Table of Contents:

Chapter 1: Introduction

Components Required:

- **ESP32**

Fig. 1: ESP32 board

ESP32(Fig.1) is a powerful, low-cost, and highly integrated microcontroller developed by Espress if Systems. It also has built-in Wi- Fi and Bluetooth connectivity, making it ideal for IoT applications. The Wi-Fi connectivity supports both 2.4GHz and 5GHz frequency bands, while the Bluetooth connectivity supports classic Bluetooth and Bluetooth Low Energy (BLE) protocols [1].

The ESP32 microcontroller comes with a range of interfaces, including SPI, I2C, UART, I2S, PWM, and ADC. It also has a built-

in hall sensor, temperature sensor, and touch sensor, making it suitable for a wide range of applications.

The ESP32 microcontroller supports several programming languages, including C, C++, and Micro Python. It also has an integrated development environment (IDE) called the ESP-IDF, which provides a set of software development tools for developing applications [4].

- **ACS712 Sensor**

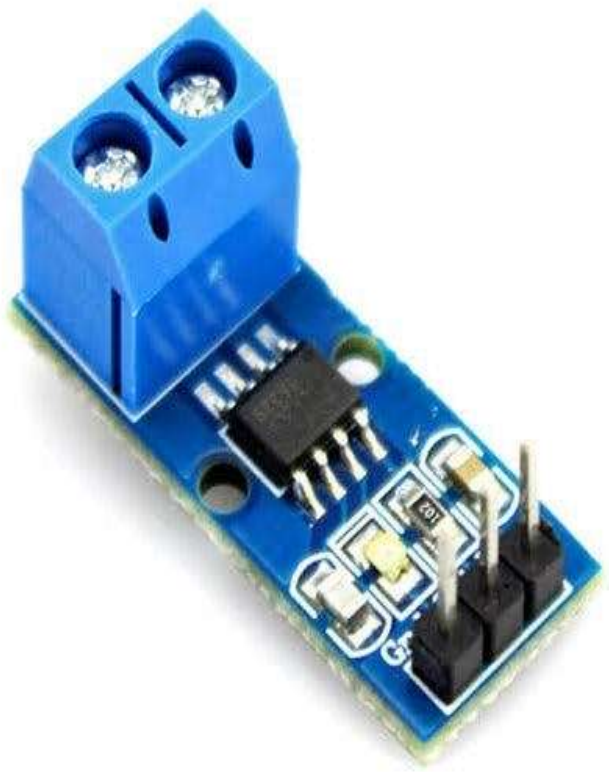


Fig. 2:- ACS712[2]

The ACS712 is a fully integrated, hall effect-based linear current sensor with 2.1kVRMS voltage isolation and a integrated low-resistance current conductor. Technical terms aside, it's simply put forth as a current sensor that uses its conductor to calculate and measure the amount of current applied [2].

- **ZMPT101B**

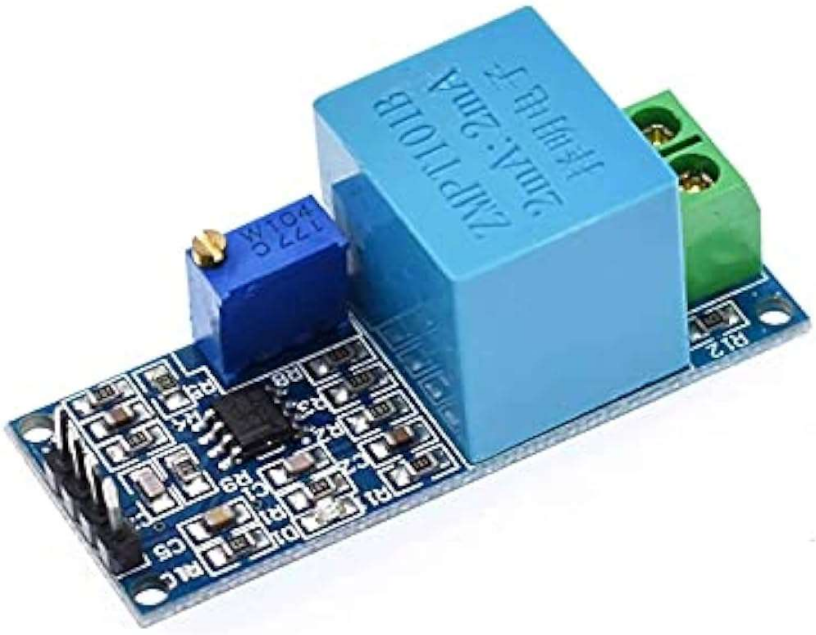


Fig. 3:- ZMPT101B[3]

The ZMPT101B is a voltage sensor module that is commonly used to measure alternating current (AC) voltage. It is specifically designed for low-voltage and low-power applications. This sensor module is widely used in electronics and microcontroller-based projects for monitoring and measuring AC voltage.

When using the ZMPT101B voltage sensor, you typically connect the AC voltage you want to measure to the input pins of the module. The output of the module can then be connected to an analog input pin on a microcontroller or other measurement device. By reading the output voltage and knowing the sensor's characteristics, you can calculate and monitor the AC voltage accurately.

It's important to note that while the ZMPT101B is useful for measuring AC voltage, it should not be used for high-voltage applications, and safety precautions should always be taken when working with electrical circuits. Additionally, the specific pin configuration and characteristics may vary depending on the manufacturer, so it's important to consult the datasheet or documentation provided with the module you are using.

Chapter 2: Connection Diagram

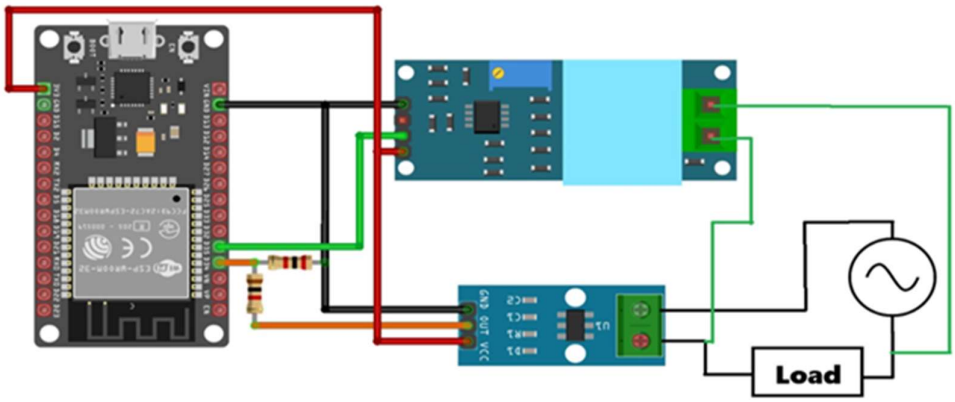
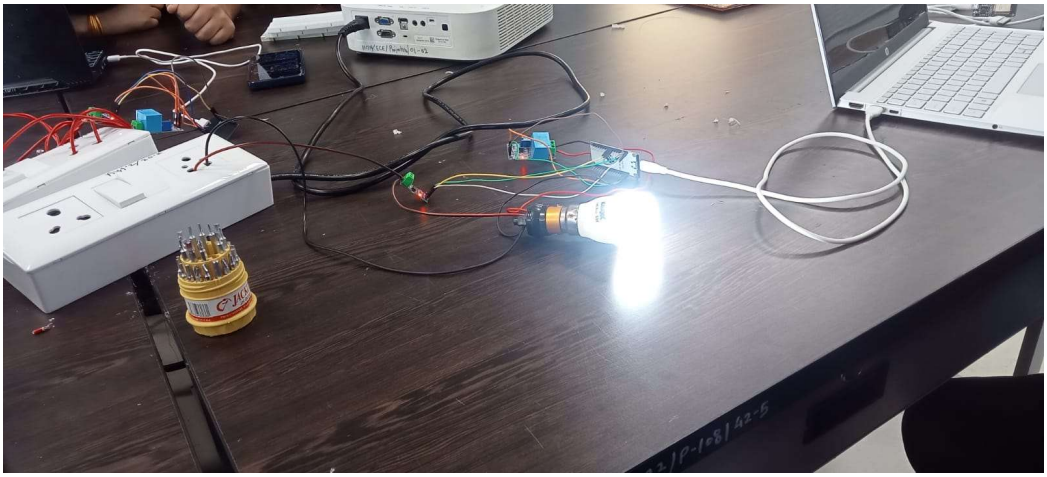


Fig 4. Connection Diagram [7]



Chapter 3: Code

```
#include "ACS712.h"
#include <WiFi.h>

const char* ssid = "OPPO A96";
const char* password = "123456789";

String header;

WiFiServer server(45);

ACS712  ACS(25, 5.0, 1023, 100);

void setup()
{
  Serial.begin(115200);
  while (!Serial);
  Serial.println(__FILE__);
  Serial.print("ACS712_LIB_VERSION: ");
  Serial.println(ACS712_LIB_VERSION);

  ACS.autoMidPoint();
  Serial.print("MidPoint: ");
  Serial.println(ACS.getMidPoint());
```

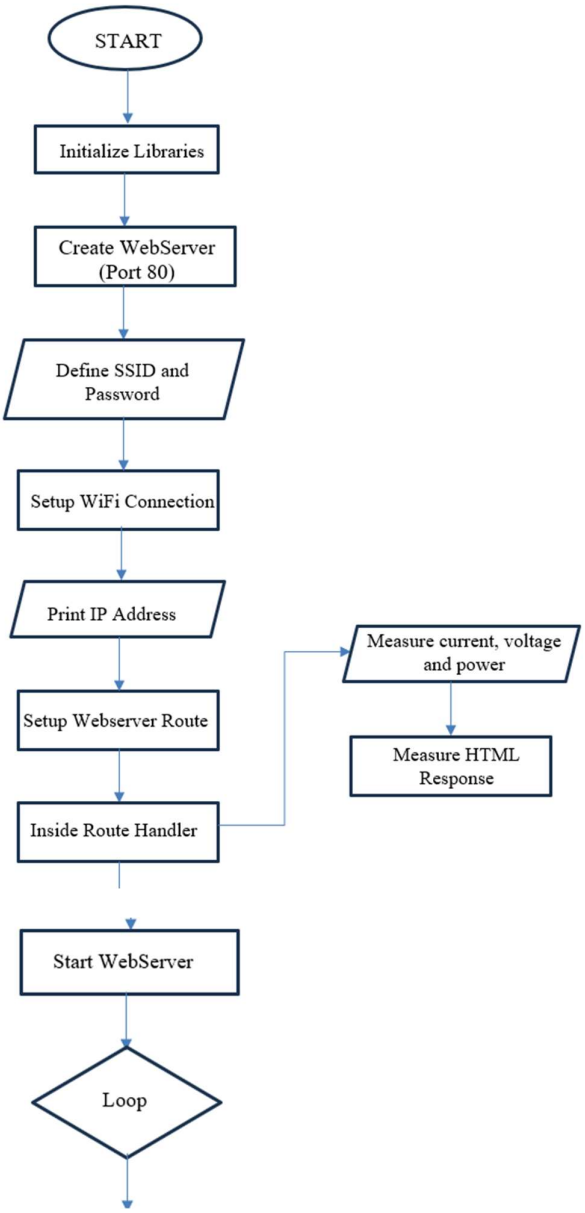
```
  Serial.print("Noise mV: ");
  Serial.println(ACS.getNoisemV());

  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
  delay(500);
  Serial.print(".");
    }

  //Serial.println("");
  //Serial.println("WiFi connected.");
  //Serial.println("IP address: ");
  //Serial.println(WiFi.localIP());
  server.begin();
  pinMode(26, INPUT);
}


void loop()
{
  float average = 0;
  uint32_t start = millis();
  for (int i = 0; i < 100; i++)
```

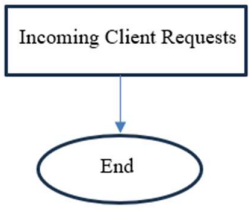
```
{
  //  select sppropriate function
  //  average += ACS.mA_AC_sampling();
  average += ACS.mA_AC();
 // Serial.println(ACS.mA_AC());
 // delay(1000);
}
float mA = average / 100.0;
mA=mA/1000;
// uint32_t duration = millis() - start;

Serial.print("  Ampere current: ");
Serial.println(mA);

 // Read analog input
 float voltage=analogRead(26)/1000;
 Serial.print(" volts ");
 Serial.println(voltage);
 float power=(mA*voltage);
 Serial.println(power);

   WiFiClient client = server.available();

if (client) {
  currentTime = millis();
```

```
    previousTime = currentTime;
    Serial.println("New Client.");
    String currentLine = "";
    while (client.connected() && currentTime - previousTime <= timeoutTime) {
      currentTime = millis();
      if (client.available()) {
        char c = client.read();
        Serial.write(c);
        header += c;
        if (c == '\n') {

          if (currentLine.length() == 0) {

            client.println("HTTP/1.1 200 OK");
            client.println("Content-type:text/html");
            client.println("Connection: close");
            client.println();

            client.println("<!DOCTYPE html><html>");
            client.println("<head><meta name=\"viewport\" content=\"width=device-width, initial-scale=1\">");
            client.println("<link rel=\"icon\" href=\"data:,\">");

            client.println("<body><h1>Smart Energy Meter </h1>");
            client.println("<p>Current :" + mA + "</p>");
```

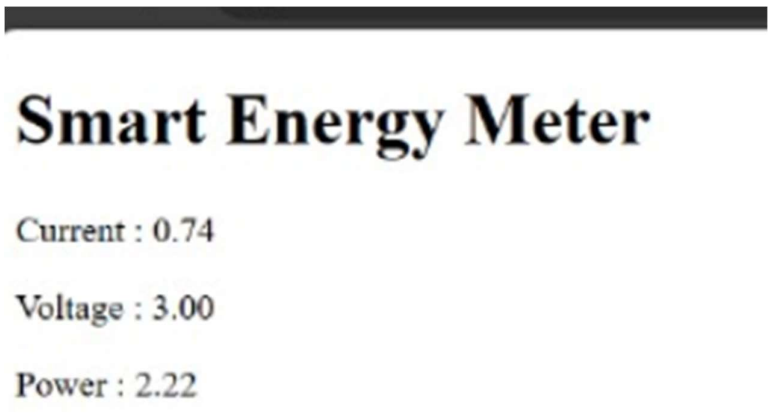
```
            client.println("<p> Voltage :" + voltage + "</p>");
            client.println("<p> Power :" + power + "</p>");
 break;
          } else {
            currentLine = "";
          }
        } else if (c != '\r') {
          currentLine += c;
        }
      }
    }
    header = "";
    client.stop();
    Serial.println("Client disconnected.");
    Serial.println("");
  }
}
```

Flow chart:


START
Initialize Libraries
Create WebServer (Port 80)
Define SSID and Password
Setup WiFi Connection
Print IP Address
Setup Webserver Route
Inside Route Handler
Measure current, voltage and power
Measure HTML Response
Start WebServer
Loop



Result:



Chapter 4: References

[1] https://how2electronics.com/iot-based-electricity-energy-meter-using-esp32-blynk/
[2] https://www.seeedstudio.com/blog/2020/02/15/acs712-current-sensor-features-how-it-works-arduino-guide/https://en.wikipedia.org/wiki/ESP32
[3] https://robu.in/product/ac-voltage-sensor-module-zmpt101b-single-phase/
[4] https://en.wikipedia.org/wiki/ESP32
[5] https://randomnerdtutorials.com/getting-started-with-esp32/
[6] https://chat.openai.com/
[7] www.wokwi.com