**Pratice Real Hardware ESP32 (LED, WI-FI,** **Temperature Sensor)**

*Damar Lintang Priyatama*

*Fakultas Vokasi, Universitas Brawijaya*

*Email:* [*damarpriya138@gmail.com*](mailto:damarpriya138@gmail.com)

**Abstract**

This project demonstrates the hands-on experience of working with real ESP32 hardware, integrating essential components such as an LED, Wi-Fi, and a temperature sensor. The goal is to explore the practical aspects of ESP32 programming, from setting up the device and connecting it to Wi-Fi, to reading data from a DHT sensor and sending it to a remote server using HTTP POST requests. The process includes configuring the hardware, uploading code via PlatformIO, troubleshooting connections, and validating data transmission to a database. This project provides a comprehensive approach to understanding ESP32's capabilities for Internet of Things (IoT) applications.

*Keywords—* *IoT, ESP32, LED, Wi-Fi, Temperature Sensor.*

**1. Introduction**

**1.1 Background**

The ESP32 is a powerful microcontroller commonly used for IoT applications due to its Wi-Fi and Bluetooth capabilities. In this practice, the focus is on integrating a simple system consisting of an LED, a Wi-Fi module, and a temperature sensor to create an IoT-enabled device. The system sends temperature and humidity data to a server via HTTP POST, showcasing the versatility of ESP32 in real-world applications. Prior knowledge of wiring and basic ESP32 setup is leveraged in this hands-on implementation to understand the communication between the device and the internet.

**1.2 Purpose Experience**  
The main purpose of this experiment is to provide a real-world implementation using the ESP32 hardware to control an LED, connect to Wi-Fi, and interface with a DHT sensor to measure temperature and humidity. Additionally, the data collected by the sensor is sent to a remote server via HTTP requests. This practice aims to familiarize users with hardware configuration, coding, and server communication, laying the foundation for more complex IoT applications.

This project involves working with real ESP32 hardware, ensuring that the device is properly recognized by the computer and is able to establish a connection to a local Wi-Fi network. The practical experience includes setting up the platform (PlatformIO), coding the ESP32 to blink LEDs, connecting to Wi-Fi, reading sensor data, and sending that data to a remote server using HTTP. Through this process, the user gains hands-on experience with both hardware configuration and software development for IoT applications.

**2. Methodology**

**2.1 Tools & Materials**

-Laptop Asus

-Vscode for coding

-laravel 11

-Postman

-Xampp

-Ngrok

-Composer

-ESP32 Board

-LED

-DHT22 Temperature

**2.2 Implementation Steps**

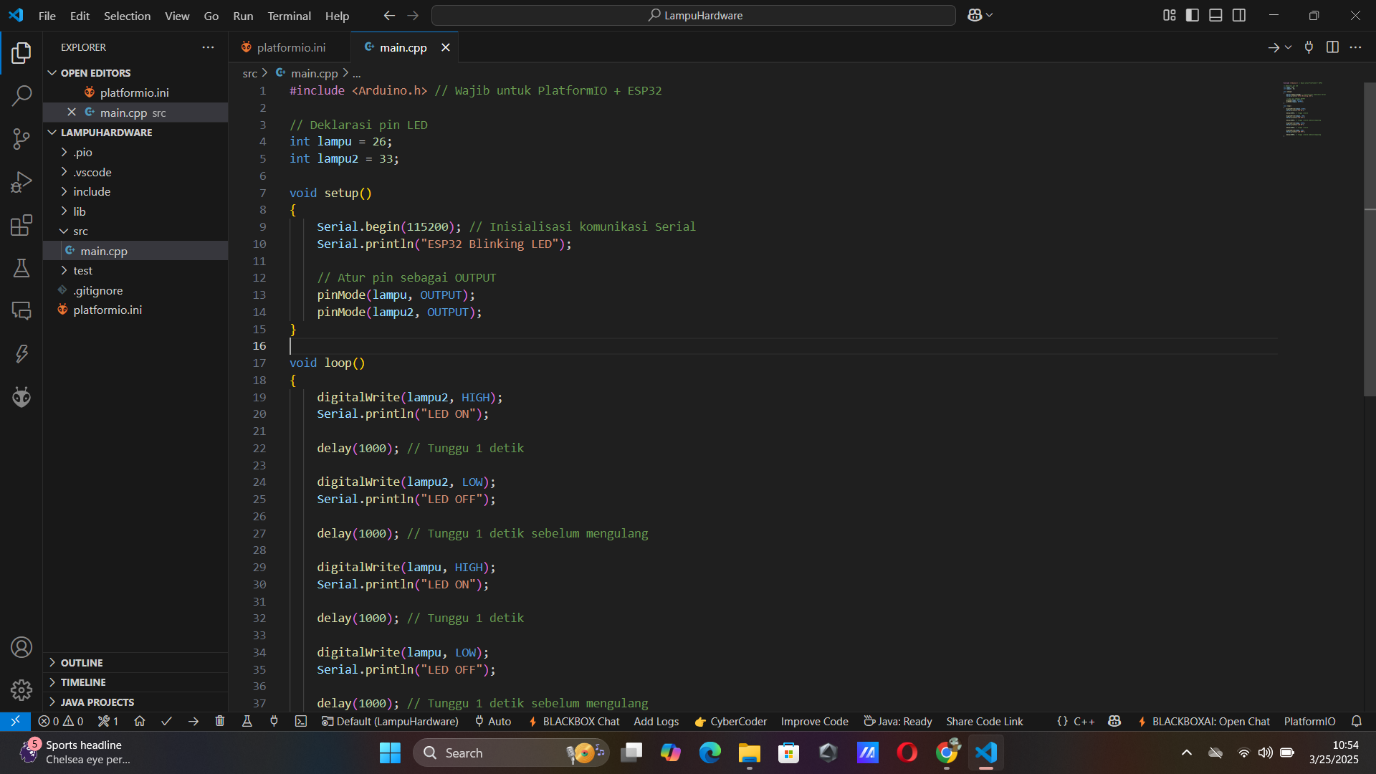
- Ensure the ESP32 is correctly connected to the computer via USB and recognized in the Device Manager.

- Download and install the necessary drivers (Silicon Labs CP210x) if the device is not detected.

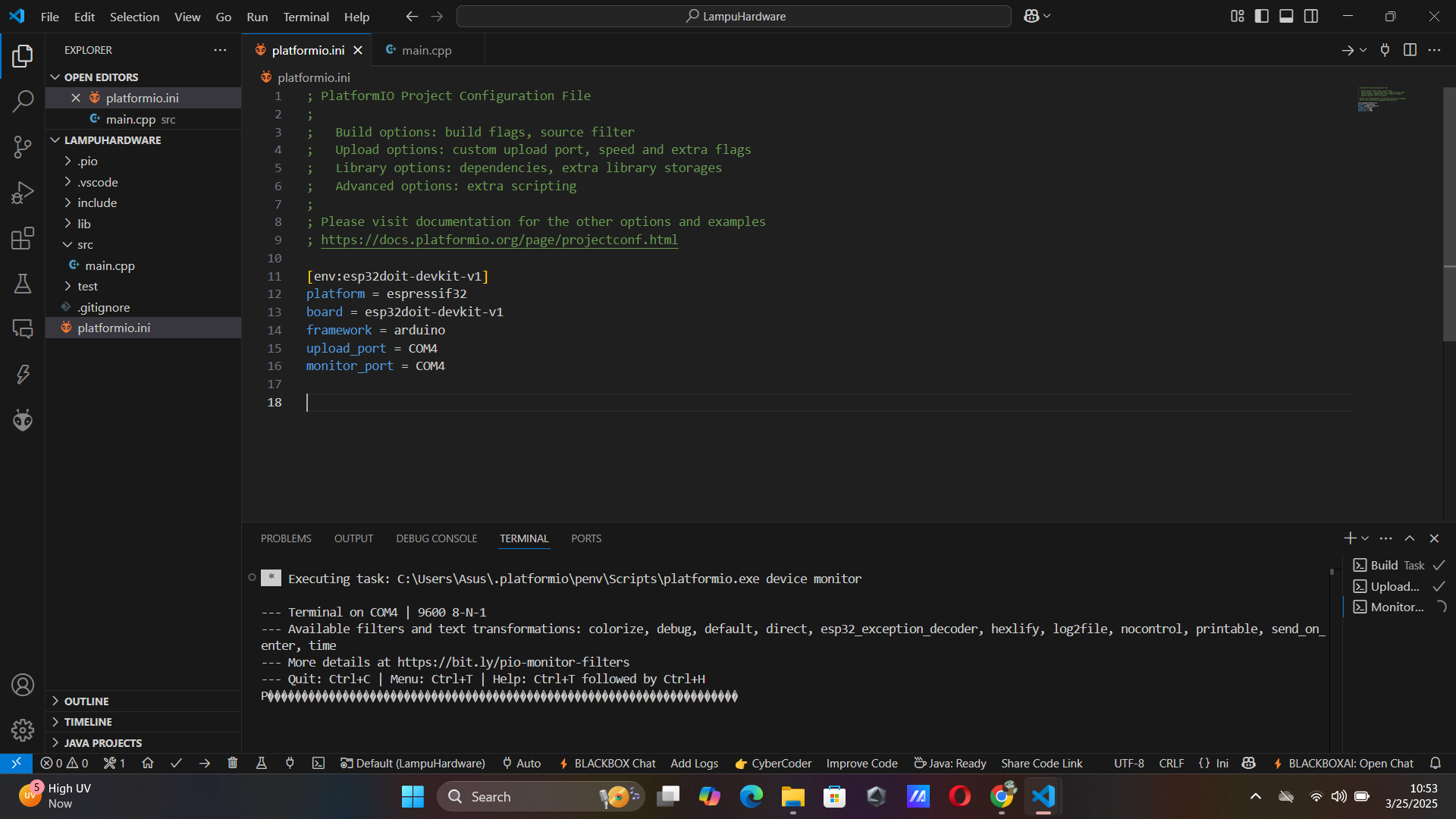
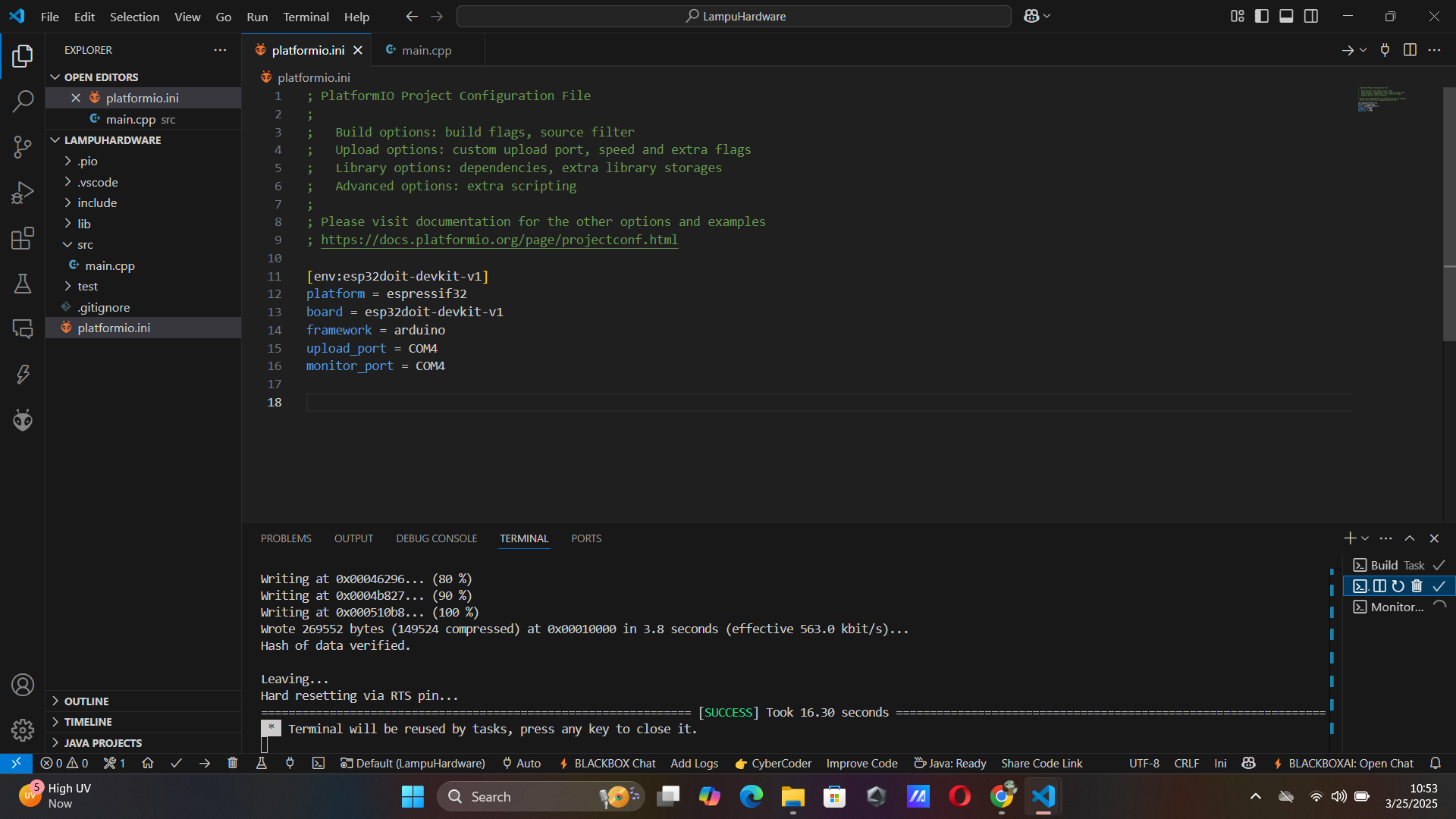
- Open Visual Studio Code (VSCode) and configure PlatformIO.

- Set up the project with the ESP32 board and make sure the correct COM port is selected.

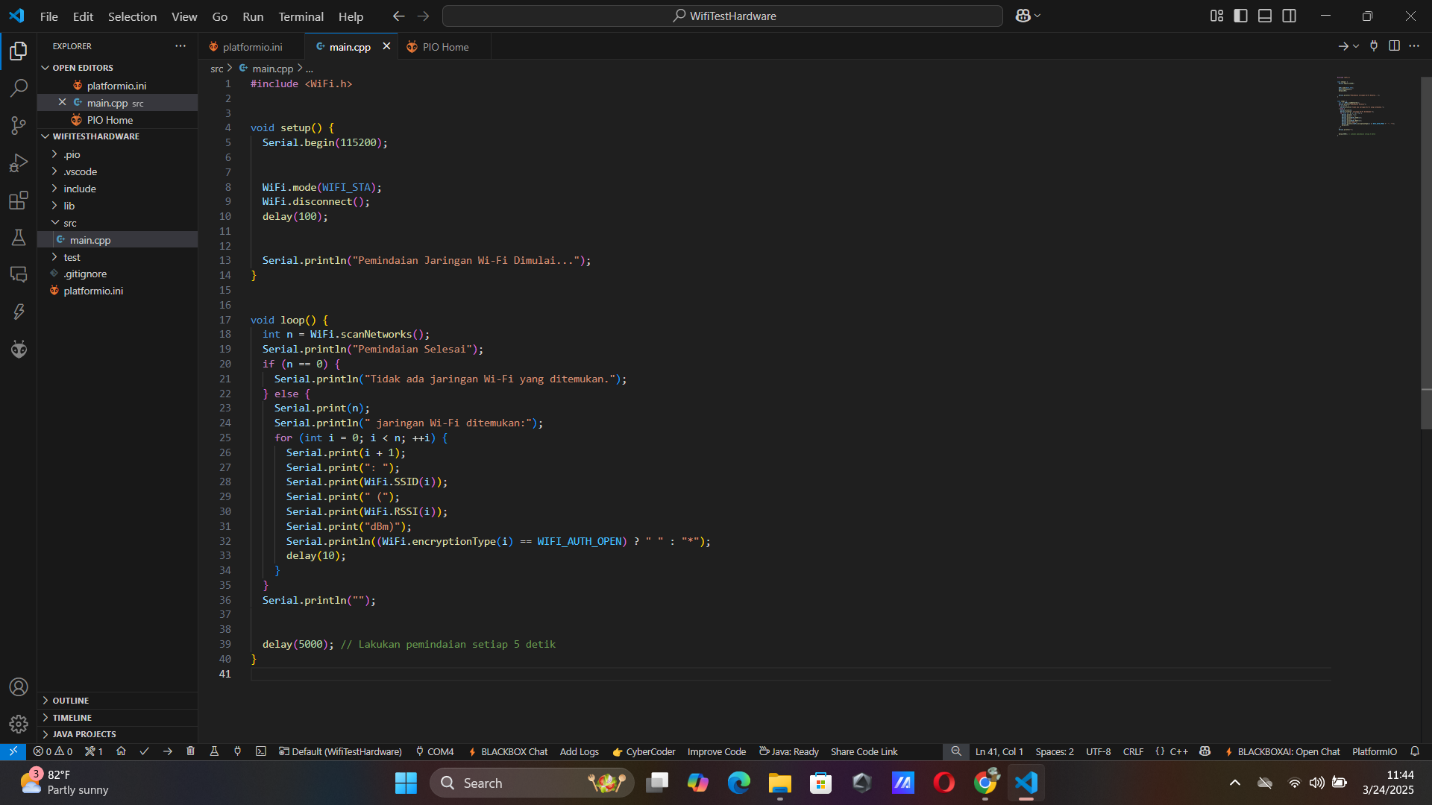
* Steps to Hardware LED

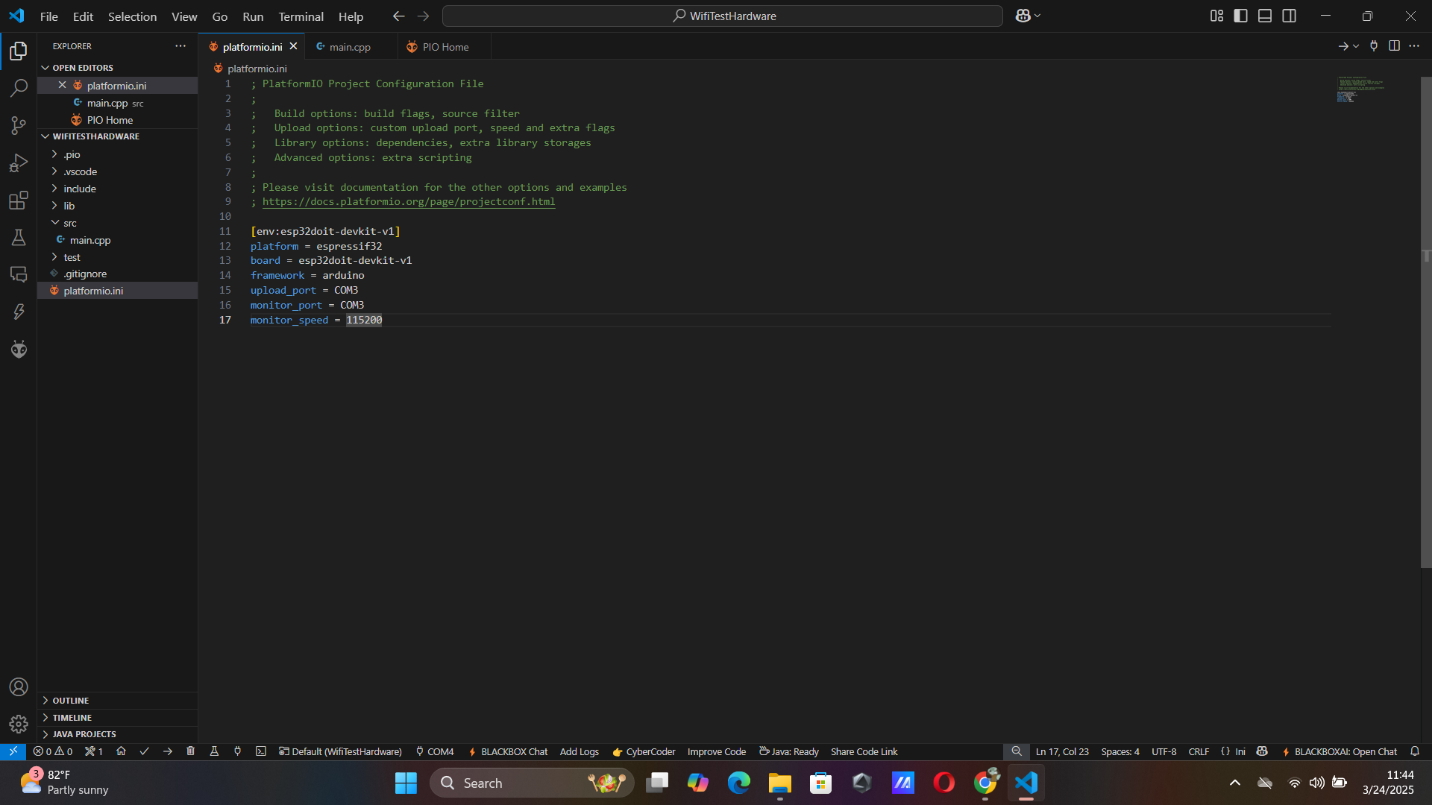
**-** Write the main.cpp code to blink LEDs using the pins. This verifies the correct setup of the hardware and basic functionality.

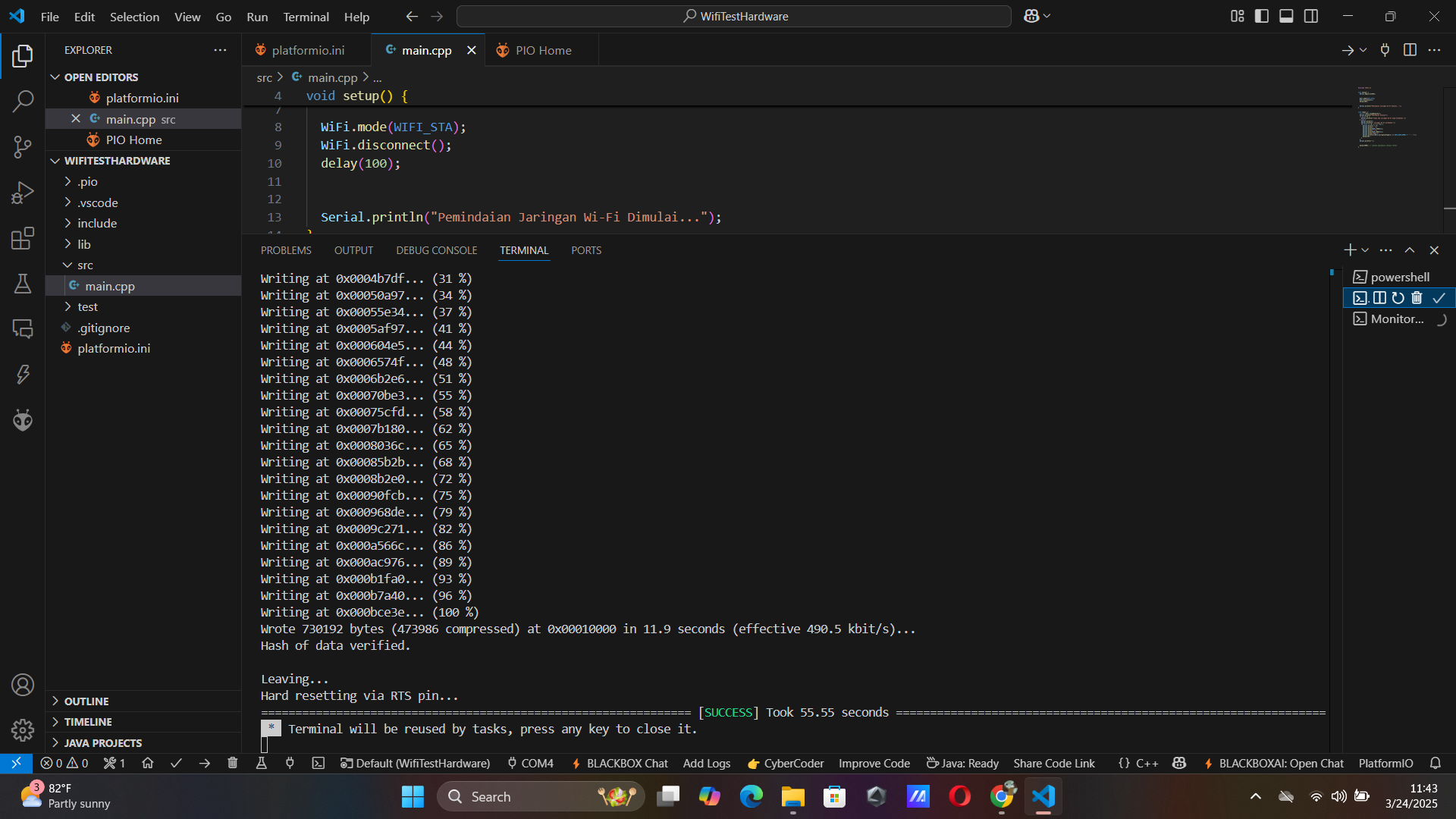
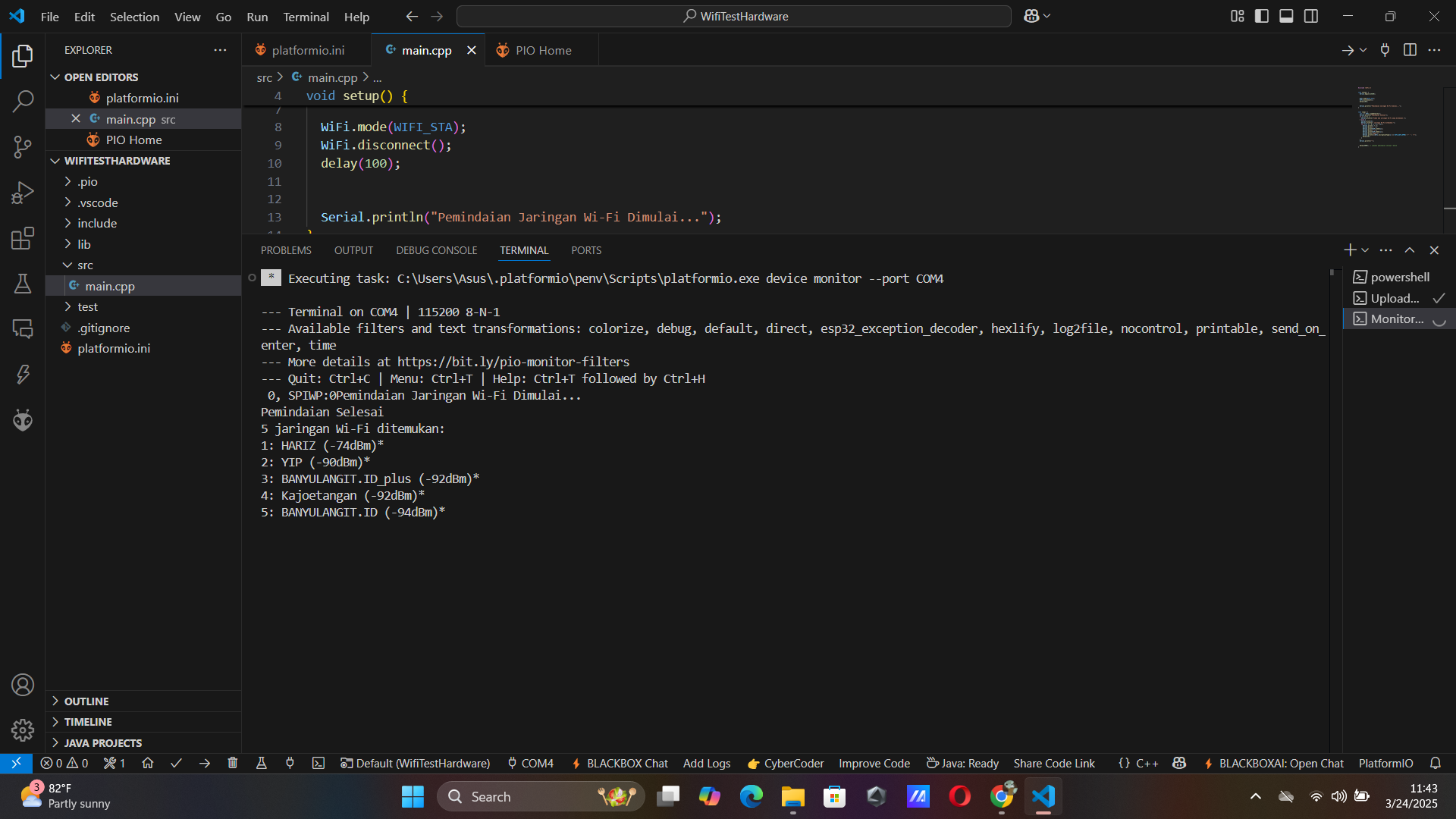
**-** Write the platformio.ini code to blink LEDs using the pins. After that do upload and click button serial monitor

****

* Steps to Hardware Wifi Connect

- Write the main.cpp code to scan for available Wi-Fi networks.

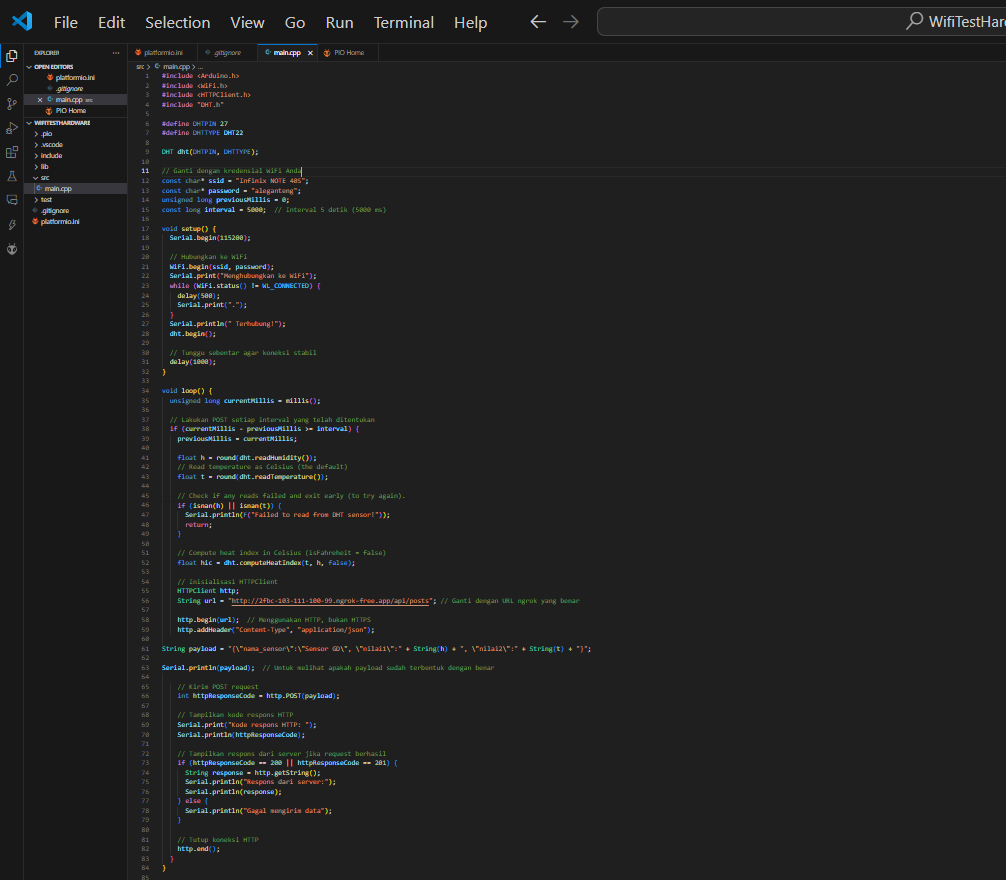
- Write the platformio.ini code to scan for available Wi-Fi networks and display the results on the Serial Monitor by upload and click button serial monitor.

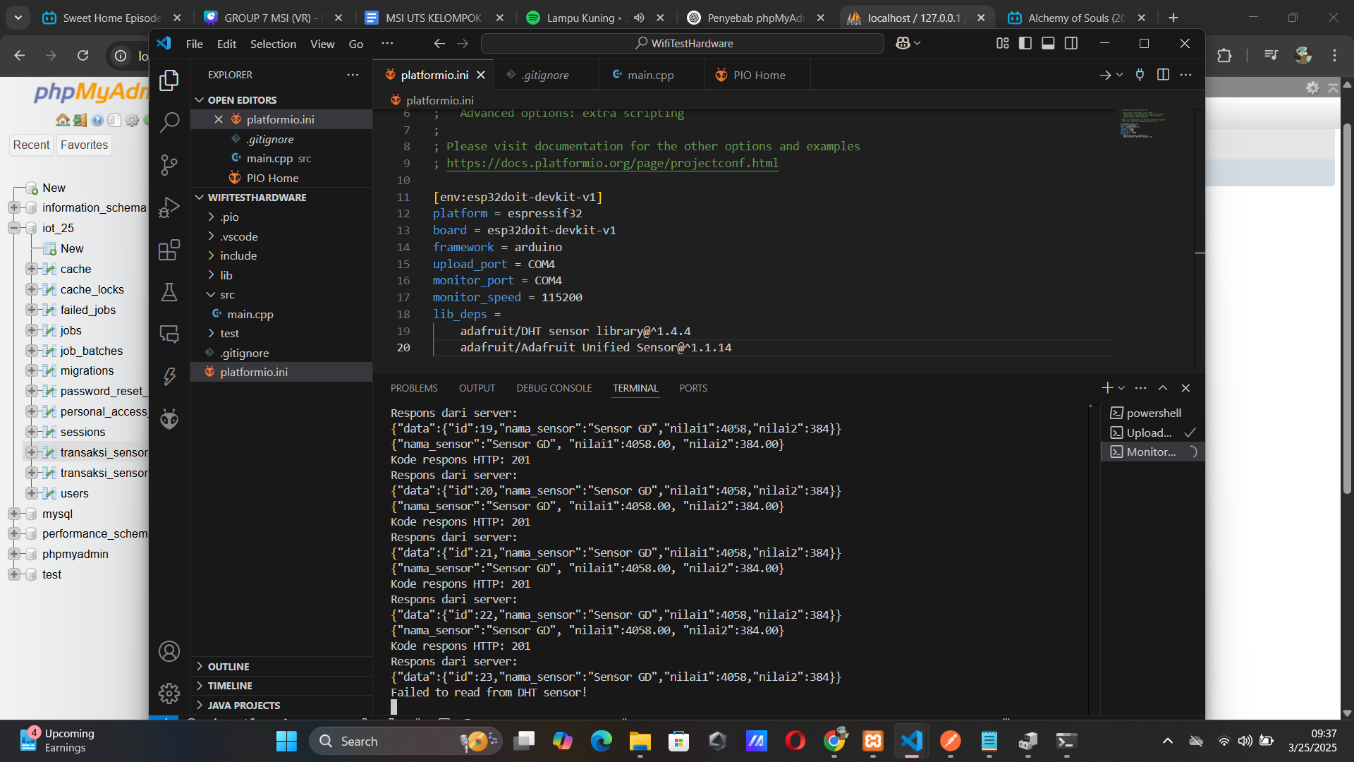


- After the result is written, u can adjust the code with the Wi-Fi credentials for local access.

* Steps to Hardware Temperature Sensor

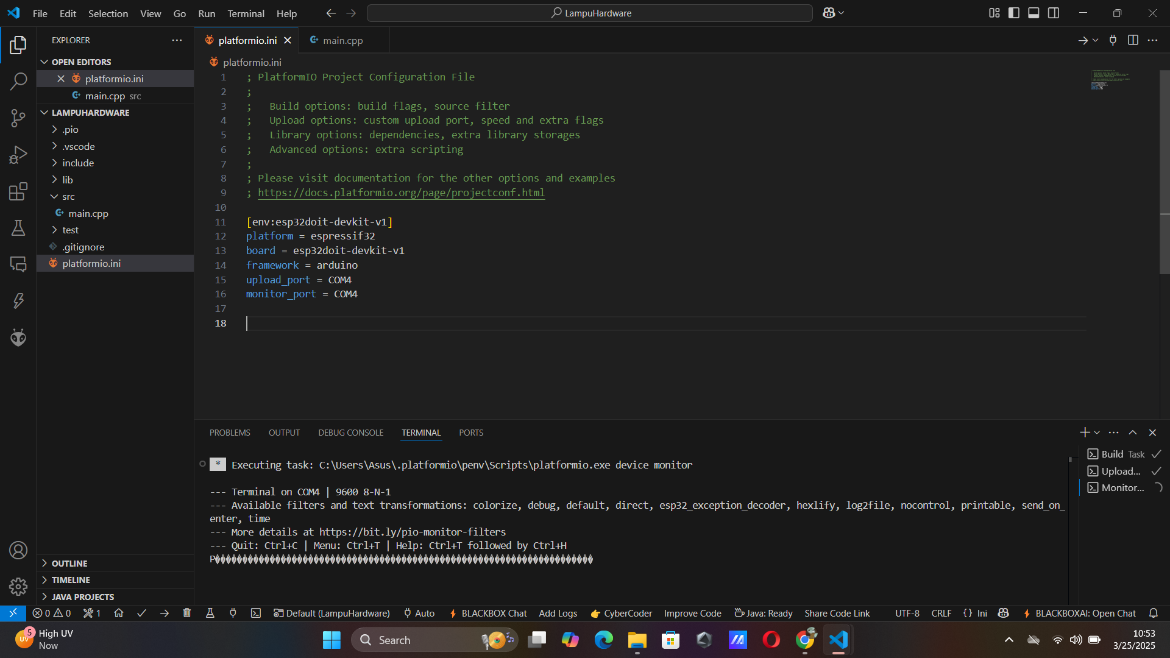
- Modify the code from wifi sensor to interface with the DHT22 sensor, collect temperature and humidity data, and display the results on the Serial Monitor.

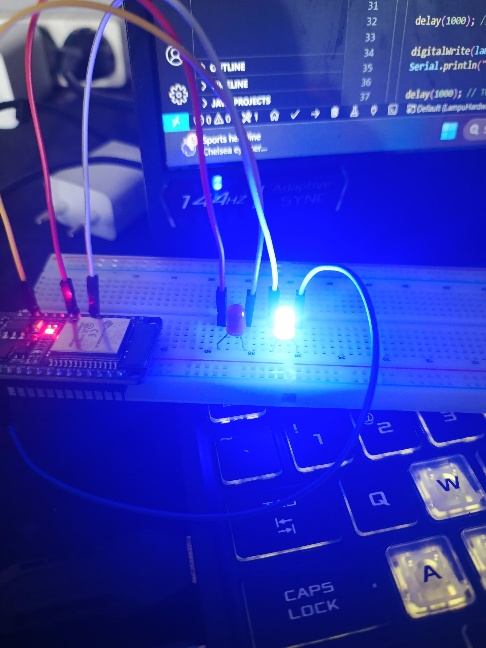
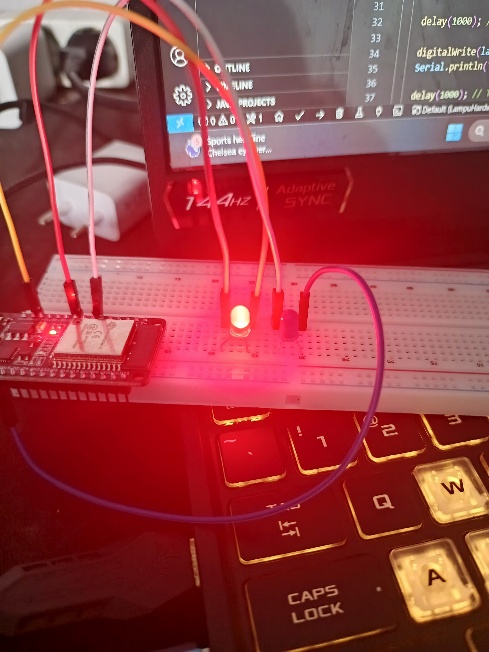
- Write the main.cpp code to scan for Temperatur Sensor.

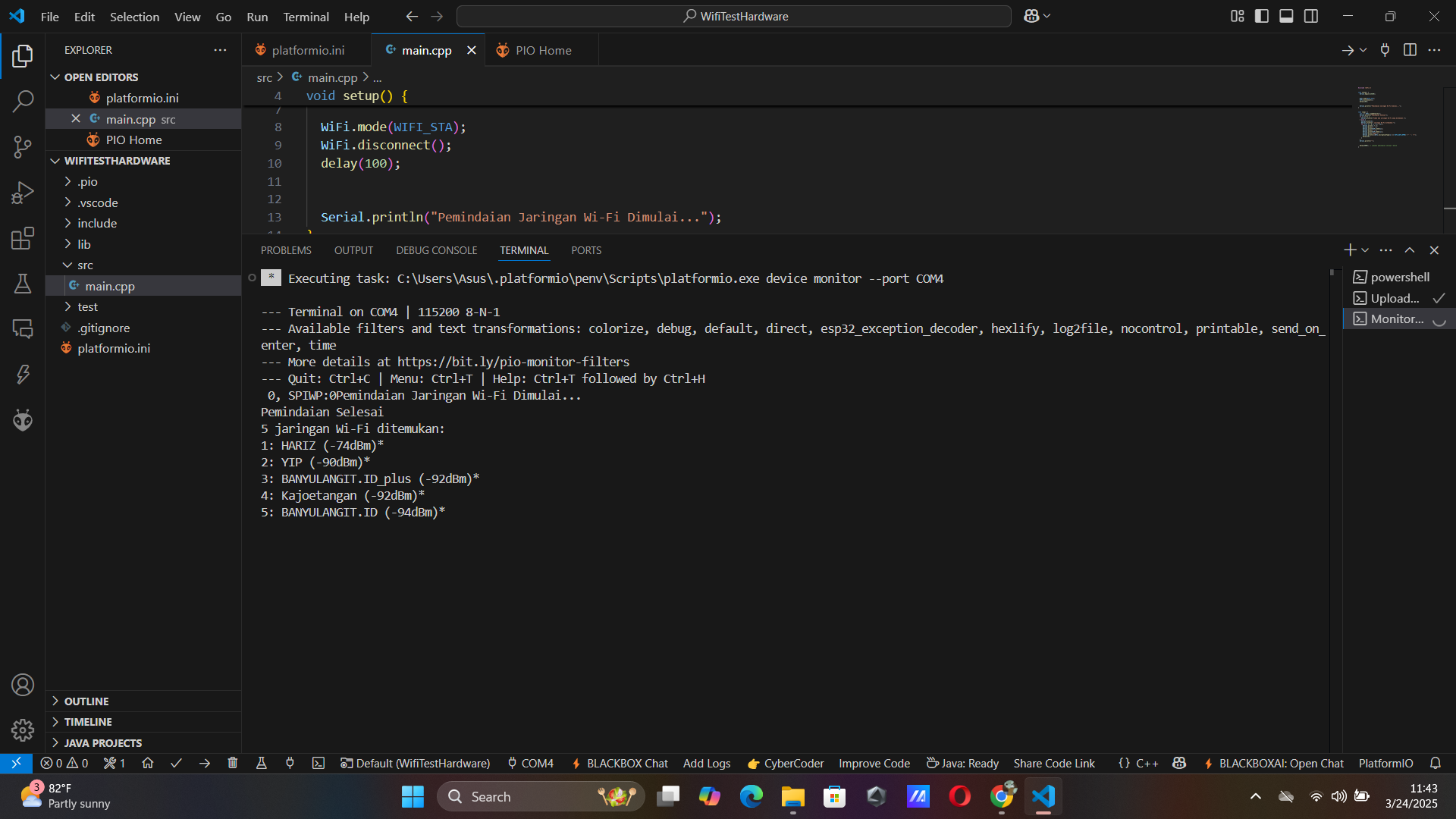
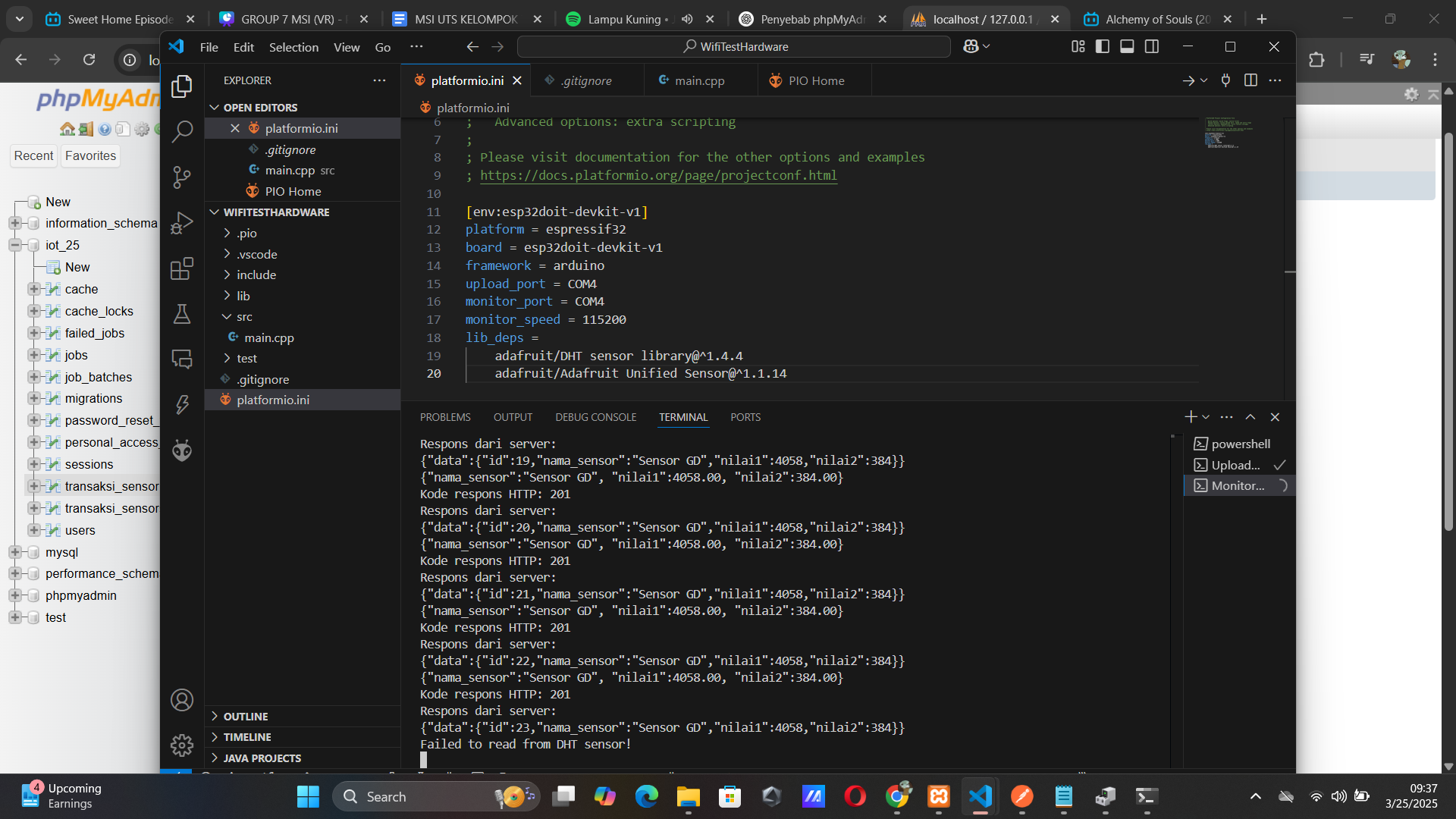
- Write the platformio.ini code to scan for Temperatur Sensor. After that u can upload and click button serial monitor also with verify that the sensor is properly reading values.

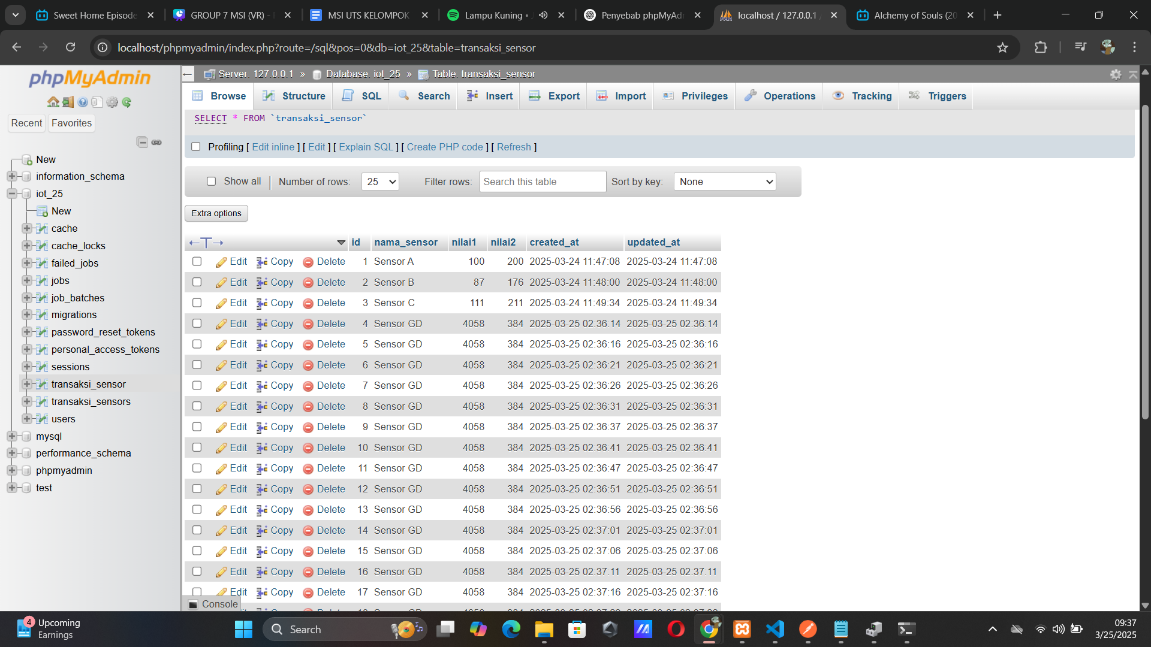
**3. Results and Discussion**

**3.1 Experimental Results**

* **Result hardware LED**

****

* **Result hardware WIFI**
* **Result hardware Temperature Sensor**

** data API from Temperature Sensor**