PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)

100 Feet Ring Road, BSK III Stage, Bengaluru-560 085

Department of Computer Science and Engineering

Session: Aug – Dec, 2021

SEMESTER – 5

Assignment-3

IoT based Smart Energy Meter

Rahul S Bhat	PES2UG19CS315
Theertha Swaroop Kumar N	PES2UG19CS430
Tejas R	PES2UG19CS308

a. **Identification of a use case for physical computing project**

In traditional Method, the noting down of energy consumption involves physical visit to the Electricity meters which is very tedious and complex task. Our IoT Project minimises this effort by reading and collecting the of the energy consumption readings in automated fashion in a remote manner.

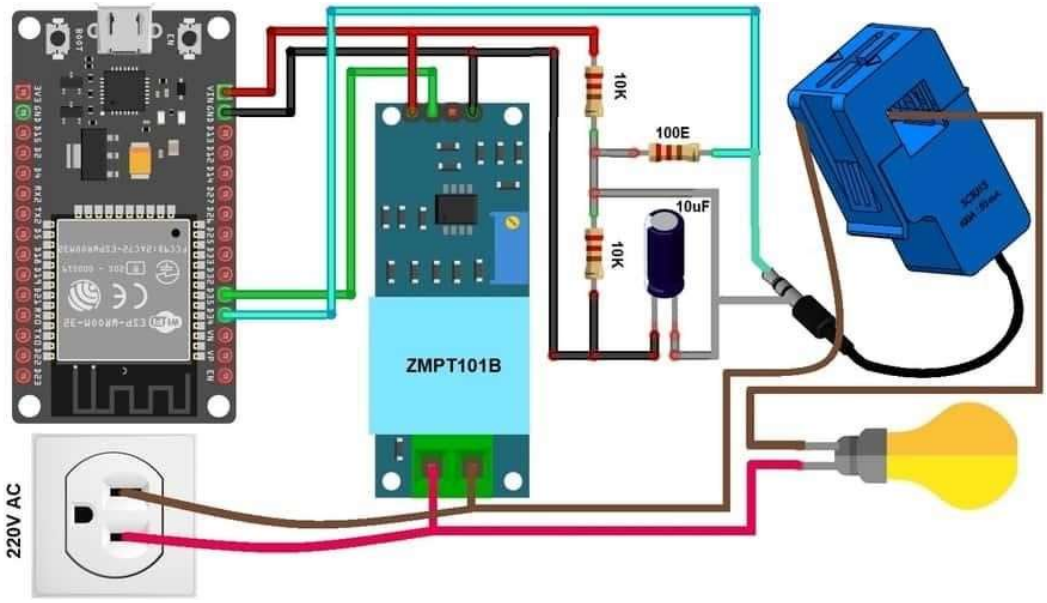
b. **Listing the features planned**

- Developing a user interface(app) for measuring the levels
- make our own IoT Based Electricity Energy Meter using ESP32 & monitor data on the Blynk Application
- making the IoT device to auto update the details of the readings
- Interfacing the current sensor and voltage sensor with a ESP32 WiFi module.

c. **Listing the requirements of SW and HW components to realise the project**

- ESP32 Board
- Voltage Sensor
- Current Sensor (SCT-013-030 Non-invasive AC Current Sensor)
- 10K resistor
- 100 Ω resistor
- Capacitor 10uF
- Connecting Wires (Jumper Wires)
- Breadboard
- Blynk Application

d. **Coming up with the circuit design for the project**

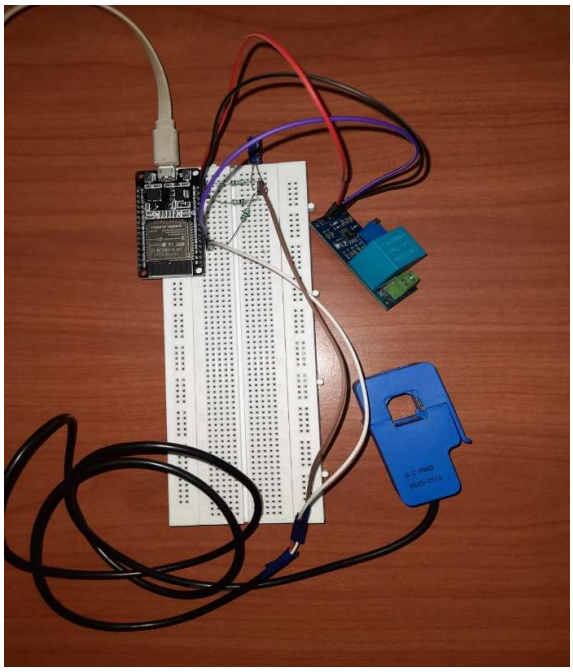


e. **Coming up with the necessary logic to implement all the listed features in (b)**

- First, we include the necessary libraries for ESP32 Board. Emon Lib handles the retrieval of data from both sensors as well as the calculation for the RMS and power values. BlynkSimpleEsp32 integrates the program to the Blynk Mobile app.
- The Energy Monitor object emon is created &calibration factors are defined. The Blynk timer object is then created to handle the sending of data to the Blynk mobile app
- Then we define the SSID & Password on our local Wi-fi network & insert the authentication code from the Blynk
- The Millis & kWh values have to be initialized. The kWh starts at 0 and will slowly go up as time goes on
- The values from the sensors are being retrieved & calculated.

Physical Implementation Of the circuit:

- The Current Sensor and the voltage sensor are connected to esp32 through a breadboard.
- Voltage sensor is connected in parallel to the power supply



Blynk implementation:

- Connecting the ESP32 to the internet using the Blynk platform.
- This platform is used to store, analyze, visualize the data and perform computations if needed.
- It also has a mobile based application to control the IOT system which is connected to the Blynk.
- Blynk Libraries are being used to - enable communication with the server and process all the incoming and outcoming commands

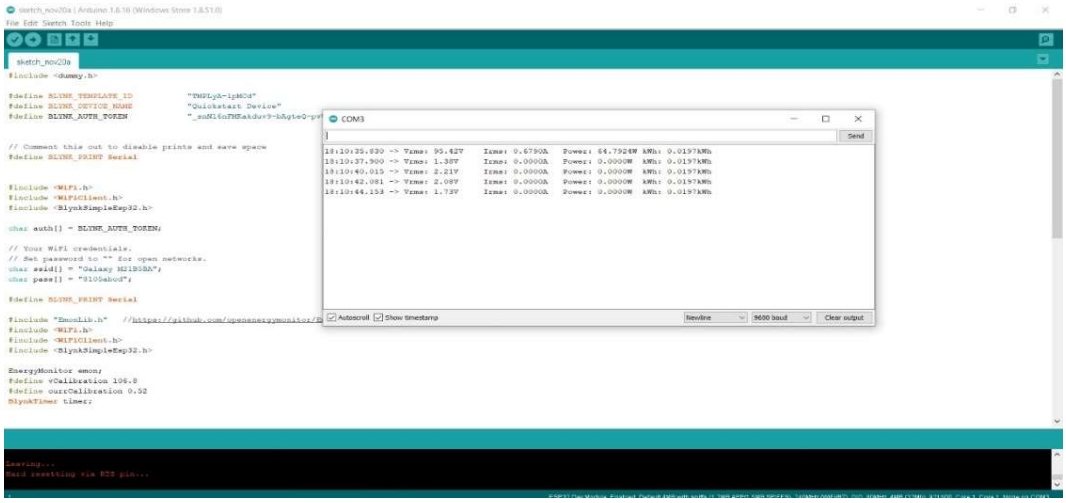
Code:

```
#include <dummy.h>

#define BLYNK_TEMPLATE_ID           "TMPLyA-1pMCd"
#define BLYNK_DEVICE_NAME           "Quickstart Device"
#define BLYNK_AUTH_TOKEN            "_snN16nFMKakduv9-bAgteQ-pvUxZtAs"


// Comment this out to disable prints and save space
#define BLYNK_PRINT Serial


#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>

char auth[] = BLYNK_AUTH_TOKEN;

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "Galaxy M21B5BA";
char pass[] = "8105abcd";

#define BLYNK_PRINT Serial

#include "EmonLib.h"   //https://github.com/openenergymonitor/EmonLib
#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
```

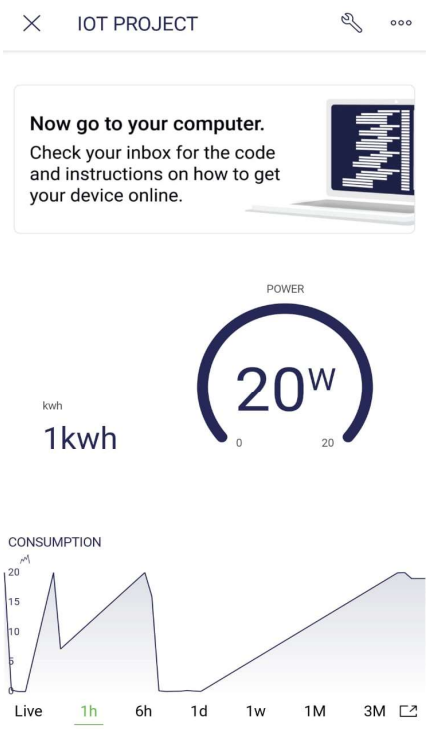
```
EnergyMonitor emon;
#define vCalibration 106.8
#define currCalibration 0.52
BlynkTimer timer;


float kWh = 0;
unsigned long lastmillis = millis();

void myTimerEvent() {
    emon.calcVI(20, 2000);
    Serial.print("Vrms: ");
    Serial.print(emon.Vrms, 2);
    Serial.print("V");

    Serial.print("\tIrms: ");
    Serial.print(emon.Irms, 4);
    Serial.print("A");

    Serial.print("\tPower: ");
    Serial.print(emon.apparentPower, 4);
    Serial.print("W");

    Blynk.virtualWrite(V5, emon.apparentPower);
    Serial.print("\tkWh: ");


    kWh = kWh + emon.apparentPower*(millis()-lastmillis)/3600000000.0;
    Serial.print(kWh, 4);
    Serial.println("kWh");
    lastmillis = millis();
    Blynk.virtualWrite(V4, kWh*1000);
}

void setup() {
  Serial.begin(9600);
  emon.voltage(35, vCalibration, 1.7); // Voltage: input pin, calibration, phase_shift
  emon.current(34, currCalibration); // Current: input pin, calibration.
  Blynk.begin(auth, ssid, pass);
  timer.setInterval(1000L, myTimerEvent);
}

void loop() {
  Blynk.run();
  timer.run();
}
```

Screenshots of the Output:

Serial monitor



Graphical Visualisation of the Energy Readings in the Blynk App:



f. **Testing and packaging the circuit**

a. The ESP32 Board will try connecting to the Wi-fi Network using the given SSID & Password
b. When no load is connected or when the load is powered off the Current and Voltage parameters should be almost 0.
c. when the load is connected, the blynk app will display the Voltage and Current value Screen along with the Power Consumption and total kWh units.
d. The energy meter data is uploaded to Blynk Application an interval of every 5 seconds. The same data can be observed on Serial Monitor as well as Blynk Application

References:

https://docs.blynk.cc/