

**SINOTIMER®**



# SMART ENERGY METER USING WI-FI

G-30

## **G-30**

- 180622H –SUDASINGHE S.A.T.N
- 180644C- H.G.I THIRANGA
- 180661B-VIBHUSHA K.K.L

## TABLE OF CONTENTS

Introduction .....	2
Methodology and Scope .....	3
Final Outcome .....	3
Procedure .....	4
Calibration process.....	8
Full Schematic .....	9
Website Design .....	13
Practical Implementation.....	16
Arduino And NodeMcu Code Explained .....	17
Future Work .....	21
APPENDIX -Website Code .....	22
References .....	24

## Introduction

At present, electricity is an indispensable thing and very important in everywhere. As a result of it, energy saving has been become a necessity. Power theft is a one big problem in these days which reasons lot of losses to electricity boards. The habit of stealing electricity power is known as power theft. This can be prevented by using smart energy meters. In addition to that, there is a conventional billing method in Sri Lanka. There are meter readers who visit each home and note down the meter readings manually. After that, the billing criteria is applied using meter readings. As results of those, human observational errors can be happened. And also, when the consumers are not available in the houses, the meter reader has to come again. To reduce those errors and the cost for the labors, and to save the time for visiting each houses, it will be much better if the utilities have a system with remote access to energy consumption of each houses. As a help for this also, Smart energy meter can be used.

Smart energy meter is an electric device which has the energy meter chip to measure the consumed electric energy and a wireless system for data communication. Nowadays, there is a lot of focus on wireless communication and reducing manual works for each industries. So, in order to save electricity, labor cost and time, smart energy meter can be used. By using this, utilities will get the ability to monitor power consumption in each houses regularly without visiting each houses. These factors motivated us to do this project.

Our main objective was proposing a method for replacing traditional energy meter reading methods and giving remote access of household energy meters to energy providers. In order to achieve that, we wanted to measure the current, voltage and power factor of an AC electric energy consumption system and show values of power and energy continuously using a website. In addition to that, we had a sub objective to form a monthly billing system using the website, although we didn't have enough time to complete it.

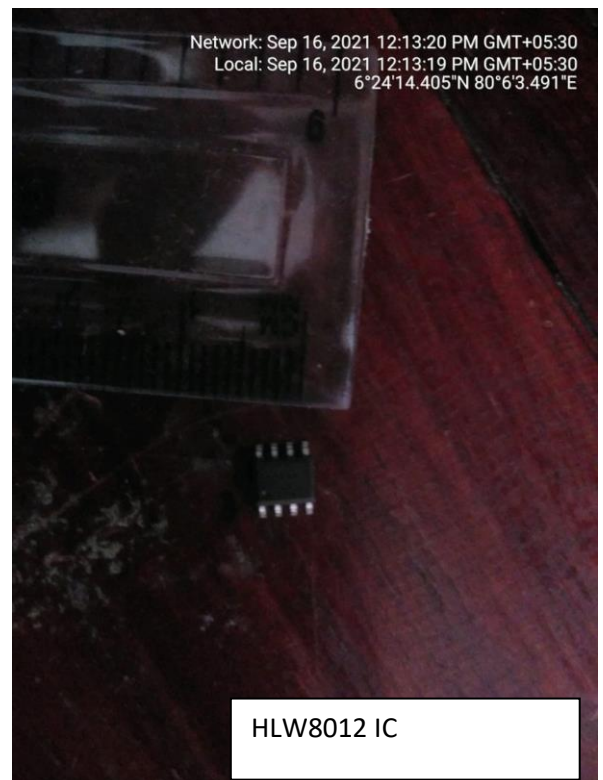
## **Project Scope**

The scope of this project is to measure the current, voltage and power factor of an AC electric energy consumption system and show values of power and energy using a website.

This project proposes a method for replacing traditional energy meter reading methods and giving remote access of household energy meters to energy providers.

## **Methodology**

- Voltage, Current, power factor , energy & power can be measured by energy metering IC.(Which is a signal processing IC)
- For example –V9881D or HLW8012



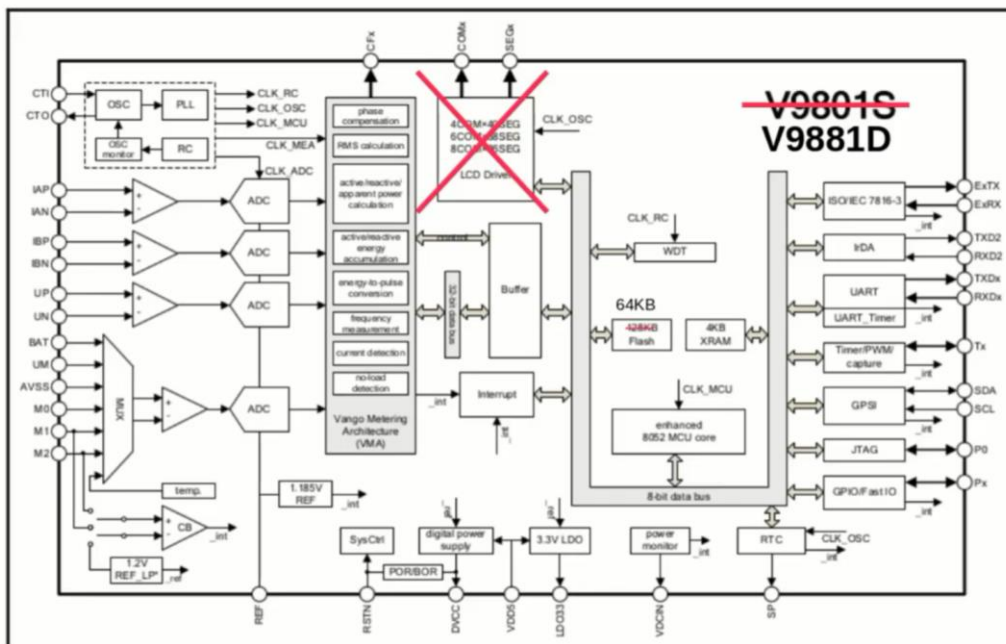
- Output of this IC is fed to Arduino board by serial communication (UART transmission protocol.)
- Then the Arduino board processes those data and by another serial communication it is given to a nodeMcu board.
- By wi-fi facility of NodeMcu board the data is transmitted to a database(Firebase) .
- In this system, the utilized energy and the active power will be displayed on the website, continuously.

## **Final Outcome of the Project**

Smart energy meter is an electric device which has the energy meter chip to measure the consumed electric energy and a wireless system for data communication.

- Final outcome is an energy meter to measure household energy consumption with remote access and get monthly bill by usage.

## Procedure



- The V9881D IC is the main IC which is used to find values (voltage , current power etc.)

## **Why We can't directly use Arduino to measure voltage, current, power factor?**

- The main reason is a general Arduino board has 400Hz sampling rate
- So we have to measure 50Hz signal.
- At least to get 4 times 50Hz sampling rate to get non aliasing signals  $50 \times 4 \times 2$  (voltage, current). With code execution time this rate can't be achieved.
- So the sampling with Arduino won't work.
- That is why we need energy metering IC to process signals.
- We have to low pass filter the Input current before we feed it to the circuit. Due to higher noise generated by magnetic field of the CT sensor.

## Prototype Module-PZEM 004T module

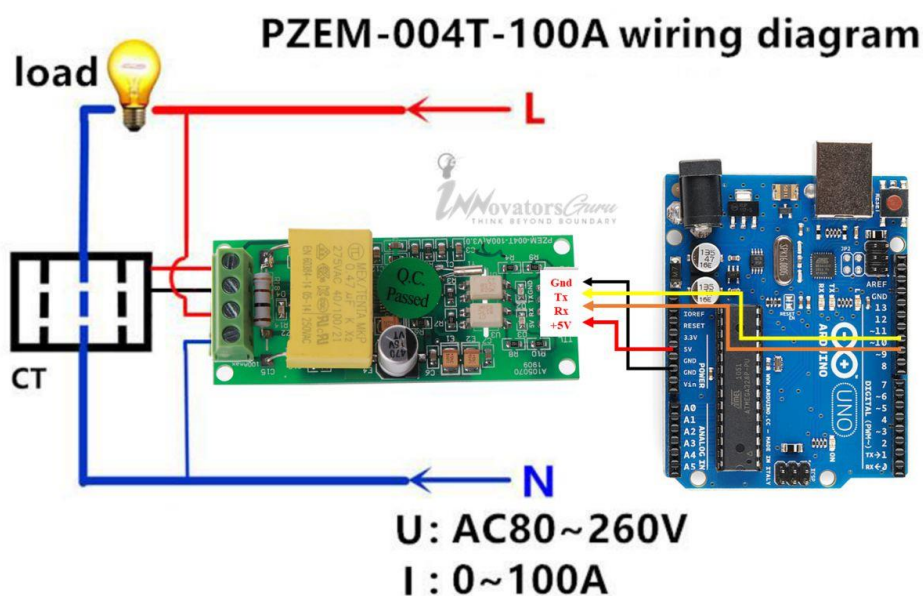
V9881D IC is a DIP24 IC. So we can't use this IC to prototype because it is a SMD IC.

We came up with a schematic wo modify this module.

We used a CT sensor to get the current inside the module. It is advantageous to use one due to galvanic isolation of the circuit from the higher voltage supply.

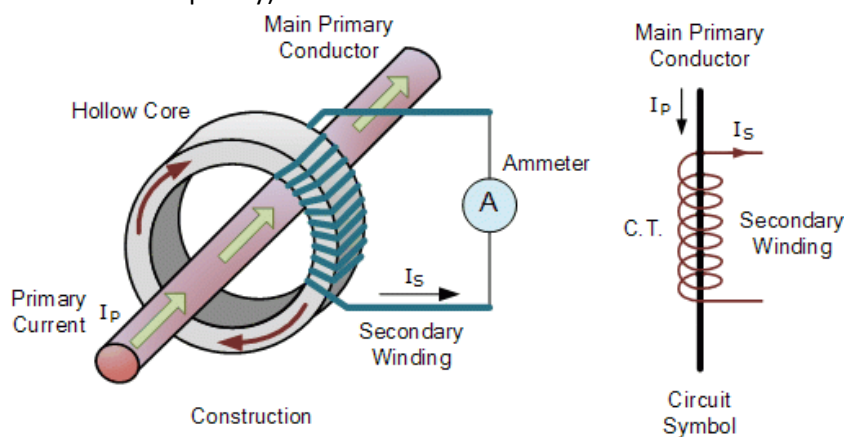
### The basic principle of the current sensor.

The induced current(or voltage) in the secondary winding is proportional to the primary winding current.



$$I_{secondary} \propto I_{primary}$$

Assuming the frequency of the sine waveform remains constant.(Due to less variance of frequency)



## 10. Energy Metering

The energy metering architecture in V98XX has features:

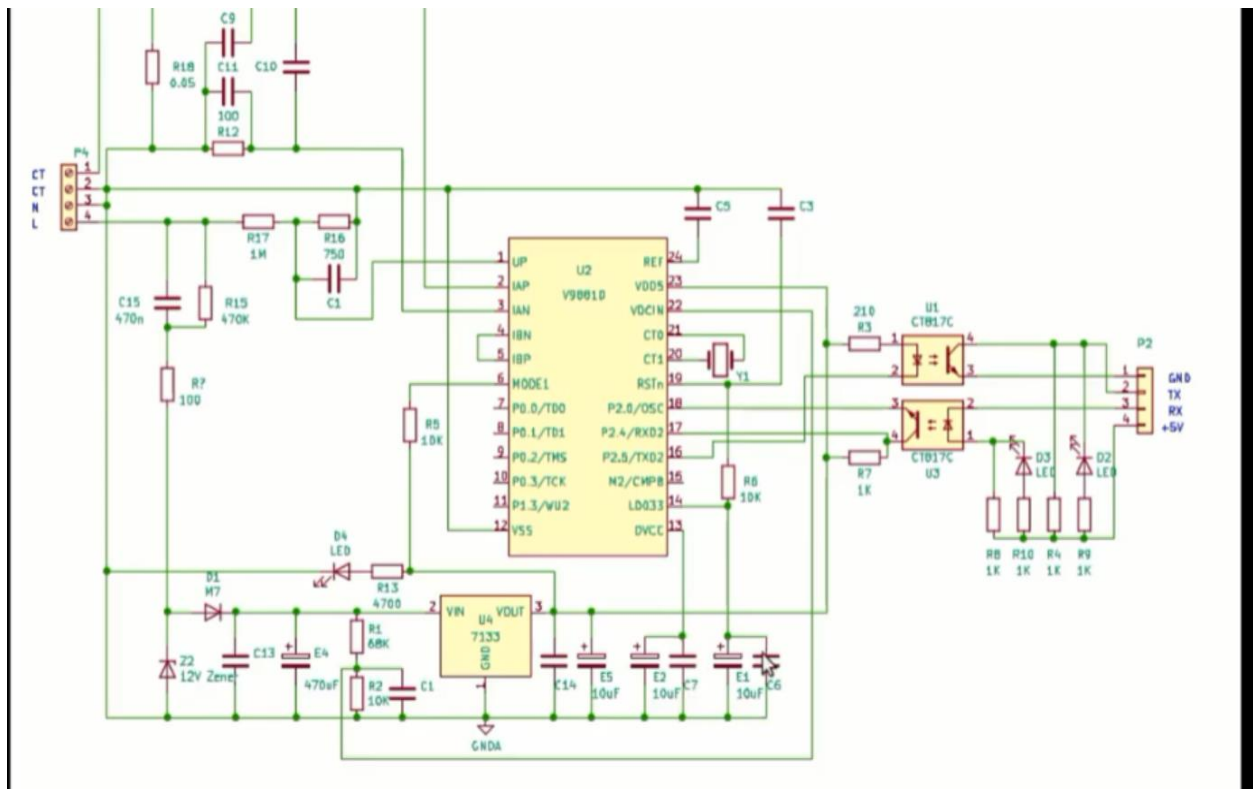
- Four independent oversampling  $\Sigma/\Delta$  ADCs: One voltage channel (U), two current channels (I), and one multifunctional channel for various signal measurements.
- High metering accuracy:
  - Less than 0.1% error on active energy metering over dynamic range of 5000:1.
  - Less than 0.1% error on reactive energy metering over dynamic range of 3000:1.
  - Less than 0.5% error on current and voltage RMS calculation over dynamic range of 1000:1.
- Providing measurements:
  - Raw waveform and DC component of current/voltage signals
  - Instantaneous/Average and active/reactive power
  - Positive/Negative and active/reactive energy
  - Average apparent power
  - Instantaneous/average current/voltage RMS
  - Line frequency

## Module Schematic that is given on the datasheet

### Specifications of the PZEM 004T module

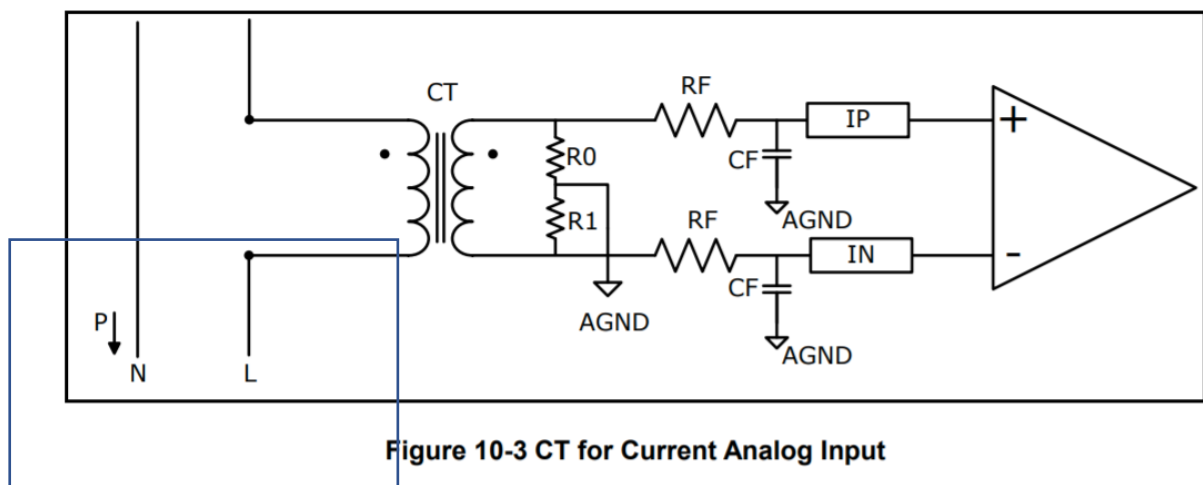
Specifications	Descriptions
Working voltage	80 - 260VAC
Current measurement	0 – 100 A
Rated power	22kW
Operating frequency	45-65Hz

According to datasheet the IC can measure up to 100A current when the CT sensor is used.



- According to this circuit diagram the voltage and current is fed to the IC by considering conditions that are mentioned in the datasheet.

### V9801S/V9811S/V9811A/V9811B/V9821/V9821S/V9881D Datasheet



In this diagram of the datasheet it is shown to filter out the current waveform before it is fed to the IC. (lowpass filtering)

- The voltage of the line is given to the IC by using a voltage divider system ( $750\Omega/1M\Omega$ )
- The conversion of 240 V to 5v is done by 12v Zener diode , general purpose diode and a LM7175 (5v regulator).
- This method is very inefficient due to non-usage of negative cycle due to diode and energy is somewhat wasted.

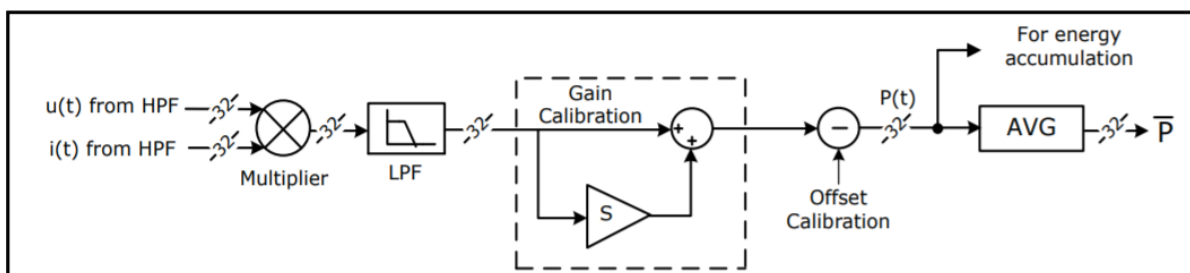


- Also there is no protection circuitry to protect circuit from overvoltage.
- So we came up with our own schematic to implement those functionalities.

## Calibration process

The datasheet specifies that if we use standard resistors & capacitors to create the circuit we can get the voltage & current to 0.5% accuracy and energy up to 0.1% accuracy. Further calibration can be done inside the IC by reprogramming it and calibrating it.

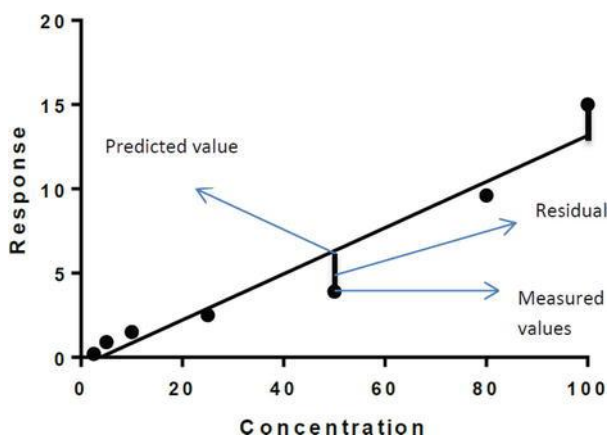
### 10.12.1. Active Power Calculation and Calibration



If we take a

look at the diagram up here in datasheet,

By changing the **gain calibration coefficients** & **offset calibration coefficients** we can adjust the output value to be more accurate.

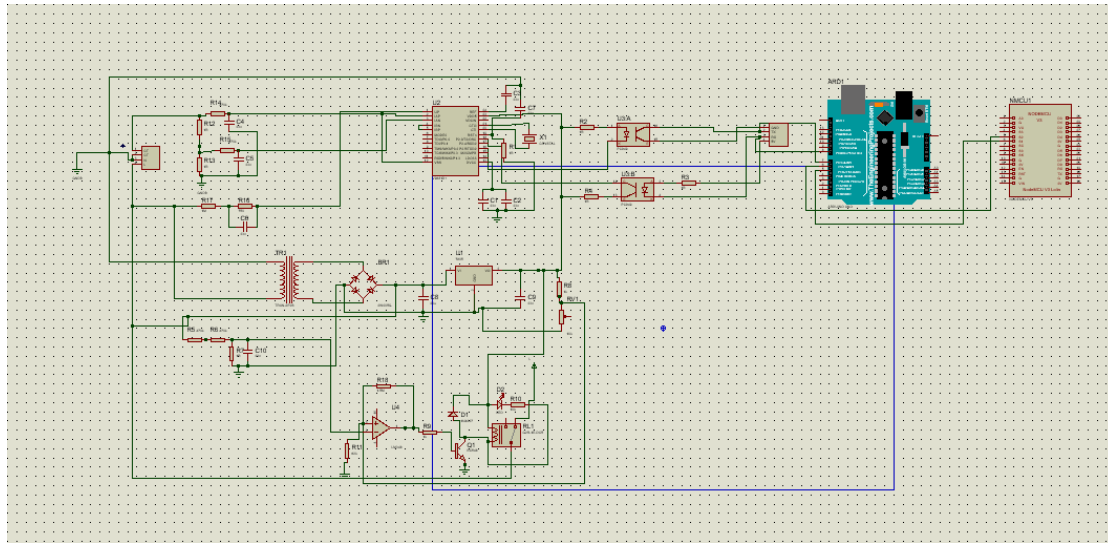


Calibration curve can be drawn using a standard load and a accurate power meter and by tweaking parameters of gain calibration and offset calibration.

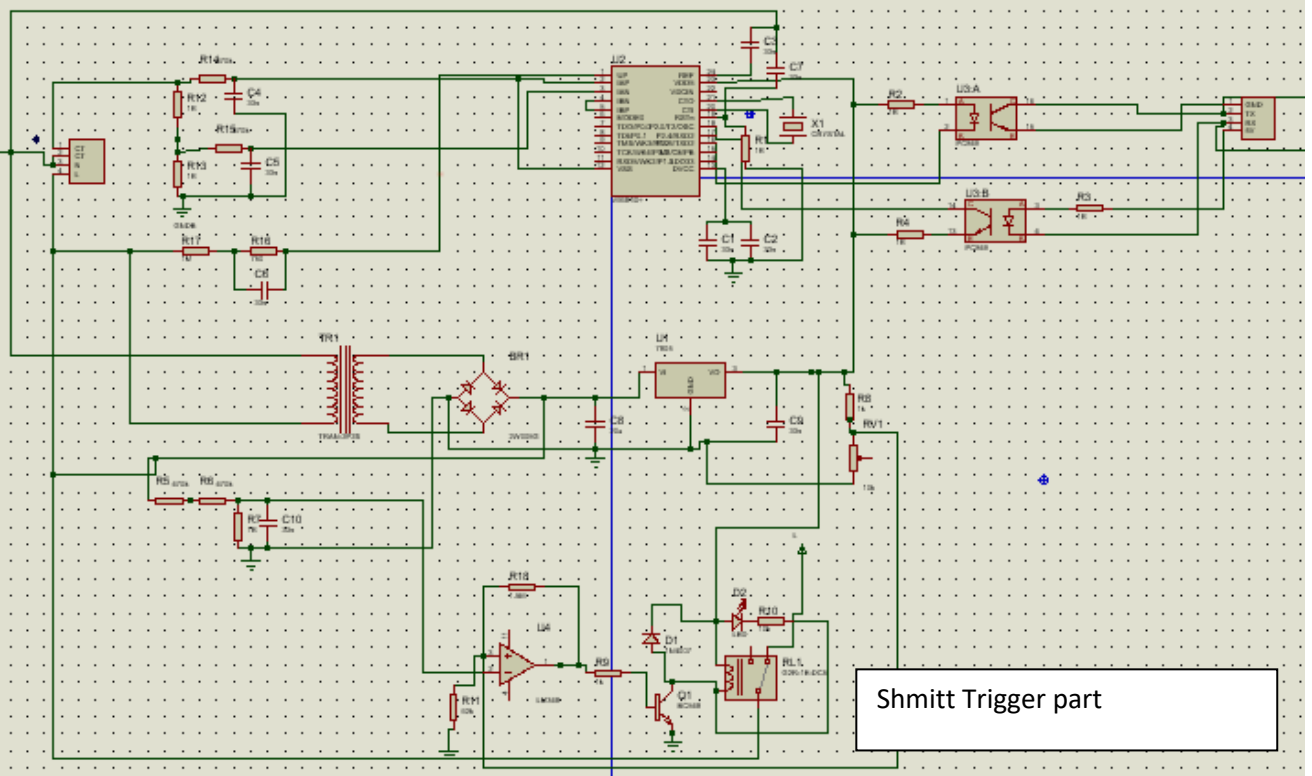
### Circuit Schematic Implementation

We did a proteus 8 schematic (but didn't do a simulation due to unavailability of the IC in proteus libraries).

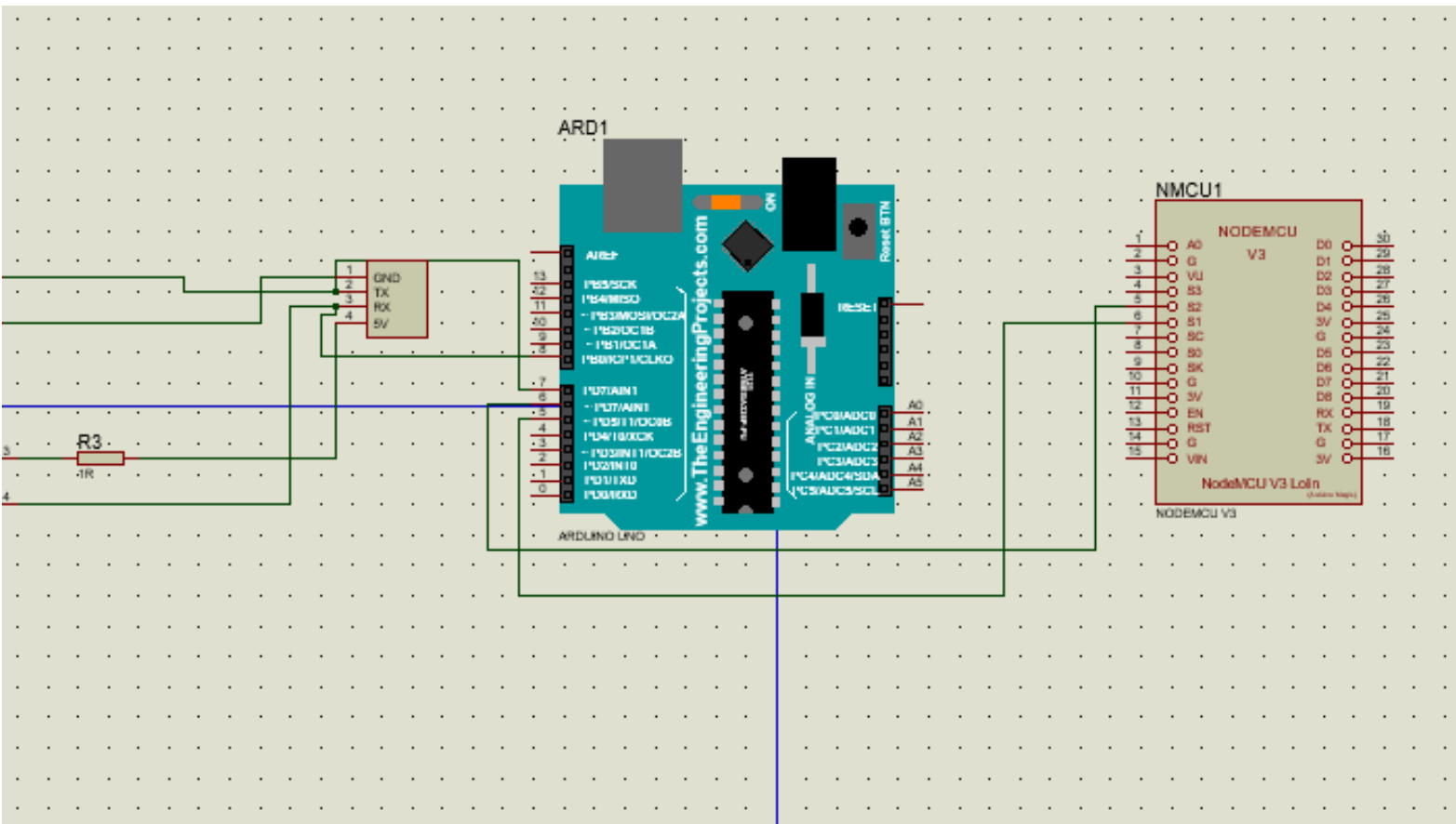
## Full Schematic



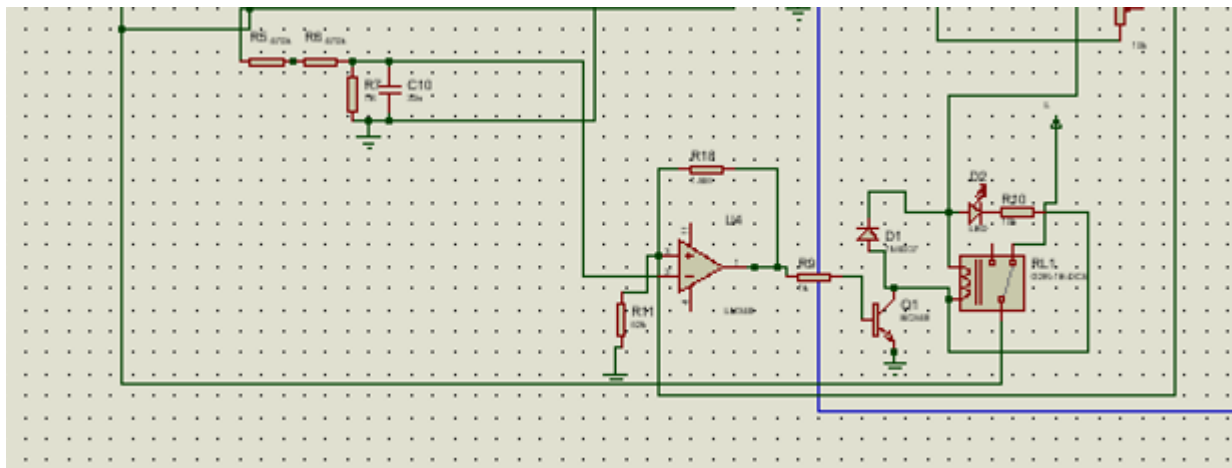
Zoom in 1



Shmitt Trigger part



The Arduino board process the gotten data from serial communication and it is fed to NodeMcu board by serial communication.



In here what we do is we create a voltage divider to divide up our voltage that we can handle. We use the op amp in the positive feedback(“Schmitt Trigger”). By changing the variable resistor we can change the schmitt trigger threshold value . If there is an overvoltage in the circuit the schmitt trigger will output 5V and the BJT will be forward biased. the RED LED will light on and the relay will disconnect the circuit from the grid. So the electronics will be safe from overvoltage.

Also we can send a signal to microcontroller alerting of the incident.

To efficiently convert AC to DC(5v) we used a step down transformer and a rectifier and by using voltage regulator (LM7135 ) IC and by using capacitors we smoothed out the DC input voltage. The step down transformer we used was maximum 1A output step down transformer.(1 A is more than enough current to run the electronics of the circuit)

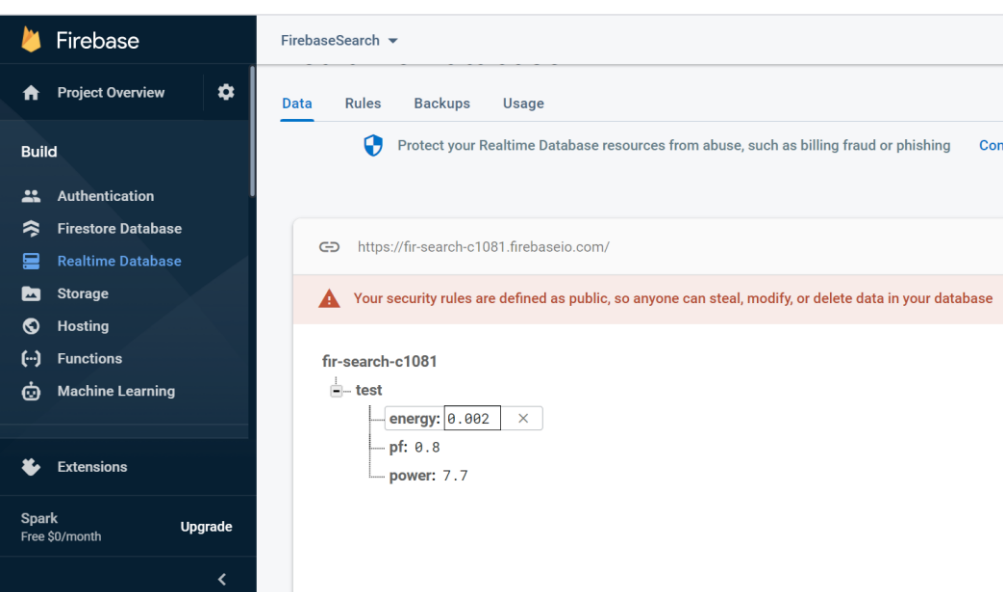
## Why did we use optocouplers for Tx Rx communication?

If somehow there was a voltage surge which happened to effect left side of the circuit due to electronic isolation provided form optocouplers the Arduino board and the NodeMcu board will be safe.

## Serial communication between Arduino and the NodeMcu

The values gotten from the V9881D IC(By UART transmission) are being processed by the Arduino board and those are again fed to NodeMcu board(by UART comm) to send to firebase database.

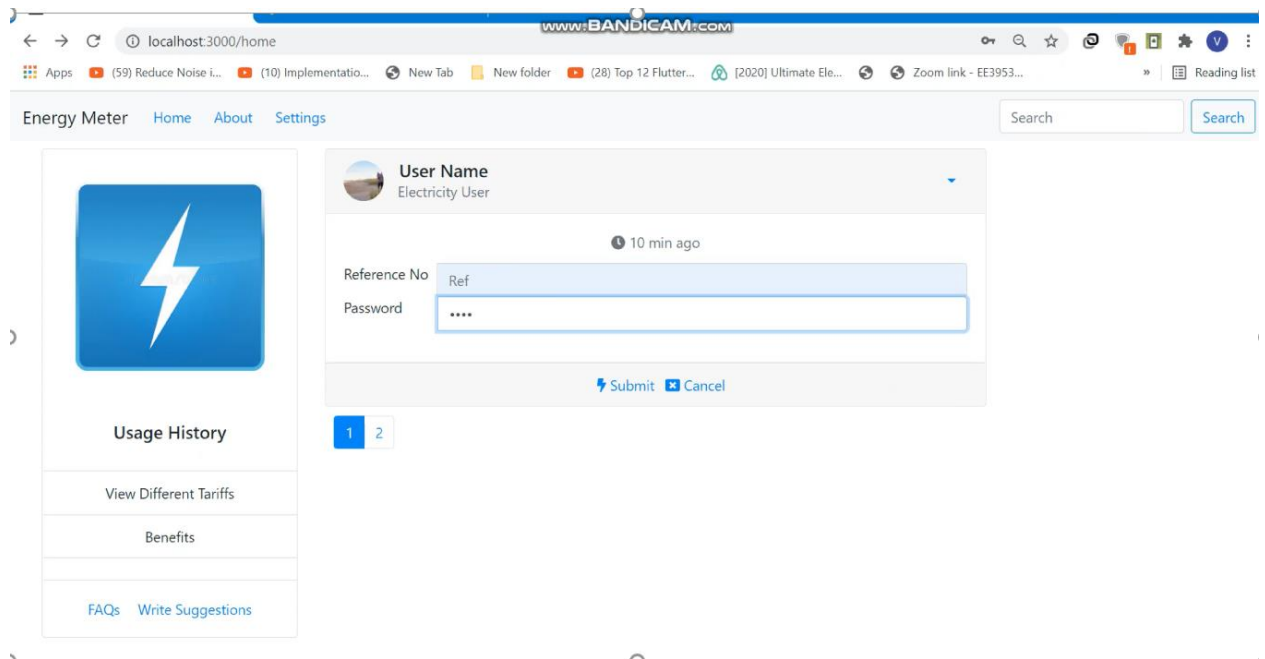
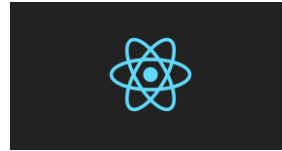
The advantage of this method is somehow wi-fi is not connected the Arduino still runs and the code won't stop and computational power would also be higher due to 'parallel threading'.



- The firebase database is updated by data fed from NodeMCU board.
- Firebase is a backend software which is created by google.
- It is easy to integrate to a website so we used that to implement our backend of our software.

# Website Design

The website was created by us using REACT libraries.



- We planned on a system which users can view their bill using their Reference No and password.
- Also users could change their tariffs online and view details of electricity service.
- Also we can further implement this system to get a power factor of a certain region and by getting those data we can identify the regions we can improve power factor of the system.


Energy Meter

[Home](#)

[About](#)

[Settings](#)

Search



Current Usage

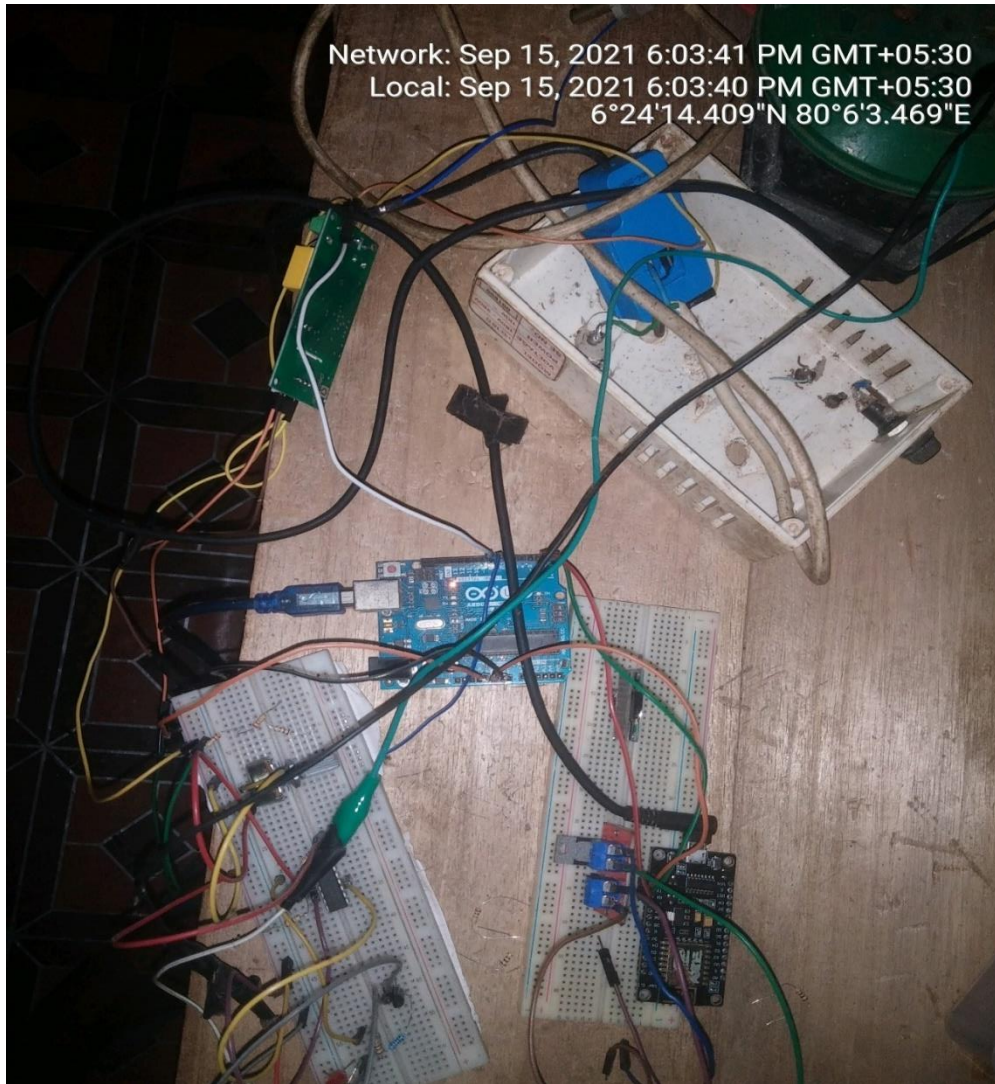
Active Power-7.7 W

Energy - 0.02 kWh

power factor-0.8

In here we get the current usage from the firebase database in real-time.

## Practical Implementation of the project



In here we have done a practical implementation of the project. As it is shown we got values from the prototype module and fed it to the Arduino. And we set up a CFL bulb to measure power output.





## Coding the Arduino Board Explained

After measuring the power, energy, voltage, current and the power factor is measured using the energy metering IC in the PZEM004Tv30 board the data is transmitted through an optocoupler to the Arduino board through serial UART transmission protocol. Using the Arduino, we receive the data and process it before sending it to the NodeMcu board. The Arduino code for this process is attached below.

1. First we import the required Libraries.

```
#include <WiFi.h>

#include <WiFiClient.h>

#include <WiFiServer.h>

#include <WiFiUdp.h>

#include <SoftwareSerial.h>

#include <ArduinoJson.h>

#include <PZEM004Tv30.h>
```

2. Then we initialize the Arduino to nodeMcu serial communication.

```
//Initialise Arduino to NodeMCU (5=Rx & 6=Tx)

SoftwareSerial nodemcu(5, 6);
```

3. Then we define the required variables

```
float energy;
float power;
Float pf;
```

4. Define the transmission and receiving pins of the PZEM004Tv30 Module.

```
PZEM004Tv30 pzem(7, 8);
```

5. Code the void setup of the Arduino board to start serial communications with the computer and the nodeMcu board

```
void setup() {

  Serial.begin(9600);

  nodemcu.begin(9600);


  delay(1000);

}
```

6. Code the Arduino void loop to read the power, energy and power factor every second and send them to the nodeMcu board using a Json file.

```
void loop() {
```

```

StaticJsonBuffer<1000> jsonBuffer;

JsonObject& data = jsonBuffer.createObject();

float power = pzem.power();

if( !isnan(power) ){

data["power"] = power;

} else {

Serial.println("Error reading power");

}

float energy = pzem.energy();

if( !isnan(energy) ){

data["energy"] = energy;

} else {

Serial.println("Error reading energy");

}

float pf = pzem.pf();

if( !isnan(pf) ){

data["pf"] = pf;

} else {

Serial.println("Error reading power factor");

}

data.printTo(nodemcu);

jsonBuffer.clear();

Serial.println();

delay(1000);

}

```

## Coding the NodeMcu Board

After the data are fetched to the nodeMcu board from Arduino, we need to send the received data to the online database “Firebase” using the Wi-Fi facility in the nodeMcu board. For this task the below code is used.

1. First we define the nodeMcu board version, include firebase libraries, initialize serial communications with the Arduino board and configure the wifi.

```

#if defined(ESP32)
#include <WiFi.h>
#include <FirebaseESP32.h>
#elif defined(ESP8266)
#include <ESP8266WiFi.h>
#include <FirebaseESP8266.h>
#include <SoftwareSerial.h>
#include <ArduinoJson.h>
#endif

#include "addons/TokenHelper.h"
#include "addons/RTDBHelper.h"

SoftwareSerial nodemcu(D6, D5);

#define WIFI_SSID "HUAWEI-4388"
#define WIFI_PASSWORD ""
#define API_KEY "AlzaSyADDi1-A3jZdMU0GjCS6nPsE-vn76eXYVA"

```

## 2. Define the RTDB URL

```

#define DATABASE_URL "fir-search-c1081.firebaseio.com"

```

## 3. Define the user Email and password that is already registered

```

#define USER_EMAIL vibhooshakannangara@gmail.com
#define USER_PASSWORD

//Define Firebase Data object

```

## 4. Define variables

```

FirebaseData fbdo;

FirebaseAuth auth;

FirebaseConfig config;

unsigned long sendDataPrevMillis = 0;

float power;

float energy;

Float pf;

```

5. Code for void setup to start communications with the Arduino and the computer and configurate firebase

```

void setup(){
  Serial.begin(9600);
  nodemcu.begin(9600);
  while (!Serial) continue;    <---- while not logged in
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(300);
  }
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();
  Serial.printf("Firebase Client v%s\n\n", FIREBASE_CLIENT_VERSION);
  config.api_key = API_KEY;
  auth.user.email = USER_EMAIL;
  auth.user.password = USER_PASSWORD;
  config.database_url = DATABASE_URL;
  config.token_status_callback = tokenStatusCallback;
  Firebase.begin(&config, &auth);
  Firebase.reconnectWiFi(true);
  Firebase.setDoubleDigits(5);

```

6. Coding void loop to extract data from the json file and feed it to firebase database.

```

`void loop(){
  StaticJsonBuffer<1000> jsonBuffer;
  JsonObject& data = jsonBuffer.parseObject(nodemcu);
  if (data == JsonObject::invalid()) {
    Serial.println("Invalid Json Object");
    jsonBuffer.clear();
    return;
  }
  power = data["power"];

```

```

pf = data["pf"];

energy = data["energy"];

if (Firebase.ready() && (millis() - sendDataPrevMillis > 15000 || sendDataPrevMillis == 0)) {

sendDataPrevMillis = millis();      <--- update database time difference

int iVal = 0;

    Serial.printf("Get int ref... %s\n", Firebase.getInt(fbdo, "/test/int", &iVal) ? String(iVal).c_str() :
fbdo.errorReason().c_str());

    Serial.printf("Set float... %s\n", Firebase.setFloat(fbdo, "/test/power", power) ? "ok" :
fbdo.errorReason().c_str()); <---- set(update) power value in firebase

    Serial.printf("Get float... %s\n", Firebase.getFloat(fbdo, "/test/power") ?
String(fbdo.to<float>()).c_str() : fbdo.errorReason().c_str());

    Serial.printf("Set float... %s\n", Firebase.setFloat(fbdo, "/test/pf", pf) ? "ok" :
fbdo.errorReason().c_str()); <---- set(update) power factor value in firebase

    Serial.printf("Get float... %s\n", Firebase.getFloat(fbdo, "/test/pf") ? String(fbdo.to<float>()).c_str()
: fbdo.errorReason().c_str());

    Serial.printf("Set float... %s\n", Firebase.setFloat(fbdo, "/test/energy", energy) ? "ok" :
fbdo.errorReason().c_str()); <---- set(update) energy value in firebase

    Serial.printf("Get float... %s\n", Firebase.getFloat(fbdo, "/test/energy") ?
String(fbdo.to<float>()).c_str() : fbdo.errorReason().c_str());

    Serial.println();

    }

}

```

## Future Work

1. Apply an OLED display to show the energy meter values.

Currently we didn't implement this due to delay occurred due to time taken for Arduino to print to OLED display.

With clever coding we can minimize the delay.

2. Add a battery power to project

We got a WeMos battery shield which can charge the battery as well as protect it from overvoltage. We didn't have a LiPo battery to practically implement this part of project.



the



## ***Appendix:Website Design Code(firebase.js)***

(We only added two components of the code because full code consists of 16 components)

```
import firebase from 'firebase/compat/app';
import 'firebase/compat/auth';
import 'firebase/compat/firestore';
import { initializeApp } from "firebase/app";
import { getMessaging, getToken } from "firebase/messaging";

const firebaseConfig = {
  apiKey: "AlzaSyADDi1-A3jZdMU0GjCS6nPsE-vn76eXYVA",
  authDomain: "fir-search-c1081.firebaseio.com",
  databaseURL: "https://fir-search-c1081.firebaseio.com",
  projectId: "fir-search-c1081",
  storageBucket: "fir-search-c1081.appspot.com",
  messagingSenderId: "830000209697",
  appId: "1:830000209697:web:9bc666d33cb9d462bb2860"
};

firebase.initializeApp(firebaseConfig);
export default firebase;
```

## App.js component

```
import { Row,Col ,Container} from 'react-bootstrap'
import './App.css';
import firebase from './components/firebase'
import NavBars from './components/navbar'
import Counter from './components/counter'
import Sidebars from './components/sidebar';
import Post from './components/post'
import Bill from './components/bill'
import MidComponent from './components/midcomponent'
import {BrowserRouter as Router,Switch,Route} from 'react-router-dom';
import SideNavPage from './components/rightnav'
import Homepage from './components/homepage'
import About from './components/about';
import React,{useState} from 'react';
import { divide } from 'lodash';
import SettingsPage from './components/settings';
import LoginPage from './components/loginpage';

const App={()=>{

  const [data,setData] = useState([]);
  const [loading,setLoading] = useState(false);
  const ref= firebase.firestore().collection("schools");
  return(
    <React.Fragment>
      <Switch>
        <Route exact path="/login" component={LoginPage}> </Route>
      <div>

        <div className='App'>
          <NavBars/>

          <Route path="/home" render={({props})=> <Homepage sortBy="newest" {...props} /> } ></Route>
          <Route path="/about" component={About}></Route>
          <Route path="/bill" component={Bill} value={ref}></Route>
          <Route path="/settings/:id" component={SettingsPage}></Route>

        </div>
      </div>
    )
  }
}
```



```

    </Switch>
  </React.Fragment>
);

}

export default App;

```

## REFERENCES

- (2019, 01 11). Retrieved from Open Energy Monitor: <https://openenergymonitor.org/>
- Feynman. (2016, 11 07). *Non-Invasive Current sensor SCT-013-000*. Retrieved from Scidle: <https://scidle.com/non-invasive-current-sensor-sct-013-000/>
- Feynman. (2018, 10 30). *How to use Non-invasive AC Current Sensors with Arduino*. Retrieved from Scidle: <https://scidle.com/how-to-use-non-invasive-accurent-sensors-with-arduino/>
- Grant, C. (2018, 04 20). *Using the ESP8266 WiFi Module with Arduino Uno publishing to ThingSpeak*. Retrieved from Medium: <https://medium.com/@cgrant/usingthe-esp8266-wifi-module-with-arduino-uno-publishing-to-thingspeak99fc77122e82>
- CT sensors - Interfacing with an Arduino*. (2012, 02 20). Retrieved from Open Energy Monitor: <https://openenergymonitor.org/forum-archive/node/156.html>
- Lanka, P. U. (2013). *Applicability of Smart Metering Technology in Sri Lanka*. Colombo: Public Utilities Commission of Sri Lanka.
- LBO. (2016, 01 08). Sri Lanka introduces Smart Metering to reduce electricity generation costs. *Lanka Business Online*.
- Santos, R. (2015, 10 08). *Sending Data From an Arduino to the ESP8266 via Serial*. Retrieved from Randomnerd Tutorials: <https://randomnerdtutorials.com/sending-data-from-an-arduino-to-theesp8266-via-serial/>
- Tech, J. L. (2019, 01 07). *JLanka launches Sri Lanka's first ever Electricity Monitoring ePRO1000 Smart System*. Retrieved from J Lanka Tech: <https://www.jlankatech.com/jlanka-launches-sri-lankas-first-ever-electricitymonitoring-epro1000-smart-system/>
- Datasheet of V988XX- <http://www.vangotech.com/int/uploadpic/152782258251.pdf>