**IOT BASED ELECTRICITY ENERGY METER USING ESP32 & BLYNK**

**A Mini Project Report**

Submitted

In partial fulfillment of the requirements

for the award of the Degree of

**Bachelor of Technology**

In

**Electronics and Communication Engineering**

By

**B.Naga Mani Abdul Sattar**

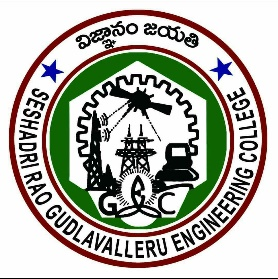
**(19481A0417) (20485A0401)**

**Ch.Jaimini Sowmya G. Ravi Kiran**

**(1981A0444) (19481A0460)**

Under the Guidance of

**Dr.Ch.Balaswamy**



**Department of Electronics and Communication Engineering**

**SESHADRI RAO GUDLAVALLERU ENGINEERING COLLEGE**

SESHADRI RAO KNOWLEDGE VILLAGE

GUDLAVALLERU - 521356

ANDHRA PRADESH

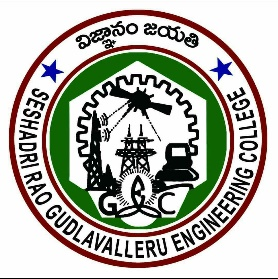
2022-2023

**Department of Electronics and Communication Engineering**

**SESHADRI RAO GUDLAVALLERU ENGINEERING COLLEGE**

SESHADRI RAO KNOWLEDGE VILLAGE

GUDLAVALLERU – 521356



**CERTIFICATE**

This is to certify that the project report entitled **“IOT Based Electricity Energy Meter Using ESP32 & Blynk”** is a bonafide record of work carried out by B.Naga Mani(19481A0417), Abdul Sattar(20485A0401), Ch.Jaimini Sowmya(19481A0444), G.Ravi Kiran(19481A0460)under my guidance and supervision in partial fulfillment of the requirements, for the award of the degree of Bachelor of Technology in Electronics and Communication Engineeringby Jawaharlal Nehru Technological University, Kakinada

**Dr.Ch.Balaswamy Dr.Y.RAMA KRISHNA**

Project Guide Head of the Department

**Acknowledgement**

We are very glad to express our deep sense of gratitude to Dr.Ch.Bala Swamy,**Professor, ECE DEPT, SRGEC, GUDLAVALLERU** for guidance and cooperation in completing this mini project. We convey our heartfelt thanks to him for his inspiring assistance till the end of our mini project.

We convey our sincere and indebted thanks to our beloved Head of the Department **Dr. Y. Ramakrishna**, for his encouragement and help in completing our mini project successfully.

We also extend our gratitude to our Principal **Dr. G.V.S.N.R.V. Prasad**, for the support and for providing facilities required for the completion of our mini project.

We impart our heartfelt gratitude to all the Lab Technicians for helping us in all aspects related to our mini project.

We thank our friends and all others who rendered their help directly and indirectly to complete our mini project.

B.Naga Mani (19481A0417)

Abdul Sattar (20485A0401)

Ch.Jaimini Sowmya (19481A0444)

G.Ravi Kiran (19481A0460)

**CONTENTS**

**TITLE PAGE NO**

LIST OF FIGURES vi

LIST OF TABLES vii

ABSTRACT viii

**CHAPTER-1: INTRODUCTION**  9

* 1. Background 10
  2. Aim of the project 10
  3. Methodology 10
  4. Significance of the project 10
  5. Outline 11
  6. Conclusion 11

**CHAPTER-2: LITERATURE SURVEY**  12

2.1 Introduction 12

2.2 Existing solution 13

2.3 Proposed solution 13

**CHAPTER-3: WORK TITLE EXPLANATION**  14

3.1 Explanation of Circuit diagram 15

**CHAPTER-4: IMPLEMENTATION**  16

4.1Arduino IDE Definition 16

4.2 How to Get Arduino IDE 16

4.3 Selecting Board of ESP32 21

4.4 Blynk App 23

4.5 Program Description 24

**CHAPTER-5: HARDWARE DISCRIPTION**  28

5.1 Hardware Components Used 28

5.1.1 SCT-013 Current Sensor 29

5.1.2 ZMPT10 1B AC Single phase voltage sensor 29

5.1.3 ESP32 30

**CHAPTER-6: RESULTS**  32

**CHAPTER-7: CONCLUSION**  39

**BIBLIOGRAPHY**

**LIST OF FIGURES**

**Fig. No. NAME OF THE FIGURE**

3.1 Explanition of Circuit Diagram

4.2  How to Get Arduino ID

4.3Selecting Board of ESP32

4.3.4 Setting Band Rate

4.4.1 Adding Libraries

4.5.1 Selecting Board

4.7.1 Blynk Application

5.1.1 ESP32

5.1.2 SCT-013 Current Sensor

5.1.3 ZMPT10 1B AC Single phase voltage sensor

**LIST OF TABLES**

**Table No. NAME OF THE TABLE**

5.1 List of Components Used

**ABSTRACT**

In this project, we will learn how to make our own **IoT Based Electricity Energy Meter** using **ESP32** & monitor data on the **Blynk Application**. With the current technology, you need to go to the **meter reading** room and take down readings. Thus monitoring and keeping track records of your **electricity consumption** is a tedious task. To automate this, we can use the Internet of Things. The Internet of Things saves time and money by automating **remote data collection**. Smart **Energy Meter** has received quite a lot of acclaim across the globe in recent years. So, why not to build our own **IoT Based Electricity Energy Meter**?

**CHAPTER -1**

**INTRODUCTION**

Electric energy use has surged in recent years. As a result, a large increase in energy

Supply was required, due to population growth and other factors in the coming decades

Development of the economy as a result, there is a demand-supply imbalance. According to

The current scenario, the power generated, which is mostly derived from fossil fuels, will be

Depleted within the next 20 years. Electronic energy monitoring solutions are currently available on the market that is extremely accurate. In the case of residential applications, the majority of these monitor the power utilized in a domestic household. Consumers are frequently disappointed with their power bills since they do not display the power used at the device level. The Internet of Things (IoT) is a new sector, and IoT-based devices have ushered in a revolution in electronics and information technology.

Energy usage, particularly electricity consumption, is one of the most critical issues we face today. An effective technique to monitor this energy consumption is required. The Internet of Things (IOT) offers a solution to these issues. Hardware, software, and the cloud are all interconnected. As a result, we offer an energy consumption model. Household appliance monitoring system that can be used to calculate energy consumption of the family and to keep the user up to date on his or her electricity usage and be able to make informed decisions .With the advent of Internet of Things (IOT) technology, an existing energy meter with an industrial communication protocol can be adapted to improve connection and observability of power and energy consumption. This can be accomplished by utilizing IOT technology. As a result, this study presents an approach that incorporates IoT technology so that current digital energy meters in buildings can be modified to enable for online monitoring.

* 1. **Background**

In general we measure readings by manual readings that is electrician need to visit each and every electric meter which is an hectic task. So IOT technology is used to make this task simple. That’s the objective for this project.

* 1. **AIM OF THE PROJECT**

The main aim of the project is to develop an IOT based electricity energy meter Using ESP32 to calculate the voltage, current readings from which can determine power and energy consumption of an user load.

**1.3 Methodology**

This project, Smart Energy Electricity Meter, enables an innovative and crisp methodology. In this we use ESP32 which consists of both Wi-Fi and Bluetooth modules and current, voltage sensors are used to measure current and voltages. These measured values are sent to email through Blynk software which can be monitored through our mobile phones.

**1.4 Significance of the project**

* The main aim of the project is to know the consumption of power and to reduce the wastage of power.
* It sends the alert notifications to the user and the user responds accordingly with the help of the connected device like a smart phone from any location.
  1. **Outline**

In this report, first chapter deals with introduction which gives the background and aim of the project. Second chapter deals with literature survey which gives the details about existing system and proposed system. Third chapter deals with work title explanation which includes the block diagram. Fourth chapter deals with hardware implementation in which the hardware components used in this project are explained. Fifth chapter deals with Results of this project which gives the screenshots obtained about the result implementing the project. Sixth project deals with advantages and future scope of the project.

**1.6 Conclusion:**

After working on this project we can say that it will be helpful and useful. Through this project one can know their daily consumption of power, so that one can reduce the rate of consumption.

**CHAPTER-02**

**LITERATURE SURVEY**

**2.1 Introduction**

An IoT-based smart energy meter was developed by Kumar L using AT Mega 328p microcontroller, voltage sensor, current sensor, ESP 8266 Wi-Fi chip. The ESP 8266 sends data to the Thing Speak cloud platform through the internet. When updating data to the cloud fails, an alternative solution is a 4G GSM module that sends data to the appropriate contacts as an SMS. The energy meter calculates the amount of energy consumed each hour using Fast Fourier Transform as well as the cost of that consumption. However, because the AT Mega 328p is expensive, a low-cost microcontroller is required for use in smart meters.

Nowadays, the demand for electricity is rapidly increasing around the world, and most specifically in Saudi Arabia. In fact, a study that was conducted three years ago, and published by the Saudi Press Agency (SPA), recorded unprecedented energy consumption in the Kingdom. On August 31st, 2014, the electricity usage during the peak time reached

56.500MW, that’s an increase by 7.7% of the recorded values of the summer of 2013

Moreover, studies showed that Saudi Arabia is one of the highest countries in the world in regard to electricity demand, where the per capita usage is equal to more than double the international average, with an annual increase of 5.5%. In addition, a recent study that was carried out back in October of 2016 estimated the per capita usage / month in Saudi Arabia in the entire year of 2015 to be 779KWh

**2.1 EXISTING SYSTEM**

Many researches have worked on this smart energy meter but they are varied based on the equipment or software used but the only purpose is for the efficient calculation of energy consumption. So now we are going to modify the existing design in order to increase its efficiency and to overcome some of the drawbacks observed in it.

**2.2 PROPOSED SYSTEM**

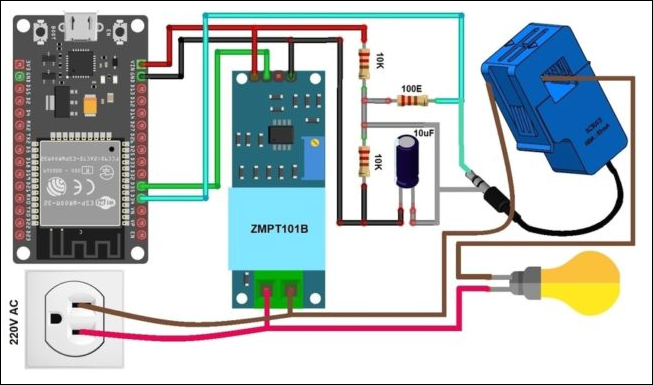
The system mainly consists of ESP32 which is the best version among all which is of both wifi and Bluetooth module device. It comprises of voltage sensor for the calculation of amount of voltage and a current sensor which gives the measure of current from these we can calculate and power and energy consumption that are displayed to the user through the blynk software to their personal device in the form of a notification.

**CHAPTER- 03**

**WORK TITLE EXPLANATION**

**3.1 Explanation of Circuit Diagram:**

Now let us see the **circuit diagram** of IoT Based Electricity Energy Meter using ESP32. The circuit has been designed using Fritzing software.



The connection diagram is simple. Both the Sensor, i.e. SCT-013 Current Sensor & ZMPT101B Voltage Sensor VCC is connected to **Vin** of ESP32 which is a **5V Supply**. The GND pin of both the modules is connected to the **GND** of ESP32. The output analog pin of the ZMPT101B Voltage Sensor is connected to **GPIO35** of ESP32. Similarly, the output analog pin of SCT-013 Current Sensor is connected to **GPIO34**of ESP32.

You need a two resistor of **10K** & a single resistor of **100 ohms** connected along with a **10uF** Capacitor. Apart from the circuit part, the AC wires where the current and voltage needs to measure are connected to the **input AC Terminal** of Voltage Sensor.

Similarly, the **current sensor clip** doesn’t have any connection and a single live wire or neutral wire is inserted inside the clip part as shown in the above circuit.

The **16x2 LCD** used in this project is optional. There is no need to connect the LCD as we will be monitoring the ESP32/SCT-013 ZMPT101B/ Energy Meter Data on Blynk Application. In case you want to connect the LCD, you need so many connections. Connect the pin numbers **4, 6, 11, 12, 13, 14** of LCD to ESP32 **D13, D12, D14, D27, D26, D25** pins. Also connect the LCD **1, 5, 16** Pin to GND & **2, 15** Pin to 5V VCC. Use a **10K Potentiometer** at Pin 3 of LCD to adjust the LCD Contrast

**CHAPTER-04**

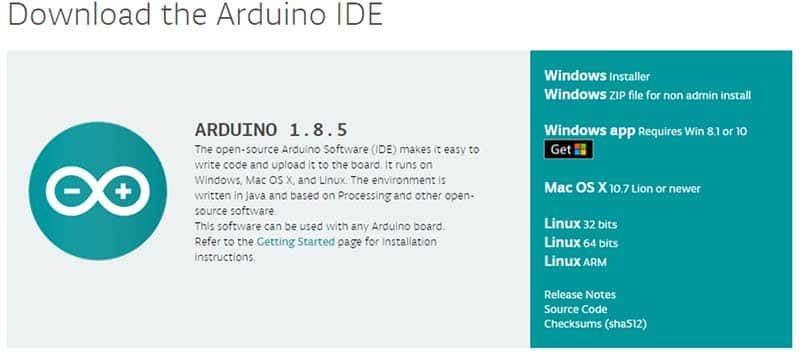
**IMPLEMENTATION**

**4.1 Arduino IDE Definition:**

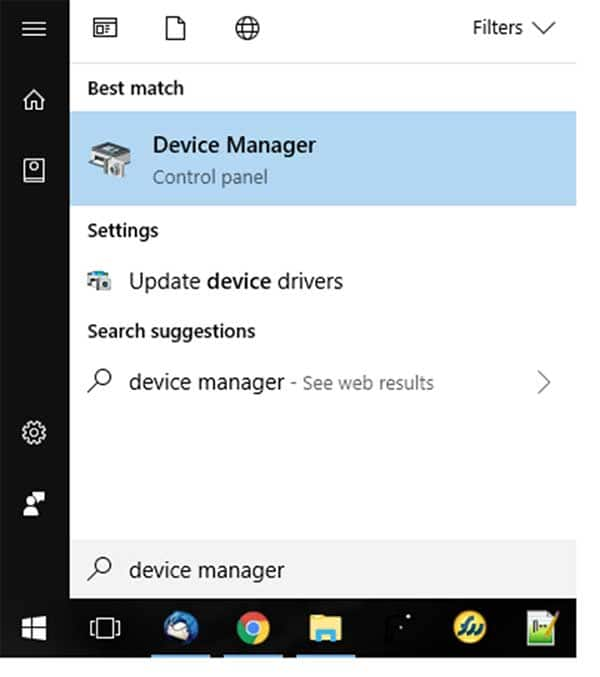
The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them.

**4.2 How to Get Arduino IDE :**

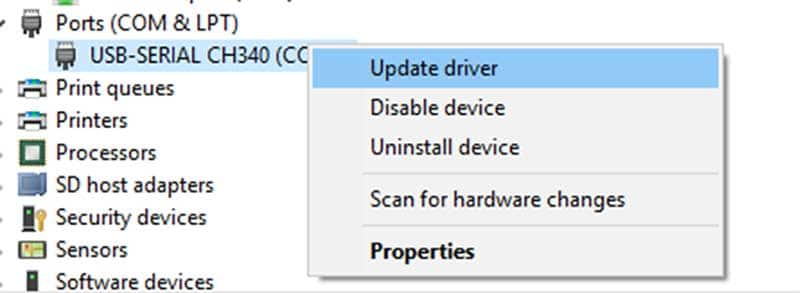
You can download the IDE from the official [Arduino website](https://www.arduino.cc/en/Main/Software" \t "https://www.digikey.in/en/maker/blogs/2018/_blank). Since the Arduino uses a USB to serial converter (which allow it to communicate with the host computer), the Arduino board is compatible with most computers that have a USB port. Of course, you will need the IDE first. Luckily, the Arduino designers have released multiple versions of the IDE for different operating systems, including Windows, Mac, and Linux. In this tutorial, we will use Window 10, so ensure that you download the correct version of the IDE if you do not have Windows 10.

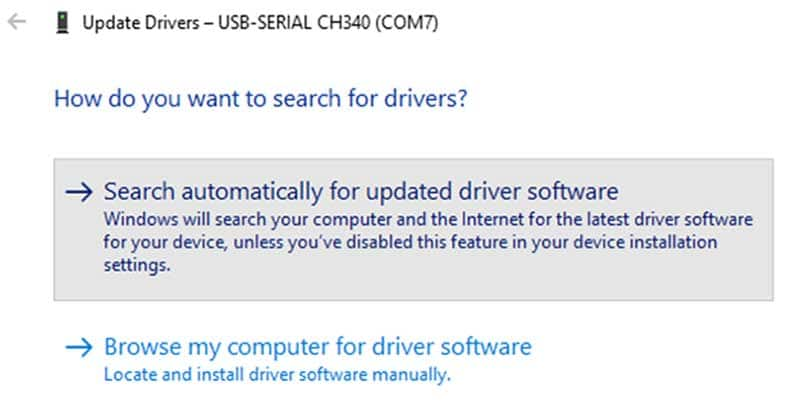


Next, you’ll need to connect the Ardunio UNO broad to the computer. This is done via a USB B connection. Thanks to the wonderful world of USB, we do not need to provide power to the Arduino , as the USB provides 5V up to 2A.



In the Device Manager window, look for a device under “Ports (COM & LPT)”, and chances are the Arduino will be the only device on the list. In my Device Manager, the Arduino shows up as COM7 (I know this because CH340 is in the device name).



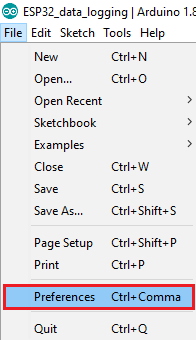


Windows can be a real pain sometimes with COM ports, as it can magically change their numbers between connections. In other words, one day, your Arduino may be on port 7 (as shown here), but then on other days, Windows may shift it to a different port number. As I understand it, this happens when you connect other COM ports to your system (which I do frequently).So, if you can’t find your Arduino on the port that you usually use, just go to your Device Manager and check what port it’s actually on and, if necessary, update your driver.

### Configure the IDE

Now that we have determined the COM port that the Arduino is on, it’s time to load the Arduino IDE and configure it to use the same device and port. Start by loading the IDE. Once it’s loaded, navigate to Tools > Board > Arduino Uno. However, if you are using a different board (i.e., not the Arduino Uno), you must select the proper board.

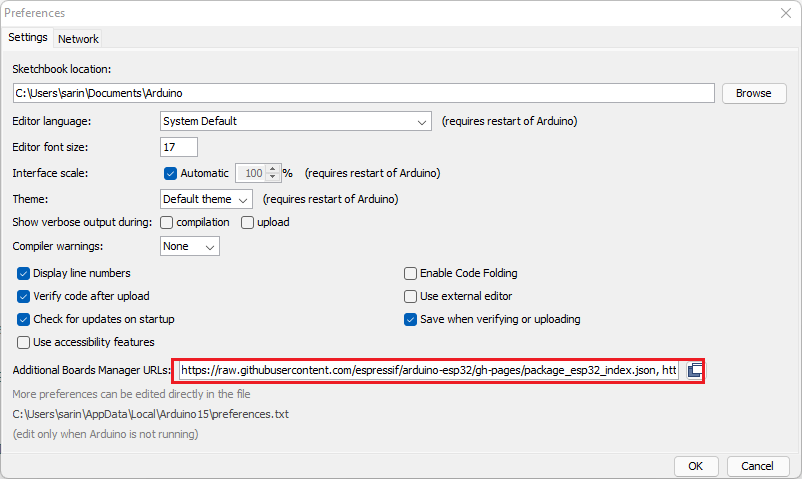
In your Arduino IDE, go to **File**> **Preferences**



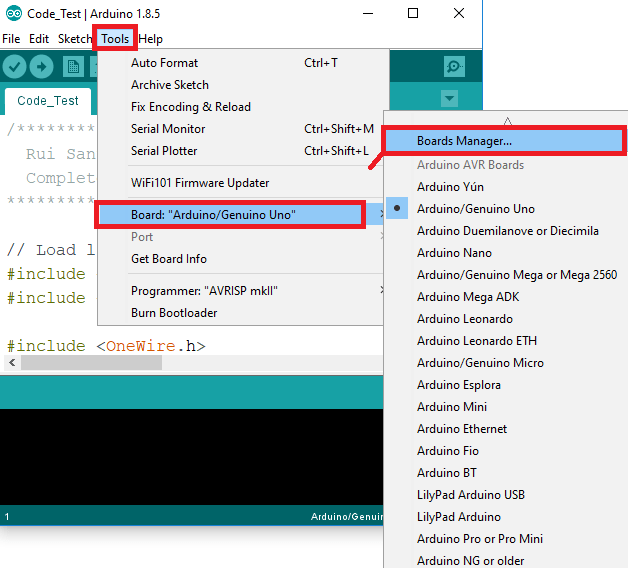
Enter the following into the “Additional Board Manager URLs” field:

https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package\_esp32\_index.json

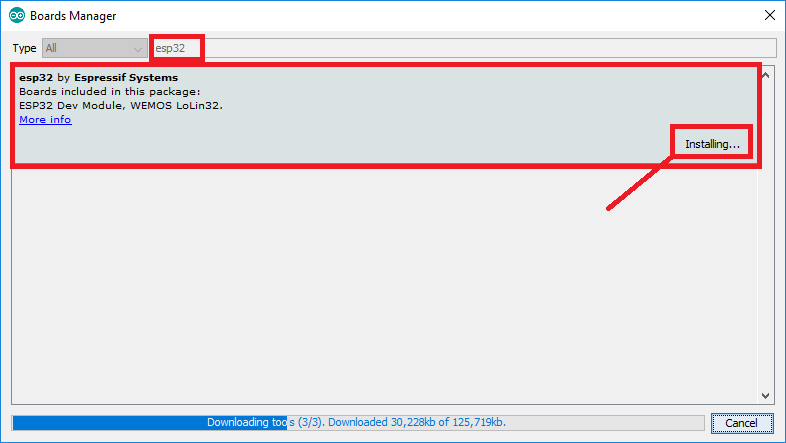
Then, click the “OK” button:



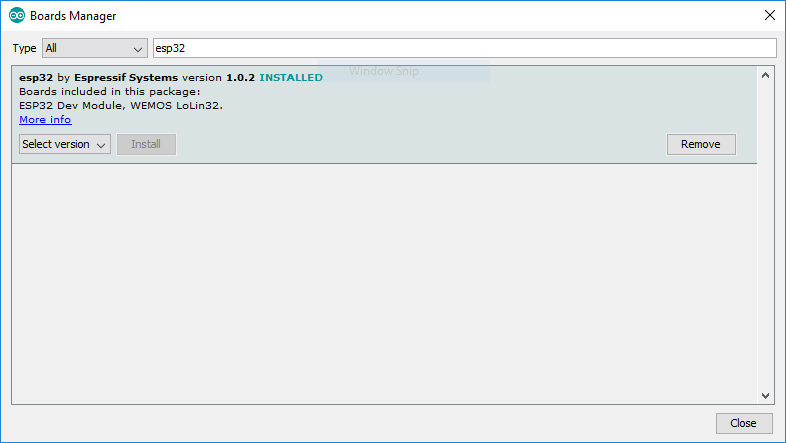
Open the Boards Manager. Go to **Tools** > **Board** > **Boards Manager…**



Search for **ESP32** and press install button for the “**ESP32 by Espressif Systems**“:

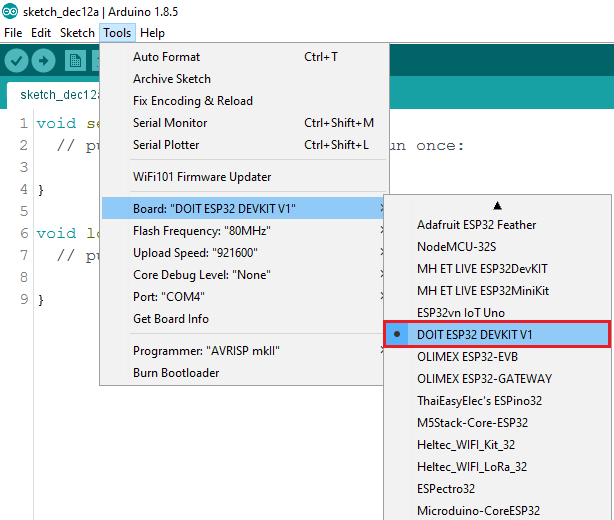


That’s it. It should be installed after a few seconds.

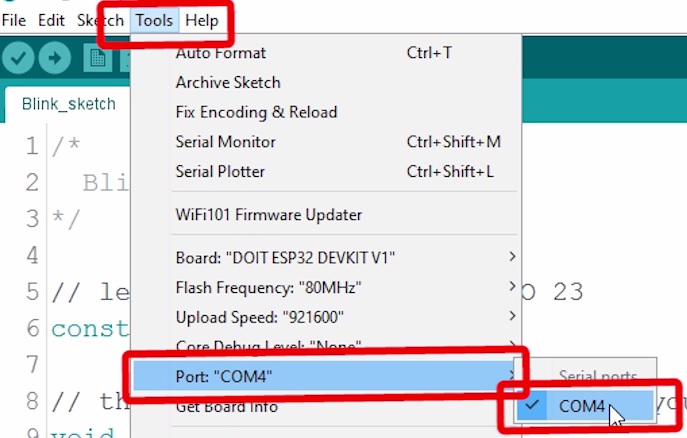


**4.3 Selecting Board of ESP32 :**

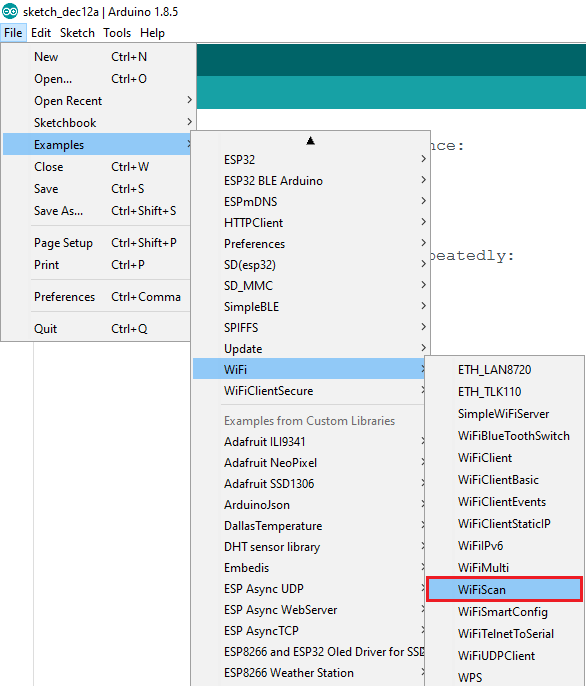
Select your Board in **Tools** > **Board** menu (in my case it’s the **DOIT ESP32 DEVKIT V1**)



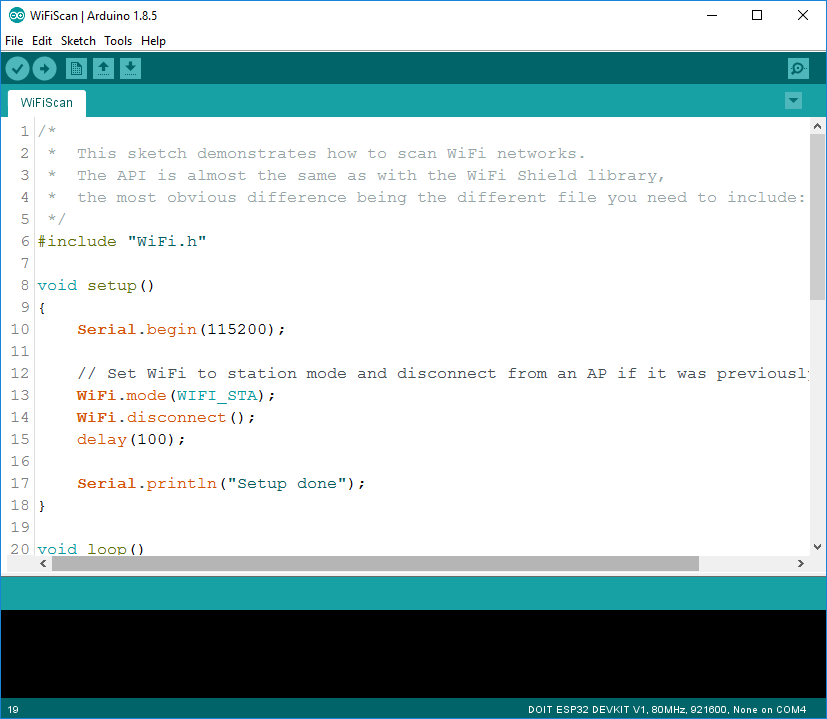
Select the Port (if you don’t see the COM Port in your Arduino IDE, you need to install the [CP210x USB to UART Bridge VCP Drivers](https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers" \t "https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-windows-instructions/_blank)):



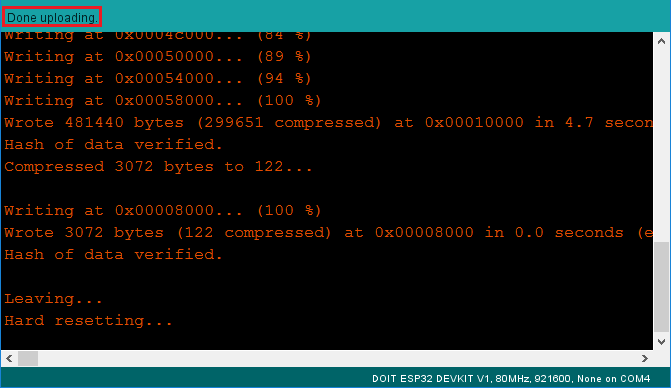
Open the following example under **File** > **Examples** > **WiFi (ESP32)** > **WiFiScan**



A new sketch opens in your Arduino IDE:



Press the **Upload** button in the Arduino IDE. Wait a few seconds while the code compiles and upload to the device

.

If everything went as expected, you should see a “**Done uploading.**” Message

**4.4 Blynk App :**

[Blynk](https://blynk.io/" \t "https://how2electronics.com/iot-based-electricity-energy-meter-using-esp32-blynk/_blank) is an application that runs over**Android** and **IOS** devices to **control any IoT based application** using Smartphones. It allows you to create your Graphical user interface for IoT application. Here we will display the IoT Energy Meter Data on Blynk Application. So download and install the **Blynk Application** from [Google Play Store](https://play.google.com/store/apps/details?id=cc.blynk&hl=en_IN&gl=US" \t "https://how2electronics.com/iot-based-electricity-energy-meter-using-esp32-blynk/_blank). IOS users can download from the [App Store](https://apps.apple.com/us/app/blynk-iot/id1559317868" \t "https://how2electronics.com/iot-based-electricity-energy-meter-using-esp32-blynk/_blank). Once the installation is completed, open the app & sign-up using your Email id and Password.From the dashboard create a new project and select ESP32 & Wi-Fi Connection. Then drag & drop or add 4 widgets and assign the variable as per code and then email the authentication code. You will get the authentication code in the mail. Copy this authentication code. This will be used in your code.

**Required Library Installation :**

**1. EmonLib Library :**  
 The Emonlib Library is used for **Electricity Energy Meter**. EmonLib is a Continuous Monitoring of Electricity Energy repeats, every **5 or 10s**, a sequence of voltage and current measurements. EemonLib continuously measures in the background the voltage and all the current input channels, calculates a **true average quantity** for each and then informs the sketch that the measurements are available and should be read and processed.

**2. Blynk Library**  
 Blynk is the most popular **Internet of Things** platform for connecting any hardware to the cloud, designing apps to control them, and managing your deployed products at scale. With Blynk Library you can connect over 400 hardware models including A**rduino, ESP8266 & ESP32** to the **Blynk Cloud**.

### 4.5 PROGRAM DESCRIPTION:

**#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;**

**charauth[]="\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";**

**charssid[]="\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";**

**charpass[]="\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*3";**

**floatkWh=0;**

**unsignedlonglastmillis=millis();**

**voidmyTimerEvent(){**

**emon.calcVI(20,2000);**

**Serial.print("Vrms: ");**

**Serial.print(emon.Vrms,2);**

**Serial.print("V");**

**Blynk.virtualWrite(V0,emon.Vrms);**

**Serial.print("\tIrms: ");**

**Serial.print(emon.Irms,4);**

**Serial.print("A");**

**Blynk.virtualWrite(V1,emon.Irms);**

**Serial.print("\tPower: ");**

**Serial.print(emon.apparentPower,4);**

**Serial.print("W");**

**Blynk.virtualWrite(V2,emon.apparentPower);**

**Serial.print("\tkWh: ");**

**kWh=kWh+emon.apparentPower\*(millis()-lastmillis)/3600000000.0;**

**Serial.print(kWh,4);**

**Serial.println("kWh");**

**lastmillis=millis();**

**Blynk.virtualWrite(V3,kWh);**

**}**

**voidsetup(){**

**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(5000L,myTimerEvent);**

**}**

**voidloop(){**

**Blynk.run();**

**timer.run();**

**}**

**#include <LiquidCrystal.h>**

**LiquidCrystal lcd(13,12,14,27,26,25);**

**#define BLYNK\_PRINT Serial**

**#include "EmonLib.h"**

**#include <WiFi.h>**

**#include <WiFiClient.h>**

**#include <BlynkSimpleEsp32.h>**

**EnergyMonitor emon;**

**#define vCalibration 83.3**

**#define currCalibration 0.50**

**BlynkTimer timer;**

**charauth[]="hsYG\_5da4gdP9jZkL18O5RNcJSrBT-Ou";**

**charssid[]="Alexahome";**

**charpass[]="loranthus";**

**floatkWh=0;**

**unsignedlonglastmillis=millis();**

**voidmyTimerEvent()**

**{**

**emon.calcVI(20,2000);**

**kWh=kWh+emon.apparentPower \*(millis()-lastmillis)/3600000000.0;**

**yield();**

**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");**

**Serial.print("\tkWh: ");**

**Serial.print(kWh,5);**

**Serial.println("kWh");**

**lcd.clear();**

**lcd.setCursor(0,0);**

**lcd.print("Vrms:");**

**lcd.print(emon.Vrms,2);**

**lcd.print("V");**

**lcd.setCursor(0,1);**

**lcd.print("Irms:");**

**lcd.print(emon.Irms,4);**

**lcd.print("A");**

**delay(2500);**

**lcd.clear();**

**lcd.setCursor(0,0);**

**lcd.print("Power:");**

**lcd.print(emon.apparentPower,4);**

**lcd.print("W");**

**lcd.setCursor(0,1);**

**lcd.print("kWh:");**

**lcd.print(kWh,4);**

**lcd.print("W");**

**delay(2500);**

**lastmillis=millis();**

**Blynk.virtualWrite(V0,emon.Vrms);**

**Blynk.virtualWrite(V1,emon.Irms);**

**Blynk.virtualWrite(V2,emon.apparentPower);**

**Blynk.virtualWrite(V3,kWh);**

**}**

**voidsetup()**

**{**

**Serial.begin(9600);**

**Blynk.begin(auth,ssid,pass);**

**lcd.begin(16,2);**

**emon.voltage(35,vCalibration,1.7);// Voltage: input pin, calibration, phase\_shift**

**emon.current(34,currCalibration);// Current: input pin, calibration.**

**timer.setInterval(5000L,myTimerEvent);**

**lcd.setCursor(3,0);**

**lcd.print("IoT Energy");**

**lcd.setCursor(5,1);**

**lcd.print("Meter");**

**delay(3000);**

**lcd.clear();**

**}**

**voidloop()**

**{**

**Blynk.run();**

**timer.run();**

**}**

**CHAPTER – 5**

**HARDWARE DISCRIPTION:**

**5.1Hardware Componets Used:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **Component** | **Specification** | **Quantity** |
| 1 | ESP32 Board | ESP32 Development Board (ESP-WROOM-32) | 1 |
| 2 | Voltage Sensor | ZMPT101B AC Voltage Sensor Module | 1 |
| 3 | Current Sensor | SCT-013-030 Non-invasive AC Current Sensor | 1 |
| 4 | Resistor | 10K | 2 |
| 5 | Resistor | 100 ohm | 1 |
| 6 | Capacitor | 10uF | 1 |
| 7 | Connecting Wires | Jumper Wires | 10 |
| 8 | Breadboard | - | 1 |

## ****SCT-013 Current Sensor****

The **[SCT-013](https://www.mcielectronics.cl/website_MCI/static/documents/Datasheet_SCT013.pdf" \t "_blank)** is a Non-invasive AC Current Sensor Split Core Type Clamp Meter Sensor that can be used to measure **AC current** up to 100 amperes. Current transformers (CTs) are sensors are for measuring alternating current. They are particularly useful for measuring whole building electricity consumption.



The SCT-013 current sensors can be clipped straight either to the live or neutral wire without having to do any high voltage electrical work. Like any other transformer, a **current transformer** has a primary winding, a magnetic core, and a secondary winding. The secondary winding comprises many turns of fine wire housed within the casing of the transformer.

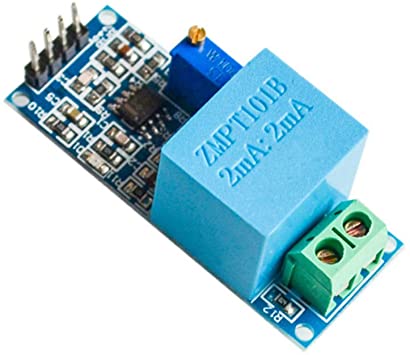
### ****Specifications****

1.Input Current: ***0-30A AC***2. Output Signal: ***DC 0-1 V***3. Non-linearity: ***2-3 %***4. Build-in sampling resistance (RL): ***62 Ω***5. Turn Ratio: ***1800:1***6. Resistance Grade: ***Grade B***7. Work Temperature: ***-25 °C~+70 °C***8. Dielectric Strength (between shell and output): ***1000 V AC / 1 min 5 mA***

## ****ZMPT101B AC Single Phase Voltage Sensor****

The **[ZMPT101B](http://5nrorwxhmqqijik.leadongcdn.com/attachment/kjilKBmoioSRqlkqjoipSR7ww7fgzb73m/ZMPT101B-specification.pdf" \t "_blank)** AC Single Phase voltage sensor module is based on **a high precision ZMPT101B voltage Transformer** used to measure the accurate AC voltage with a voltage transformer. This is an ideal choice to measure the AC voltage using Arduino or ESP32.

The Modules can measure voltage within **250V AC voltage** & the corresponding analog output can be adjusted. The module is simple to use and comes with a **multi-turn trim potentiometer** for adjusting and calibrating the ADC output.



### ****Specifications****

1. Voltage up to ***250 volts*** can be measured2. Lightweight with on-board micro-precision voltage transformer3. High precision on-board op-amp circuit4. Operating temperature : ***40ºC ~ + 70ºC***5. Supply voltage ***5 volts*** to 30 volts

**ESP32 Board**

It's a low-footprint, minimal system development board powered by the latest ESP-WROOM-32 module and can be easily inserted into a solderless breadboard.The ESP32-DevKitC contains the entire basic support circuitry for the ESP-WROOM-32, including the USB-UART bridge, reset- and boot-mode buttons, LDO regulator and a micro-USB connector. Every important GPIO is available to the developer.

**Features**

* Small volume, easily embeded to other products
* Strong function with support LWIP Protocol, Freertos
* Suppporting three modes :  AP, STA, and AP+STA
* Supporting Lua program, easily to develop
* Includes CP2102 USB-UART bridge

.



Esp32 is already integrated antenna and rf balun, power amplifier, low-noise amplifiers, filters, and power management module.The entire solution takes up the least amount of printed circuit board area this board is used with 24 ghz dual-mode wi-fi and bluetooth chips by tsmc 40nm low power technology, power and rf properties best, which is safe, reliable, and scalable to a variety of applications

**CHAPTER – 6**

**RESULTS**

The ESP32 Board will try connecting to the Wi-Fi Network using the given SSID & Password. The LCD Display will light up with the following message Initially.

****

Initially when no load is connected it shows that the readings are zero.

|  |
| --- |
| #define vCalibration 106.8  #define currCalibration 0.5 |



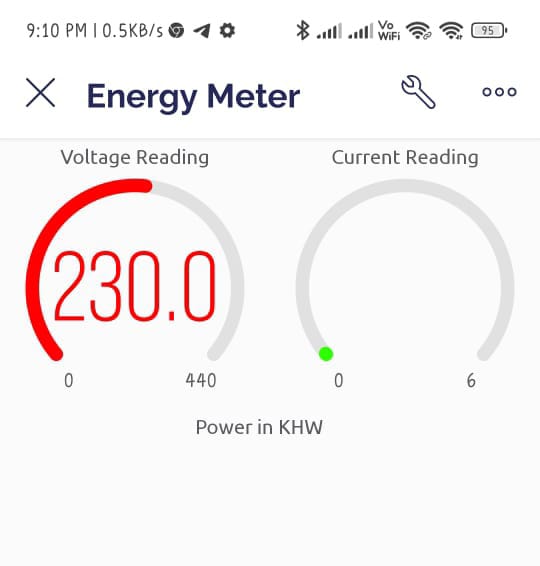
When load is connected,the change in readings can be observed.Here the voltage reading is displayed in guage settings.

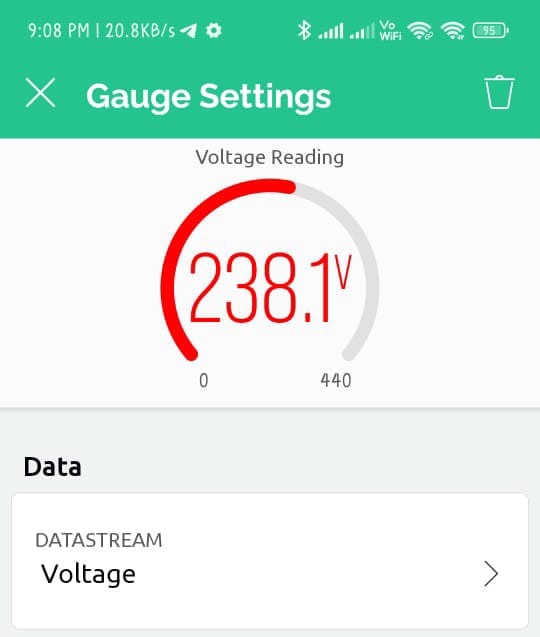
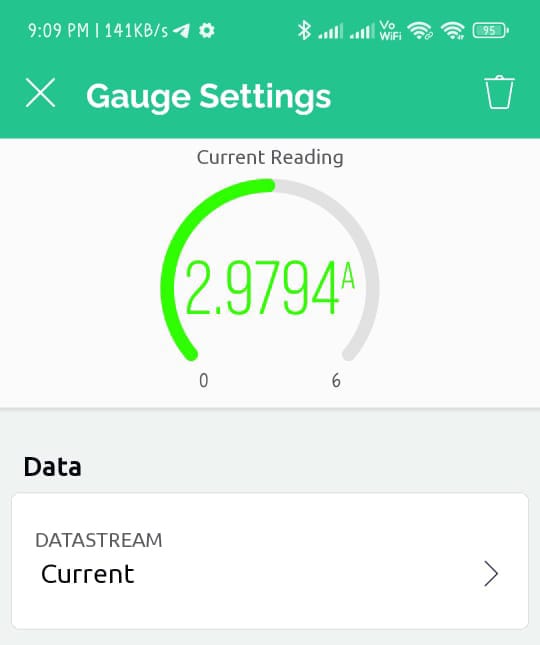
****

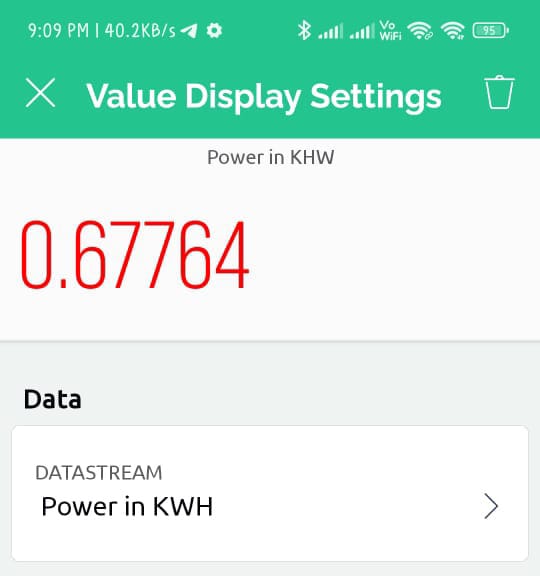
Similarly we can get the current readings in the guage settings.Thus the current and voltage readings of bulb are displayed.



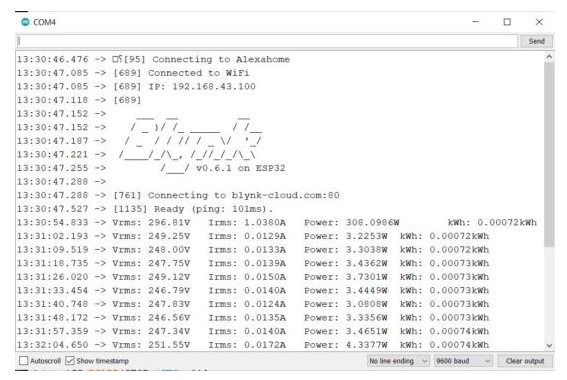
Now in value display settings we can get the readings of power.

****



The energy meter data is uploaded to **Blynk Application** after the interval of every **5 seconds**. The data can be observed on Serial Monitor as well as Blynk Application.



Thus, you can use this **IoT Based Electricity Energy Meter** to monitor the electricity consumption of your house.

**CHAPTER-7**

**CONCLUSION AND FUTURE SCOPE**

n today's smart world environment, the coexistence of many networks with different

characteristics and energy efficiency is at the top of the list of hurdles to the realization of an

automated ESP32 electric energy consumption meter. The architecture for the energy

monitoring was outlined in this study. A monitoring system that is aware of both energy and

power consumption. The proposed smart home energy monitoring system is automated. To

collect sensor data and control the system, the ESP32 Microcontroller interfaced with the current

transformer and voltage sensor where prototyped and the blynk cloud was used for retrieving

sensor data to web clients.

The results of the experiments revealed that the designed energy monitoring system can

successfully monitor voltage, current, active power, and temperature. electricity consumption

n today's smart world environment, the coexistence of many networks with different

characteristics and energy efficiency is at the top of the list of hurdles to the realization of an

automated ESP32 electric energy consumption meter. The architecture for the energy

monitoring was outlined in this study. A monitoring system that is aware of both energy and

power consumption. The proposed smart home energy monitoring system is automated. To

collect sensor data and control the system, the ESP32 Microcontroller interfaced with the current

transformer and voltage sensor where prototyped and the blynk cloud was used for retrieving

sensor data to web clients.

The results of the experiments revealed that the designed energy monitoring system can

successfully monitor voltage, current, active power, and temperature. electricity consumption

n today's smart world environment, the coexistence of many networks with different

characteristics and energy efficiency is at the top of the list of hurdles to the realization of an

automated ESP32 electric energy consumption meter. The architecture for the energy

monitoring was outlined in this study. A monitoring system that is aware of both energy and

power consumption. The proposed smart home energy monitoring system is automated. To

collect sensor data and control the system, the ESP32 Microcontroller interfaced with the current

transformer and voltage sensor where prototyped and the blynk cloud was used for retrieving

sensor data to web clients.

The results of the experiments revealed that the designed energy monitoring system can

successfully monitor voltage, current, active power, and temperature. electricity consumption

In today's smart world environment, the coexistence of many networks with different

characteristics and energy efficiency is at the top of the list of hurdles to the realization of an

automated ESP32 electric energy consumption meter. The architecture for the energy

monitoring was outlined in this study. A monitoring system that is aware of both energy and

power consumption. The proposed smart home energy monitoring system is automated. To

collect sensor data and control the system, the ESP32 Microcontroller interfaced with the current

transformer and voltage sensor where prototyped and the blynk cloud was used for retrieving

sensor data to web clients.

The results of the experiments revealed that the designed energy monitoring system can

successfully monitor voltage, current, active power, and temperature. electricity consumption

In today's smart world environment, the coexistence of many networks with different

characteristics and energy efficiency is at the top of the list of hurdles to the realization of an

automated ESP32 electric energy consumption meter. The architecture for the energy

monitoring was outlined in this study. A monitoring system that is aware of both energy and

power consumption. The proposed smart home energy monitoring system is automated. To

collect sensor data and control the system, the ESP32 Microcontroller interfaced with the current

transformer and voltage sensor where prototyped and the blynk cloud was used for retrieving

sensor data to web clients.

The results of the experiments revealed that the designed energy monitoring system can

successfully monitor voltage, current, active power, and temperature

In today's smart world environment, the coexistence of many networks with different characteristics and energy efficiency is at the top of the list of hurdles to the realization of an automated ESP32 electric energy consumption meter. The architecture for the energy monitoring was outlined in this study. A monitoring system that is aware of both energy and power consumption. The proposed smart home energy monitoring system is automated. To collect sensor data and control the system, the ESP32 Microcontroller interfaced with the current transformer and voltage sensor where prototyped and the blynk cloud was used for retrieving sensor data to web clients.

The results of the experiments revealed that the designed energy monitoring system can successfully monitor voltage, current, active power, and temperature. Electricity consumption over time. This work could be improved in future research discover more about the energy usage profile and how to detects which appliance is in use automatically using machine learning techniques.

**BIBLIOGRAPHY**

* Benzi F., Anglani N., Bassi E., Frosini L., (2011). Electricity Smart Meters Interfacing the Households, IEEE Transactions on Industrial Electronics, 58(10), 4487-4494. https://doi.org/10.1109/TIE.2011.2107713
* Rashed Mohassel R., Fung, A., Mohammadi, F., & Raahemifar K., (2014). A survey on Advanced Metering Infrastructure, International Journal of Electrical Power & Energy Systems, 63, 473-484. https://doi.org/10.1016/j.ijepes.2014.06.025