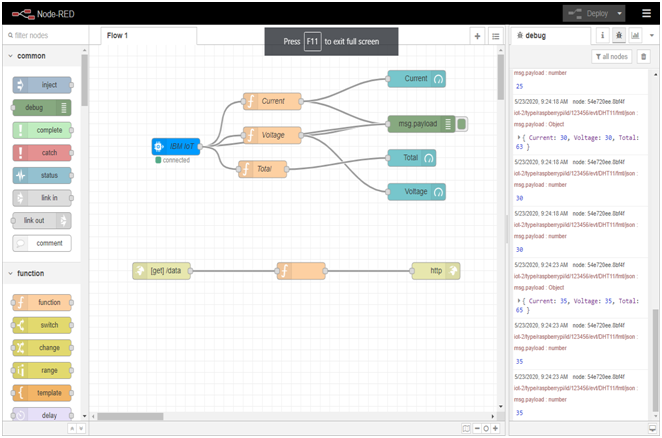
**5 FLOWCHART**



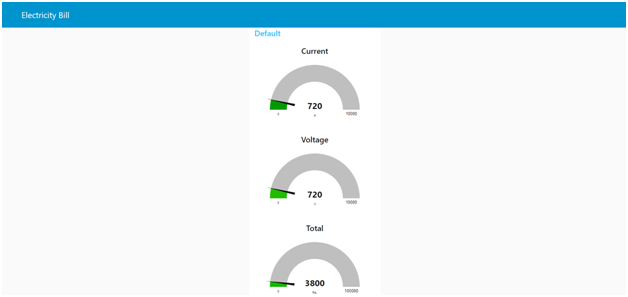
**6 RESULTS**

**Python Code:**

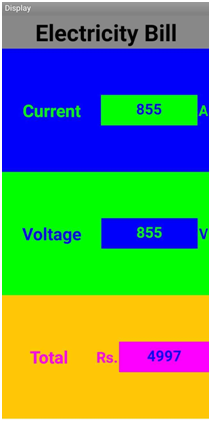


**Node-Red:**

**Node-Red UI:**



**MIT APP:**



**7 ADVANTAGES & DISADVANTAGES**

**Advantages:**

1) Automation of daily monitoring the devices

2) Time & money save

3) Efficient working

4) Easy monitoring.

**Disadvantages:**

1) Needs Highly Qualified Professionals.

2) Lesser Employment

3) Technology Takes Control of Life

4) Safety

**8 APPLICATIONS**

We can implement in various fields like

1) Health care

2) Retail

3) Manufacturing

4) Automobile Industries etc.

**9 CONCLUSIONS**

Energy systems are on the threshold of a new transition era. Large-scale deployment of VRE in distributed energy systems and the need for efficient use of energy calls for system-wide, integrated approaches to minimize the socio-economic-environmental impacts of energy systems. In this respect, modern technologies such as IoT can help the energy sector transform from a central, hierarchical supply chain to a decentralized, smart, and optimized system. In this paper, we review the role of IoT in the energy sector in general, and in the context of smart grids particularly.

**10 FUTURE SCOPE**

Applying current IoT systems for providing energy efficient solutions in the energy sector has many advantages highlighted in previous sections. However, for deploying IoT in the energy domain, new solutions and trends are needed to improve the performance of IoT and overcome the associated challenges. By improving the performance we can save for power & also provide the privacy & security aspects also. With help of this we can monitor & complete the works easily.

**11 BIBLIOGRAPHY**

<https://cloud.ibm.com/registration>

<https://cloud.ibm.com/catalog/services/watson-studio>

[http://Ai2.appinventor.mit.edu](http://ai2.appinventor.mit.edu/)

<https://flows.nodered.org/node/node-red-dashboard>

<https://developer.ibm.com/recipes/tutorials/ui-dashboard-for-iot-device-data-using-node-red/>

<https://appinventor.mit.edu/>

**APPENDIX**

**A. Source Code**

import time

import sys

import ibmiotf.application

import ibmiotf.device

import random

#Provide your IBM Watson Device Credentials

organization = "egsm4v"

deviceType = "raspberrypi"

deviceId = "123456"

authMethod = "token"

authToken = "12345678"

# Initialize GPIO

def myCommandCallback(cmd):

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

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()

amp=0

volt=0

while True:

List=[]

Power=(amp\*volt)

units=(Power/1000)

if(units>0 and units<=100):

payAmount=units\*2.96

fixedcharge=30.00

elif(units>100 and units<=300):

payAmount=(100\*2.96)+(units-100)\*5.56

fixedcharge=50.00

elif(units>300 and units<=500):

payAmount=(100\*2.96)+(300-100)\*5.56+(units-200)\*9.16

fixedcharge=50.00

elif(units>500):

payAmount=(100\*2.96)+(300-100)\*5.56+(500-300)\*9.16+(units-500)\*10.61

fixedcharge=100.00

else:

fixedcharge=50.00

payAmount=0

Total= payAmount+fixedcharge

Total= int(Total)

List.append(Total)

summation=0

for i in List:

summation=i+summation

Total+=summation

amp+=5

volt+=5

Total+=1

data = { 'Current' : amp, 'Voltage': volt, 'Total' : Total }

#print (data)

def myOnPublishCallback():

print ("Published Current = %s A" % amp, "Voltage = %s V" % volt, "Total = Rs.%s " % Total, "to IBM Watson")

success = deviceCli.publishEvent("DHT11", "json", data, qos=0, on\_publish=myOnPublishCallback)

if not success:

print("Not connected to IoTF")

time.sleep(5)

deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud

deviceCli.disconnect()