Experiment No-5

Subject-Computer Laboratory II-Industrial Internet of Things

Class-BE AI &DS

Aim: Write a program for sending sensor data to the cloud and storing it in a database

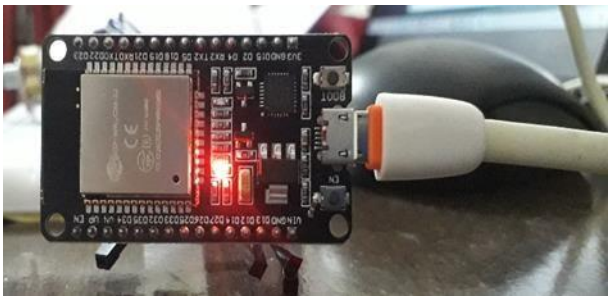
Software Requirement: Arduino IDE

Hardware Requirement: ESP-WROOM 32 board, Micro USB Data Cable, bread board, Potentiometer 10K, Male to female wires, Laptop/PC.

Theory: The program is for an ESP32-based microcontroller and is designed to create a simple web server. It reads the value from a potentiometer and displays that value on a webpage hosted by the ESP32.

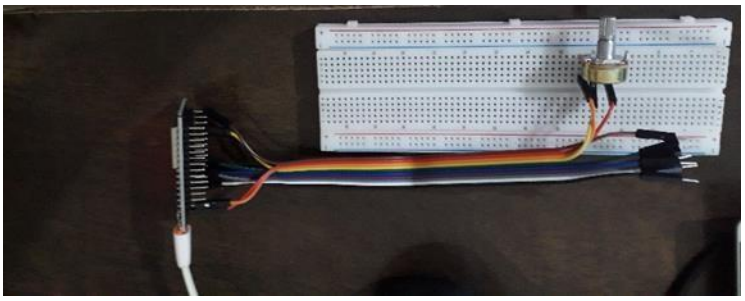
Connections:

1. First connect the data cable to the ESP 32 board, check out the notch and insert the cable in straight manner, without any tilt.

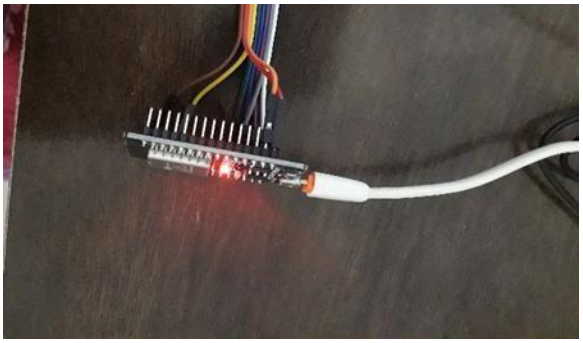


2. Connect the other end of the cable to the USB port of Laptop /PC and you should see a blue light glows.

3. Connect the potentiometer to the bread board as shown. Use the male female cables to connect the potentiometer to ESP 32.



4.The middle pin (yellow) of potentiometer is connected to GPIO 34, the red pin is connected to 3.3 V and the orange pin is connected to ground.



3.Open Arduino IDE, then go to File→New Sketch

4.And copy the given code in new sketch and upload it.

5. Make sure in Tools→Manage libraries→Library manager→type WebServer, and search for WebServer_ESP32_ENC library and install it.

6. When you run this code on your ESP32, it sets up a web server that you can access from a web browser.

7. Visiting the ESP32's IP address will display an HTML page showing the current potentiometer value.

8. For example, Potentiometer value:2233.

Procedure:

Explanation of code:

1. Include Libraries:

- **#include <WiFi.h>:** This library allows you to use the ESP32's Wi-Fi functionality.

- **#include <WebServer.h>:** This library enables you to create a web server on the ESP32.

2. Global Variables:

- **`ssid` and `password`:** These variables store the SSID (network name) and password of your Wi-Fi network.
- **`potPin`:** This variable stores the GPIO pin number to which the potentiometer is connected (pin 34 in this case).
- **`potValue`:** This variable is used to store the potentiometer reading.

3. WebServer Object:

- **`WebServer server(80)`:** An instance of the `WebServer` class is created on port 80. This server will handle incoming HTTP requests.

4. `handleRoot()` Function:

- This function is a callback that gets executed when someone accesses the root ("/") URL of the web server.
- It generates an HTML page with a header and displays the current potentiometer value on the webpage.

5. `setup()` Function:

- **`Serial.begin(115200)`:** Initializes serial communication for debugging purposes.
- **`pinMode(potPin, INPUT)`:** Sets the `potPin` as an input to read the potentiometer value.

Connecting to Wi-Fi:

- **`WiFi.begin(ssid, password) `:** Initiates a connection to the Wi-Fi network using the SSID and password.
- A `while` loop waits until the ESP32 successfully connects to the Wi-Fi network (`WL_CONNECTED` status).
- After successful connection, it prints the ESP32's local IP address to the serial monitor.

Setting up the root URL handler:

- **`server.on("/", HTTP_GET, handleRoot)`:** Configures the web server to call the `handleRoot()` function when a client requests the root URL ("/") using the HTTP GET method.

Starting the web server:

- **`server.begin()`:** Begins listening for incoming HTTP requests.

6. `loop()` Function:

- **`analogRead(potPin)`:** Reads the analog voltage at the `potPin` and stores it in the `potValue` variable.
- **`server.handleClient()`:** Handles incoming HTTP requests, including the root ("/") URL request, by calling the `handleRoot()` function.
- **`delay(100)`:** Introduces a short delay in the loop to avoid excessive server processing. Adjust the delay as needed.

Conclusion: Thus we have performed this experiment by using the ESP32 as a web server and reading an analog sensor's value to display on a webpage.

Program Code

```
#include <WiFi.h>
#include <WebServer.h>
const char *ssid = "AndroidAP1F94";
const char *password = "shilpa19";
const int potPin = 34;
int potValue = 0;
WebServer server(80);
void handleRoot() {
  String html = "<html><body>";
  html += "<h1>Potentiometer Value:</h1>";
  html += "<p>" + String(potValue) + "</p>";
  html += "</body></html>";
  server.send(200, "text/html", html);
}
void setup() {
  Serial.begin(115200);
  pinMode(potPin, INPUT);
  // Connect to Wi-Fi
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi...");
  }
```

```
  Serial.println("Connected to WiFi");
   Serial.println("");
    Serial.println("WiFi connected");
    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());
  // Handle root URL
  server.on("/", HTTP_GET, handleRoot);
  // Start server
  server.begin();
}
void loop() {
  // Read potentiometer value
  potValue = analogRead(potPin);
  server.handleClient(); // Handle incoming HTTP requests
  delay(100); // Adjust delay as needed
}
```

Output