Project Report

on

Home automation system with Smart-meter

For the course

ECE3501 (Internet of Things)

Presented By :

Siddharth Khachane

18BCE1013

Presented To :

Dr. Rekha D

Contents

Abstract

The title of this project is “Home Automation system implementation with smart-meter”. I have created a custom home automation system with a smart-meter of my own which uses MQTT protocol and power used in each device can be calculated so as to monitor each and every device. I have also created my own smart-meter that displays power usage on a graph. I am also using ThingSpeak which is an analytics platform to aggregate, visualize and analyze live data stream in the cloud.

Introduction

The project contains different nodes that help to accomplish the implementation of the project. The basic idea of the project is to generate realistic sensor values using inject and function nodes and to store the generated values in a database using sqlite, then gather data from the database to calculate daily and monthly power usage of the devices and then display the statistics on graph using node-red dashboard. I am also using ThingSpeak for displaying end of the day power usage data online using ThingSpeak API.

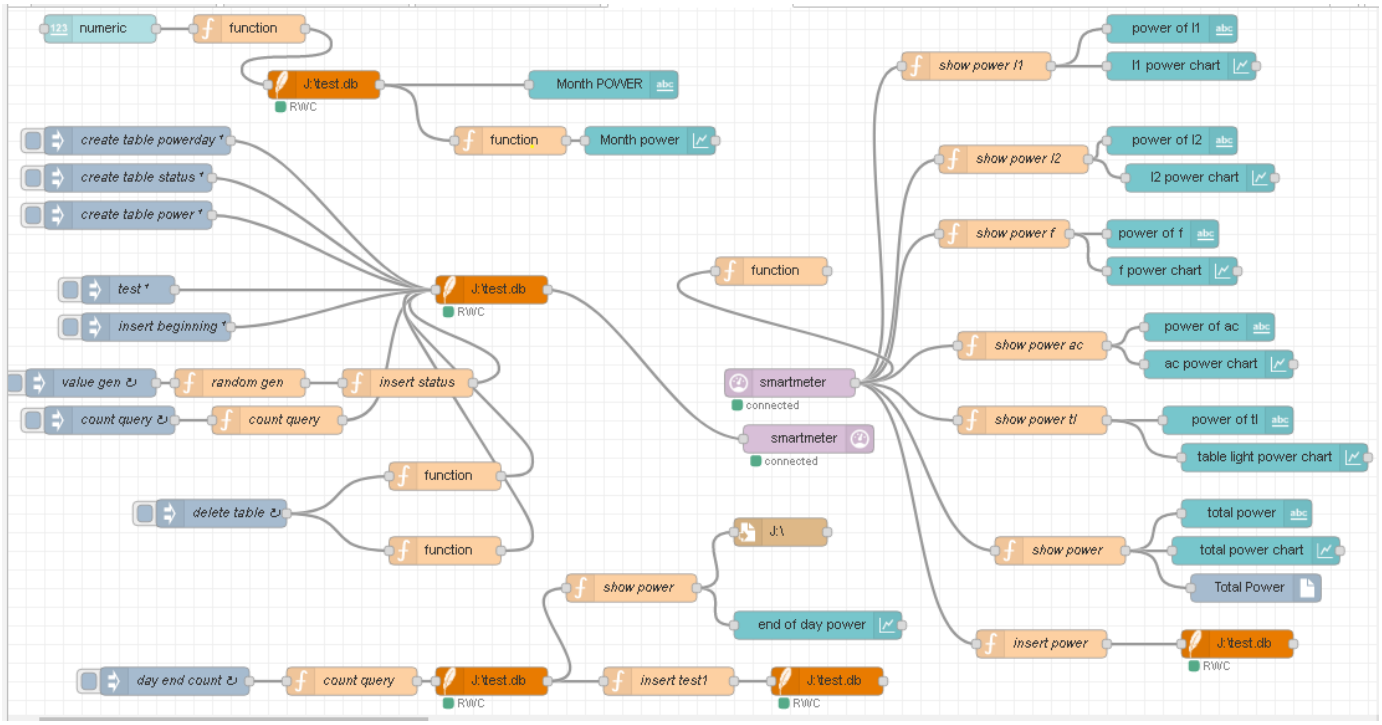
This project, if made realistic, can be implemented in smart homes. The regular check up of power usage is made easy using this project thus if there is any excessive power consumption of any device, it is reflected immediately thus preventing any accidents by alerting the owner about the excessive usage.

Software Component :

- Node Red & Node Red Dashboard
- DB Browser for SQLite
- SQLite (for storage)
- ThingSpeak

Working of the Project

FULL FLOW:



I have ensured that the randomly generated values are applicable in the real world scenario.

Devices:

Light 1,Light 2, Fan, AC, Table Lamp.

Sensors:

Light1:Light Intensity, PIR(Pyro Infrared) Sensor

Light2: Light Intensity, PIR(Pyro Infrared) Sensor

Fan: Temperature, PIR(Pyro Infrared) Sensor

AC: Temperature, PIR(Pyro Infrared) Sensor

Table Lamp: Pressure sensor (on chair).

Light Intensity: 0-4096(Values generated).

Threshold: according to our requirement.

PIR sensor: 0-15 microns (Radiation)

Threshold: 10microns for human

Temperature:20-40 degree celcius.

Threshold: Fan: 25<Temp<30

AC: Temp>30

Pressure: 0-100000Pa

Threshold:60000pa

Database:

Power		CREATE TABLE Power(l1p varchar(255),l2p varchar(255),acs varchar(255),fs varchar(255),tls varchar(255))
l1p	varchar(255)	"l1p" varchar(255)
l2p	varchar(255)	"l2p" varchar(255)
acs	varchar(255)	"acs" varchar(255)
fs	varchar(255)	"fs" varchar(255)
tls	varchar(255)	"tls" varchar(255)

Status		CREATE TABLE Status(l1s varchar(255),l2s varchar(255),acs varchar(255),fs varchar(255),tls varchar(255))
l1s	varchar(255)	"l1s" varchar(255)
l2s	varchar(255)	"l2s" varchar(255)
acs	varchar(255)	"acs" varchar(255)
fs	varchar(255)	"fs" varchar(255)
tls	varchar(255)	"tls" varchar(255)

test1		CREATE TABLE test1(date INTEGER PRIMARY KEY AUTOINCREMENT, eodpower VARCHAR(255),time VARCHAR(255))
date	INTEGER	"date" INTEGER
eodpower	VARCHAR(255)	"eodpower" VARCHAR(255)
time	VARCHAR(255)	"time" VARCHAR(255)

Table: Status

	l1s	l2s	acs	fs	tls
	Filter	Filter	Filter	Filter	Filter
1	ON	OFF	OFF	ON	OFF
2	OFF	OFF	OFF	OFF	OFF
3	OFF	OFF	ON	OFF	OFF
4	OFF	OFF	OFF	OFF	OFF
5	ON	ON	OFF	OFF	OFF
6	OFF	OFF	OFF	OFF	OFF
7	OFF	OFF	OFF	OFF	OFF
8	OFF	OFF	OFF	OFF	ON
9	OFF	OFF	OFF	OFF	ON
10	ON	OFF	OFF	OFF	OFF
11	OFF	OFF	OFF	OFF	ON
12	OFF	OFF	OFF	OFF	ON
13	OFF	OFF	ON	OFF	ON
14	ON	ON	OFF	ON	OFF

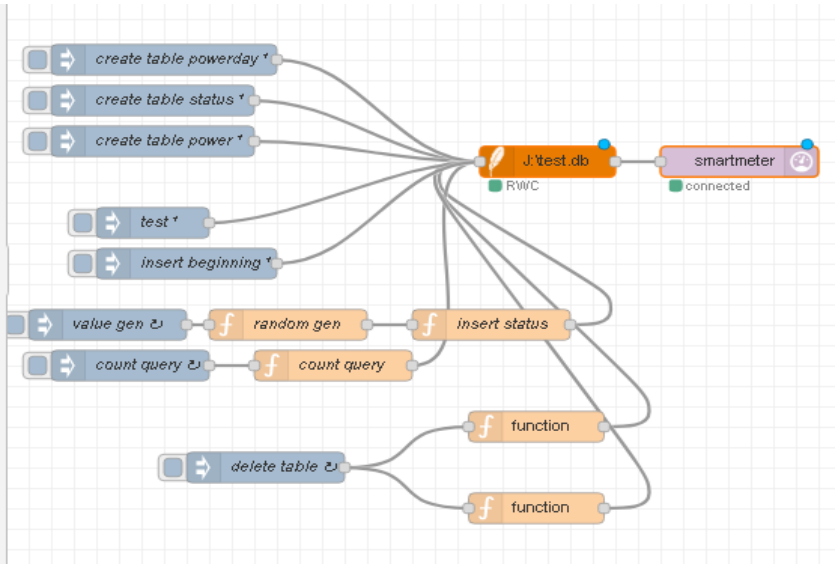
Table: test1

	date	eodpower	time
	Filter	Filter	Filter
1	4	11.466	Fri Oct...
2	5	13.559	Fri Oct...
3	6	7....	Fri Oct...
4	7	10.105	Fri Oct...
5	8	4.105	Fri Oct...
6	9	7.479	Fri Oct...
7	10	10.754	Fri Oct...
8	11	6.561	Fri Oct...
9	12	7.475	Fri Oct...
10	13	5.098	Fri Oct...
11	14	1....	Fri Oct...
12	15	5.191	Fri Oct...

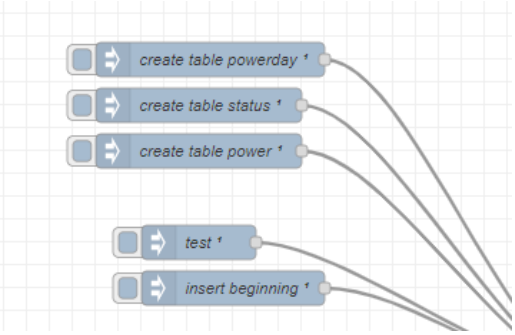
Flow Explanation

For the sake of convenience, the flow is broken into four parts which are further divided into subparts:

Part 1



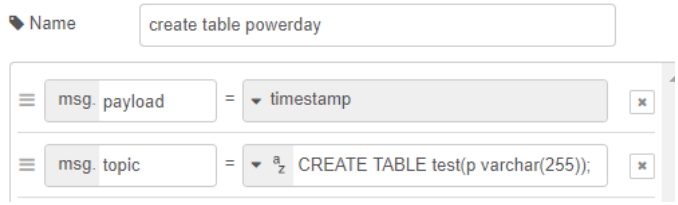
Part 1.a



This part contains nodes that create tables (Power table, Status table and end of day power table).

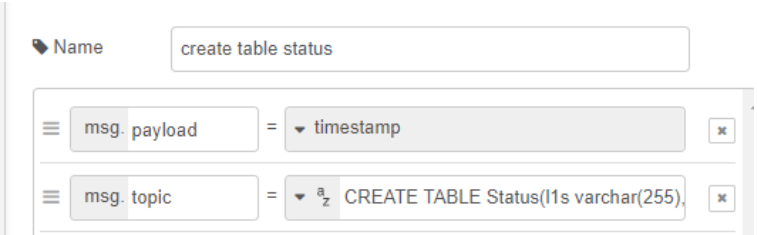
create table powerday contains the sql query –

CREATE TABLE test(p varchar(255));



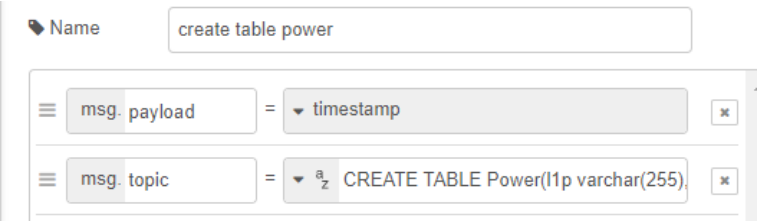
create table status contains the sql query –

CREATE TABLE Status(l1s varchar(255),l2s varchar(255),acs varchar(255),fs varchar(255),tls varchar(255));



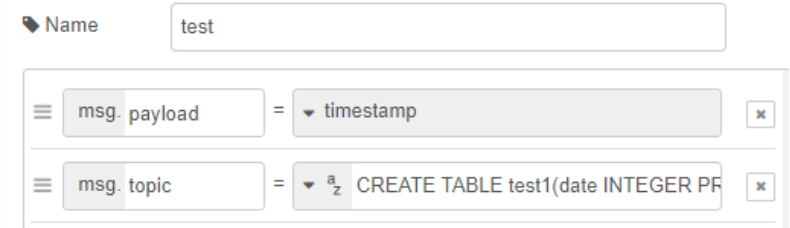
create table power contains the sql query –

CREATE TABLE Power(l1p varchar(255),l2p varchar(255),acs varchar(255),fs varchar(255),tls varchar(255));



test contains sql query –

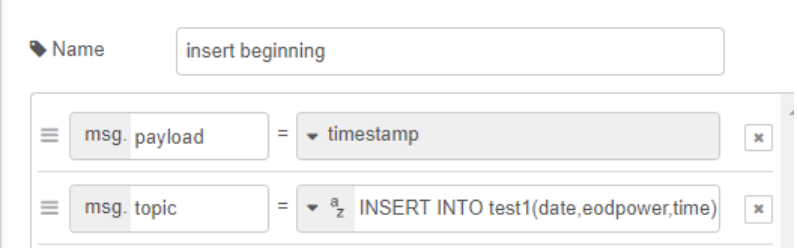
CREATE TABLE test1(date INTEGER PRIMARY KEY AUTOINCREMENT, eodpower VARCHAR(255));



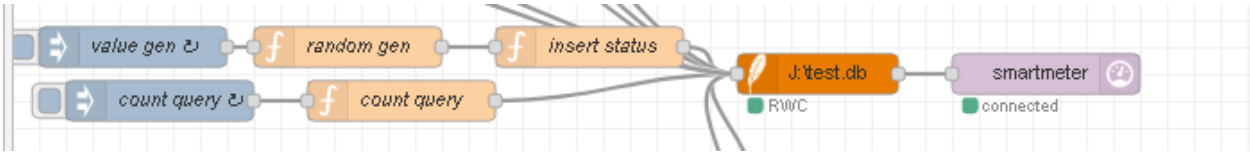
insert beginning basically initializes the value of “date” in the table test1 given above which is autoincremented. It contains sql query –

This inject node is essential for the **AUTOINCREMENT** of the date to initialize it’s value.

INSERT INTO test1(date,eodpower) VALUES(0,0);



Part 1.b



<u>This part generates sensor data which is then stored in the database (within the tables created before)</u>

value gen is just a repeater that repeats every 1 second (since 1 second = 1 hour) that triggers the

random gen function since we have to generate random value which is applicable in the real life scenario for a specific interval of time.

Random gen generates the random sensor values. It contains the following json code.

<u>Random gen Function Code:</u>

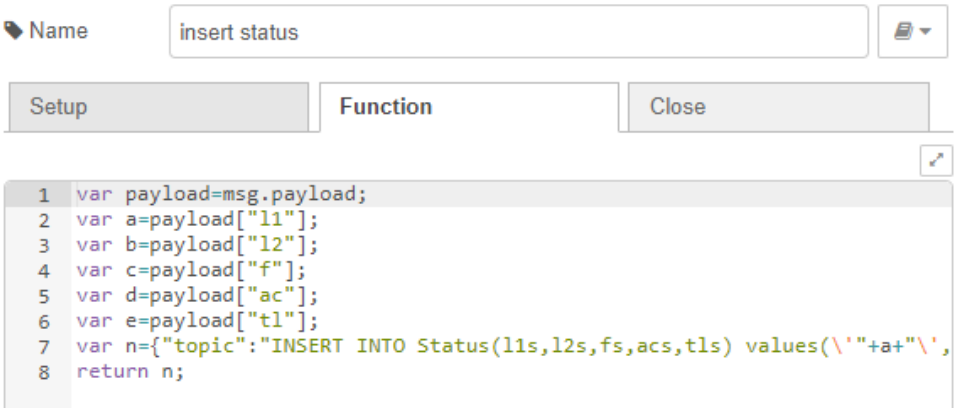
```
var a=Math.round(Math.random()*4095);

var b=Math.round(Math.random()*4095);

var c=Math.round(Math.random()*20)+20;

var d=Math.round(Math.random()*100000);

var e=Math.round(Math.random()*1);

var f1=Math.round(Math.random()*15);

var on="ON";

var off="OFF";
```

```
var L1;

var L2;

var f;

var ac;

var tl;

if(a<3000 && f1>=10)

{   L1 = on;}

else if(a>=3000 || f1<10)

{   L1 = off;}

if(b<2000 && f1>=10)

{   L2 = on;}

else if(b>=2000 || f1<10)

{   L2 = off;}

if(c>=25&&c<=30  && f1>=10)

{   f = on;

}else if(c<=25||c>=30  || f1<10)

{   f = off;}

if(c>30 && f1>=10)

{   ac= on;}

else if(c<=30 || f1<10)

{   ac = off;}

if(d>60000)

{   tl = on;}

else if(d<=60000)

{   tl= off;}
```

msg.payload={"l1":L1,"l2":L2,"f":f,"ac":ac,"tl":tl};

return msg;

insert status inserts the value generated in the **random gen** node into the status table.

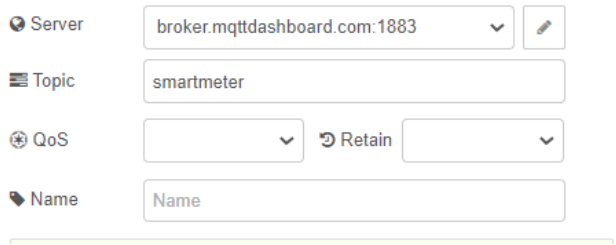


It contains the following code.

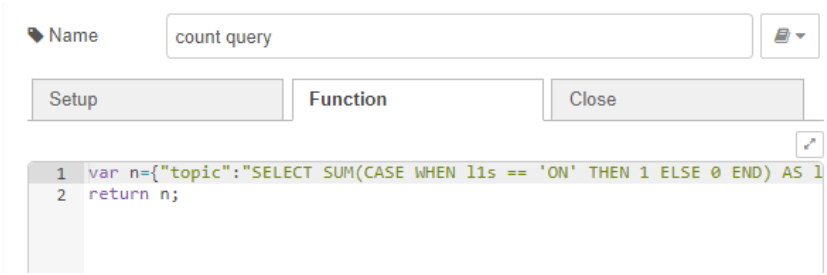
```
var payload=msg.payload;

var a=payload["l1"];

var b=payload["l2"];

var c=payload["f"];

var d=payload["ac"];

var e=payload["tl"];

var n={"topic":"INSERT INTO Status(l1s,l2s,fs,acs,tls)
values(\'"+a+"\',"+"\'"+b+"\',"+"\'"+c+"\',"+"\'"+d+"\',"+"\'"+e+"\')"};

return n;
```

The node next to **insert status** is a **sqlite** node that connects the sqlite database and the tables created within.

smartmeter node is the **mqtt out** node that sends the data to the **mqtt in** node for further use.



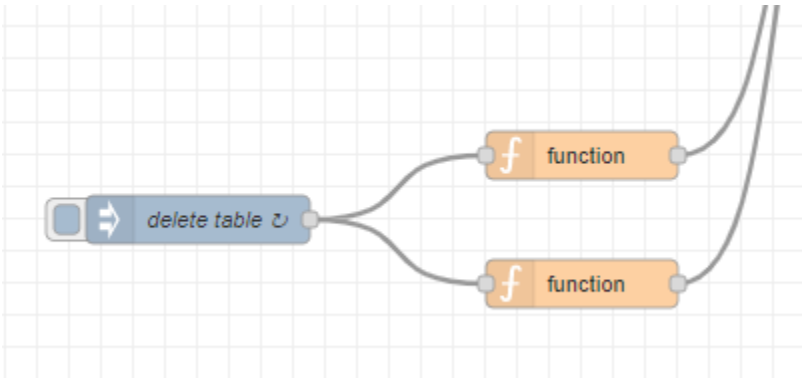
count query is just a repeater that repeats itself every 1 second (since 1 second = 1 hour) that triggers **count query** to calculate count of “ON” status in order to calculate power of a specific device.



count query (next to **count query trigger**) contains sql query to calculate count of “ON” status in the status table. It contains the following code.

```
var n={"topic":"SELECT SUM(CASE WHEN l1s == 'ON' THEN 1 ELSE 0 END) AS l1c,SUM(CASE WHEN l2s == 'ON' THEN 1 ELSE 0 END) AS l2c ,SUM(CASE WHEN fs == 'ON' THEN 1 ELSE 0 END) AS fc ,SUM(CASE WHEN acs == 'ON' THEN 1 ELSE 0 END) AS acc ,SUM(CASE WHEN tls == 'ON' THEN 1 ELSE 0 END) AS tlc FROM Status"};

return n;
```

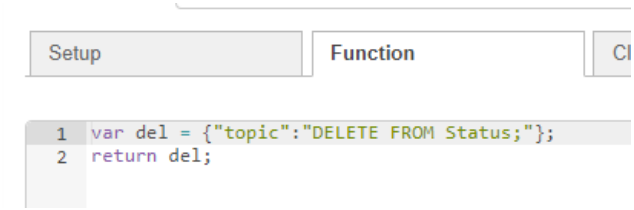
Part 1.c



This part is used to delete the content of all the tables in order to clear data after a certain amount of time.

delete table is just a repeater that repeats itself every 25 seconds (since 1hr=1sec therefore 1 day = 24 seconds)

function (upper) deletes content of status table and contains this code



var del = {"topic":"DELETE FROM Status;"};

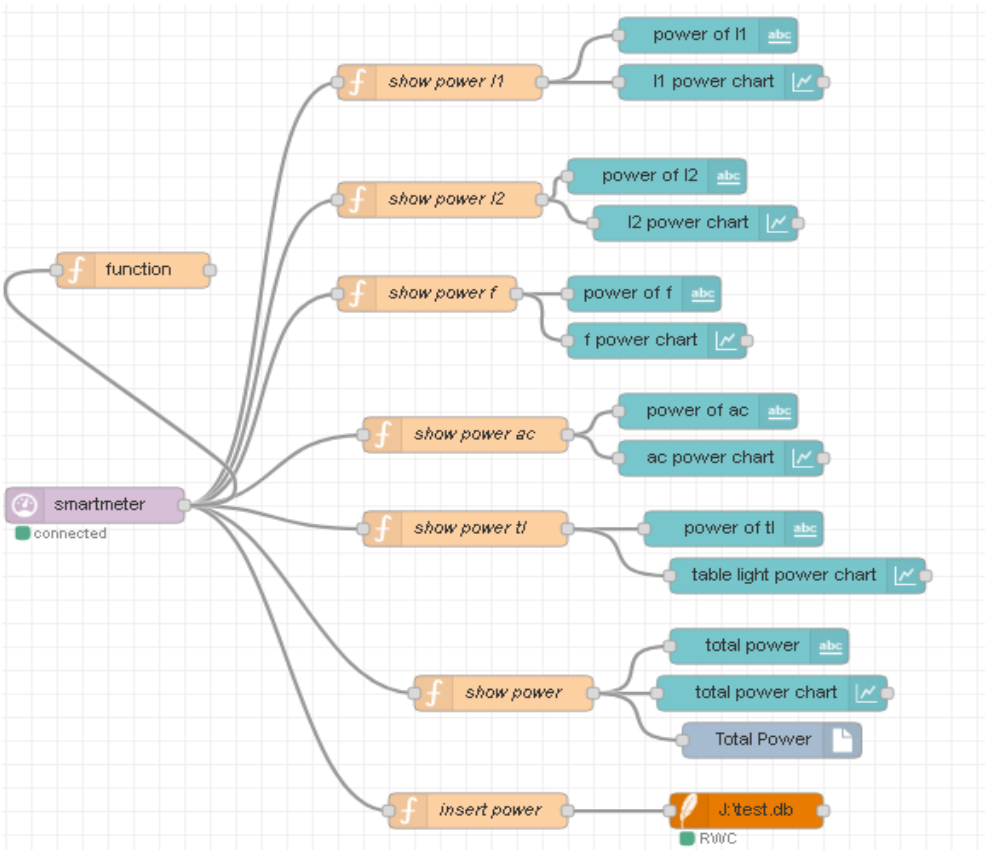
return del;

function (lower) deletes content of power table and contains this code

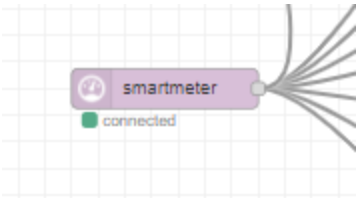
var del = {"topic":"DELETE FROM Power;"};

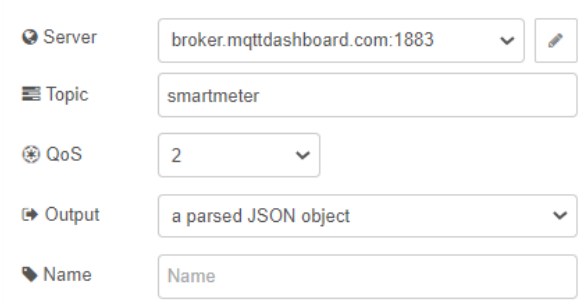
return del;

Part 2:



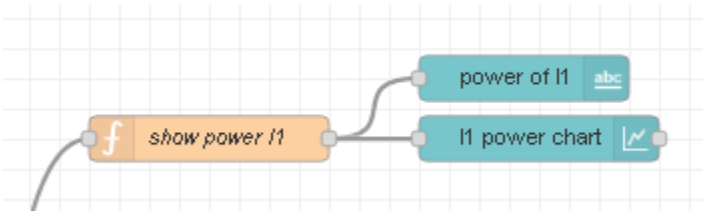
Part 2.a





smartmeter node is basically **mqtt in** node that takes data from **mqtt out** and inputs it for further processing.

Part 2.b

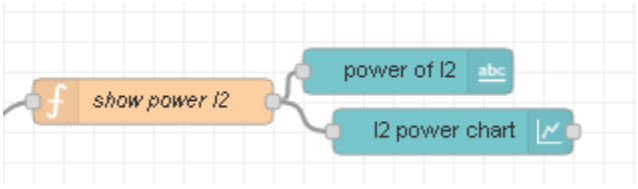


This part contains power calculation per hour (assumed 1 hour = 1 second) (**show power l1**) of device “Light 1”, graph node (**L1 Power Chart**) and text node (**Power**) for node red dashboard and thingspeak node (**l1**) to forward data (calculated power) to thingspeak dashboard.

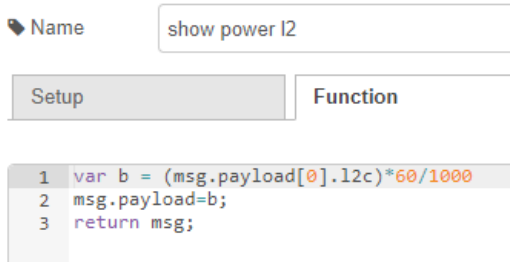
show power l1 contains following code.

```
var a = (msg.payload[0].l1c)*60/1000

msg.payload=a

return msg;
```

Part 2.c



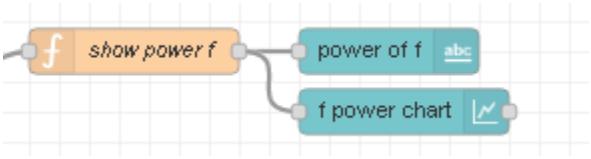
This part contains power calculation per hour (assumed 1 hour = 1 second) (**show power l2**) of device “Light 2”, graph node (**L2 Power Chart**) and text node (**Power**) for node red dashboard and thingspeak node (**l2**) to forward data (calculated power) to thingspeak dashboard.



show power l2 contains following code.

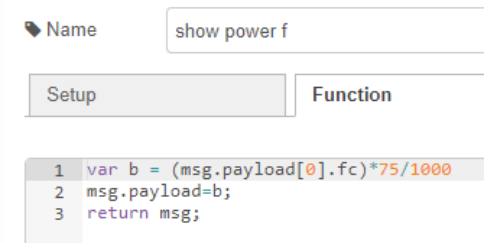
```
var b = (msg.payload[0].l2c)*60/1000
msg.payload=b;
return msg;
```

Part 2.d



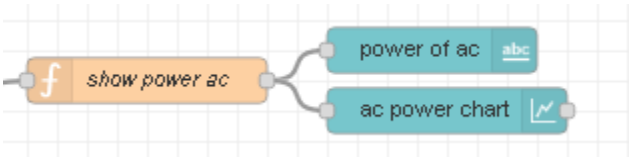
This part contains power calculation per hour (assumed 1 hour = 1 second) (**show power f**) of device “Fan”, graph node (**Fan Power Chart**) and text node (**Power**) for node red dashboard and thingspeak node (**f**) to forward data (calculated power) to thingspeak dashboard.

show power f contains following code.




```
var b = (msg.payload[0].fc)*75/1000
msg.payload=b;
return msg;
```

part 2.e

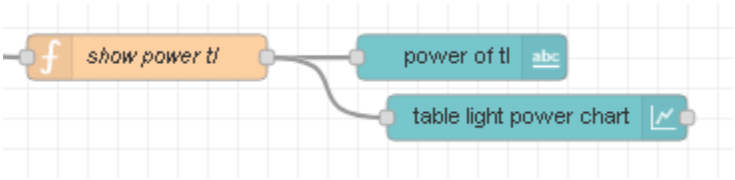


This part contains power calculation per hour (assumed 1 hour = 1 second) (**show power ac**) of device “Air Conditioner”, graph node (**AC Power Chart**) and text node (**Power**) for node red dashboard and thingspeak node (**ac**) to forward data (calculated power) to thingspeak dashboard.

show power ac contains following code.

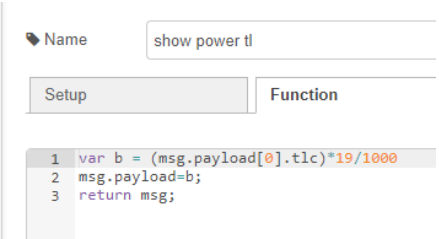

```
var b = (msg.payload[0].acc)*945/1000

msg.payload=b;

return msg;
```

part 2.f



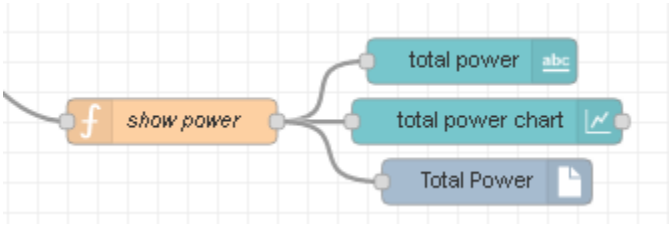
This part contains power calculation per hour (assumed 1 hour = 1 second) (**show power tl**) of device “Table Lamp”, graph node (**Table Lamp PC**) and text node (**Power**) for node red dashboard and thingspeak node (**tl**) to forward data (calculated power) to thingspeak dashboard.

show power tl contains following code.




```
var b = (msg.payload[0].tlc)*19/1000

msg.payload=b;

return msg;
```

part 2.g

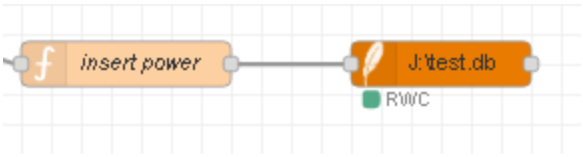


This part contains total power calculation (**show power**) of all devices, graph node (**Total Power Chart**) and text node (**Power**) for node red dashboard and thingspeak node (**Power**) to forward data (calculated total power) to thingspeak dashboard.

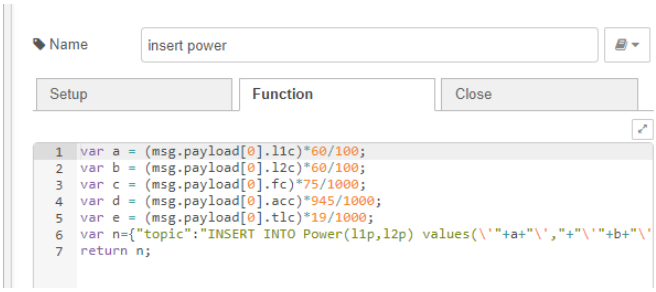
show power contains following code.

```
var a = (msg.payload[0].l1c)*60/1000;

var b = (msg.payload[0].l2c)*60/1000;

var c = (msg.payload[0].fc)*75/1000;

var d = (msg.payload[0].acc)*945/1000;

var e = (msg.payload[0].tlc)*19/1000;

var pf=a+b+c+d+e;

msg.payload=pf;

return msg;
```

part 2.h



Insert power calculates total power (same as **show power**) but also inserts the power in power table.



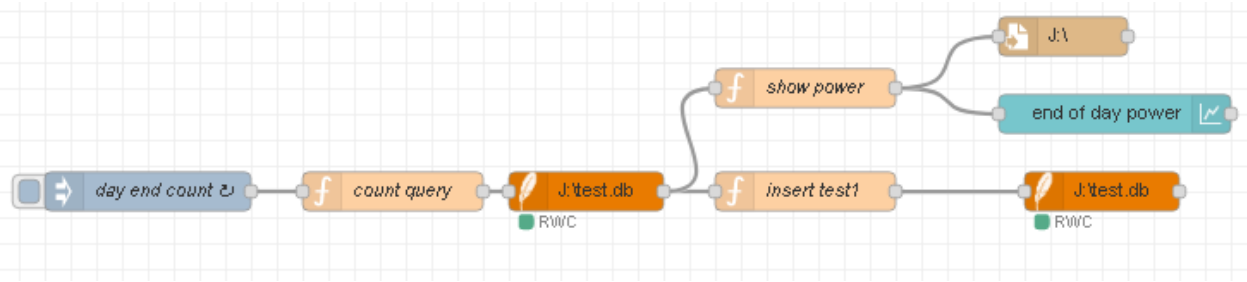
It contains following code.

```
var a = (msg.payload[0].l1c)*60/100;

var b = (msg.payload[0].l2c)*60/100;
```

```
var c = (msg.payload[0].fc)*75/1000;

var d = (msg.payload[0].acc)*945/1000;

var e = (msg.payload[0].tlc)*19/1000;

var n={"topic":"INSERT INTO Power(l1p,l2p)
values(\'"+a+"\',"+"\'"+b+"\',"+"\'"+c+"\',"+"\'"+d+"\',"+"\'"+e+"\')"};

return n;
```

The node next to it is **sqlite** node which connects to the sqlite database and tables inside it.

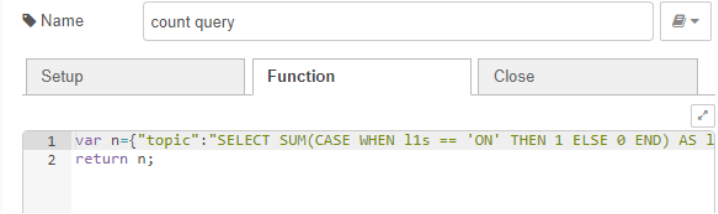
Part 3



Now,

day end count is just a repeater that repeats itself every 24 seconds (since 1 day = 24 seconds) and triggers the count query every time it is triggered.

count query is a node that calculates count of status of all devices in order to calculate power.

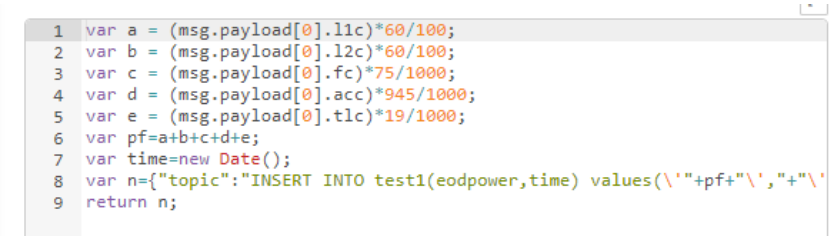


It contains following code.

```
var n={"topic":"SELECT SUM(CASE WHEN l1s == 'ON' THEN 1 ELSE 0 END) AS l1c,SUM(CASE
WHEN l2s == 'ON' THEN 1 ELSE 0 END) AS l2c ,SUM(CASE WHEN fs == 'ON' THEN 1 ELSE 0 END)
AS fc ,SUM(CASE WHEN acs == 'ON' THEN 1 ELSE 0 END) AS acc ,SUM(CASE WHEN tls == 'ON'
THEN 1 ELSE 0 END) AS tlc FROM Status"};
return n;
```

The node next to **count query** node is a **sqlite** node that connects the sqlite database and tables inside it.

insert test1 node calculates power after every day (24 seconds) and inserts the calculated power inside test1 table. It contains the following code.



```
var a = (msg.payload[0].l1c)*60/100;
var b = (msg.payload[0].l2c)*60/100;
var c = (msg.payload[0].fc)*75/1000;
var d = (msg.payload[0].acc)*945/1000;
var e = (msg.payload[0].tlc)*19/1000;
var pf=a+b+c+d+e;
var time=new Date();
var n={"topic":"INSERT INTO test1(eodpower,time) values(\'"+pf+"\',"+"\'
return n;
```


var a = (msg.payload[0].l1c)*60/100;

var b = (msg.payload[0].l2c)*60/100;

var c = (msg.payload[0].fc)*75/1000;

var d = (msg.payload[0].acc)*945/1000;

var e = (msg.payload[0].tlc)*19/1000;

var pf=a+b+c+d+e;

var time=new Date();

var n={"topic":"INSERT INTO test1(eodpower,time) values(\'"+pf+"\',"+"\'"+time+"\')"};

return n;

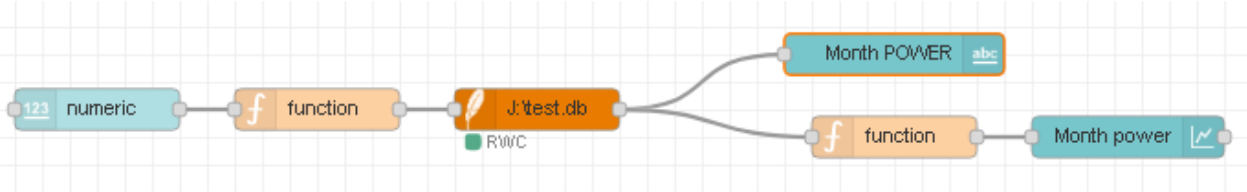
show power node is similar to **insert powerday** node but it just calculates power, does not inserts it anywhere. It contains the following json code.

```
var a = (msg.payload[0].l1c)*60/1000;
var b = (msg.payload[0].l2c)*60/1000;
var c = (msg.payload[0].fc)*75/1000;
var d = (msg.payload[0].acc)*945/1000;
var e = (msg.payload[0].tlc)*19/1000;
var pf=a+b+c+d+e;
```

```
msg.payload=pf;
return msg;
```

end of day power node is a dashboard node that shows the end of day power on a chart.

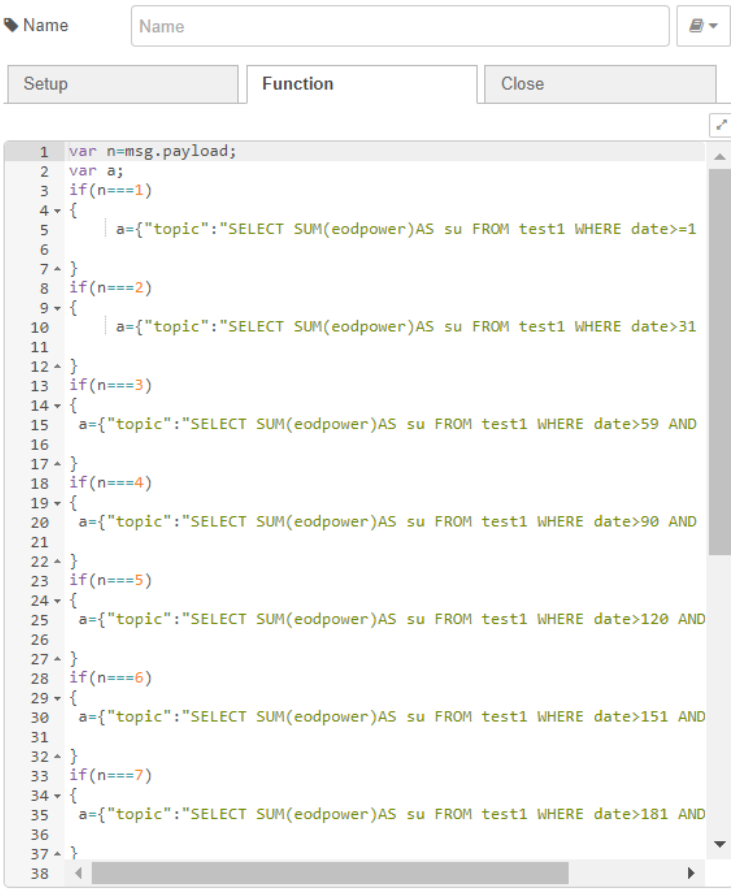
Part 4:



This takes the numeric input from the dashboard by the user.

It displays the total power consumed at the end of the month Graphically so it's easier for analysis.

The first function node contains queries for getting the Total power of the month which is inputted from the Numeric input



```
var n=msg.payload;
var a;
if(n===1)
{    a={"topic":"SELECT SUM(eodpower)AS su FROM test1 WHERE date>=1 AND date<=31"};}
if(n===2)
{    a={"topic":"SELECT SUM(eodpower)AS su FROM test1 WHERE date>31 AND date<=59"};}
if(n===3)
{ a={"topic":"SELECT SUM(eodpower)AS su FROM test1 WHERE date>59 AND date<=90"};}
if(n===4)
{ a={"topic":"SELECT SUM(eodpower)AS su FROM test1 WHERE date>90 AND date<=120"};}
if(n===5){
 a={"topic":"SELECT SUM(eodpower)AS su FROM test1 WHERE date>120 AND date<=151"};}
if(n===6){
 a={"topic":"SELECT SUM(eodpower)AS su FROM test1 WHERE date>151 AND date<=181"};}
if(n===7)
{ a={"topic":"SELECT SUM(eodpower)AS su FROM test1 WHERE date>181 AND date<=212"};}
if(n===8)
{a={"topic":"SELECT SUM(eodpower)AS su FROM test1 WHERE date>212 AND date<=243"};}
if(n===9)
{ a={"topic":"SELECT SUM(eodpower)AS su FROM test1 WHERE date>243 AND date<=273"};}
if(n===10)
{ a={"topic":"SELECT SUM(eodpower)AS su FROM test1 WHERE date>273 AND date<=304"};}
if(n===11)
```

{ a={"topic":"SELECT SUM(eodpower)AS su FROM test1 WHERE date>304 AND date<=334"};}

if(n===12)

{ a={"topic":"SELECT SUM(eodpower)AS su FROM test1 WHERE date>=334 AND date<=365"};}

return a;

The next function node gets the final sum of power of the particular and returns it.

```
1
2  var a = (msg.payload[0].su);
3  msg.payload=a;
4  return msg;
```

var a = (msg.payload[0].su);

msg.payload=a;

return msg;

CSV:

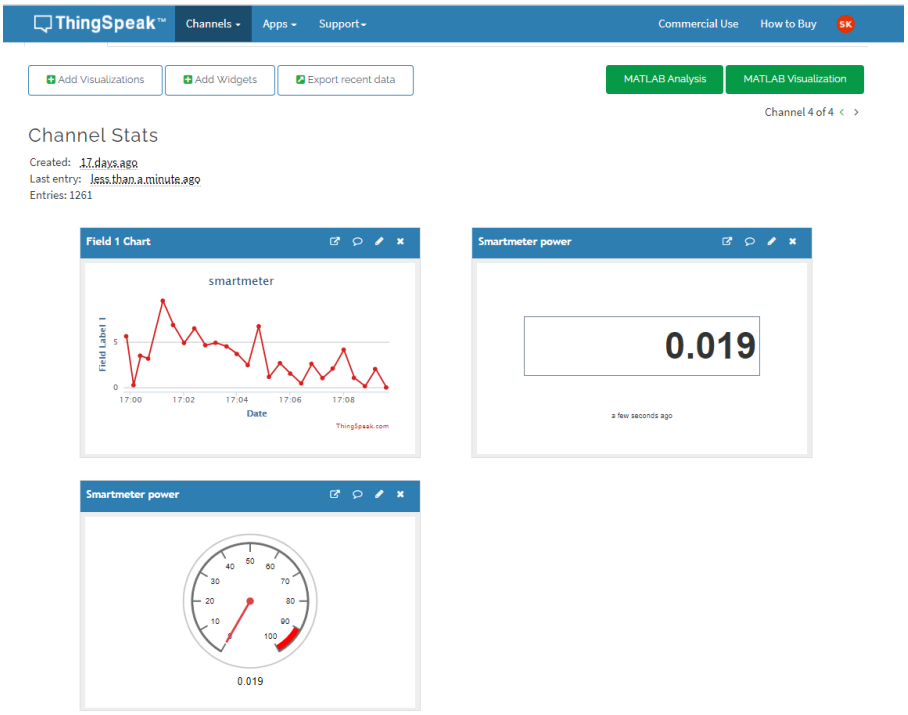
The EOD Power data is sent to CSV File for Future Reference.

	A
1	17.475
2	12.614
3	12.28
4	11.376
5	8.473
6	9.208
7	7.502
8	24.438
9	10.476
10	9.081
11	9.534
12	9.699
13	10.457
14	6.778
15	13.723
16	11.919
17	6.546

FINAL OUTPUT(RESULT):

Dashboard:

Thingspeak:



Conclusion :

As we can see I have created a **Home automation system simulation with smart-meter** using node red thus the project is complete.

Future Work :

We can study this simulation and if it is realistic, then we can implement it in the real life thus making life simple.

References :

https://cookbook.nodered.org/mqtt/connect-to-broker

https://nodered.org/docs/

http://noderedguide.com/tutorial-sqlite-and-node-red/