# LAPORAN PRAKTIKUM INTERNET OF THINGS (IoT)

# Fakultas Vokasi, Universitas Brawijaya

**Praktik Real Hardware ESP32**

**Author(s)** Muhammad Hafizh Al Furqon

*Fakultas Vokasi, Universitas Brawijaya*

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

Praktikum ini bertujuan untuk menerapkan konsep Internet of Things (IoT) pada perangkat nyata dengan menggunakan mikrokontroler ESP32. Kegiatan dimulai dengan memastikan hardware ESP32 dikenali oleh komputer melalui instalasi driver (Silicon Labs CP210x), kemudian dilanjutkan dengan pembuatan proyek di PlatformIO untuk mengunggah program ke ESP32. Praktikum ini mencakup pengendalian lampu LED, pemindaian jaringan WiFi, serta pengiriman data sensor (DHT22) ke API yang di-online-kan melalui Ngrok. Hasil praktikum menunjukkan bahwa hardware ESP32 dapat terintegrasi dengan baik ke dalam sistem IoT, mulai dari konektivitas WiFi hingga komunikasi data real-time dengan backend API.

**Keywords**—*ESP32, Real Hardware, PlatformIO, WiFi, API, Ngrok, DHT22*

**1. Introduction**

**1.1 Latar belakang**

Dalam pengembangan aplikasi IoT, penggunaan simulasi sering kali dilakukan di tahap awal pengembangan. Namun, untuk implementasi sistem yang sesungguhnya, diperlukan pengujian pada perangkat keras nyata. ESP32 merupakan salah satu mikrokontroler yang banyak digunakan karena fitur konektivitas WiFi dan kemampuan pemrograman yang fleksibel. Praktikum ini menitikberatkan pada pengujian real hardware ESP32, mulai dari instalasi driver, pengunggahan kode melalui PlatformIO, hingga implementasi aplikasi nyata seperti pengendalian LED, pemindaian jaringan WiFi, dan pengiriman data sensor ke API.

**1.2 Tujuan eksperimen**

1. Memastikan hardware ESP32 dikenali oleh komputer melalui instalasi driver (Silicon Labs CP210x).
2. Mengunggah program dasar (LED blinking dan WiFi scanning) ke ESP32 menggunakan PlatformIO.
3. Mengimplementasikan pengiriman data sensor (DHT22) ke backend API yang di-online-kan melalui Ngrok.
4. Mengintegrasikan sistem real hardware dengan API untuk aplikasi IoT secara real-time.

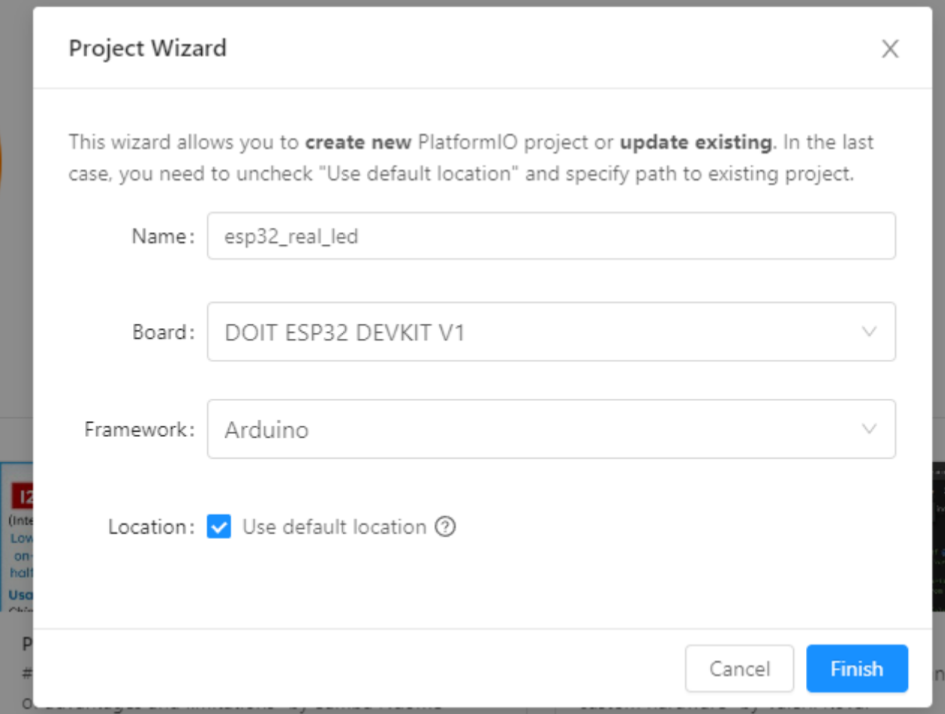
**2. Methodology (Metodologi)**

**2.1 Tools & Materials (Alat dan Bahan)**

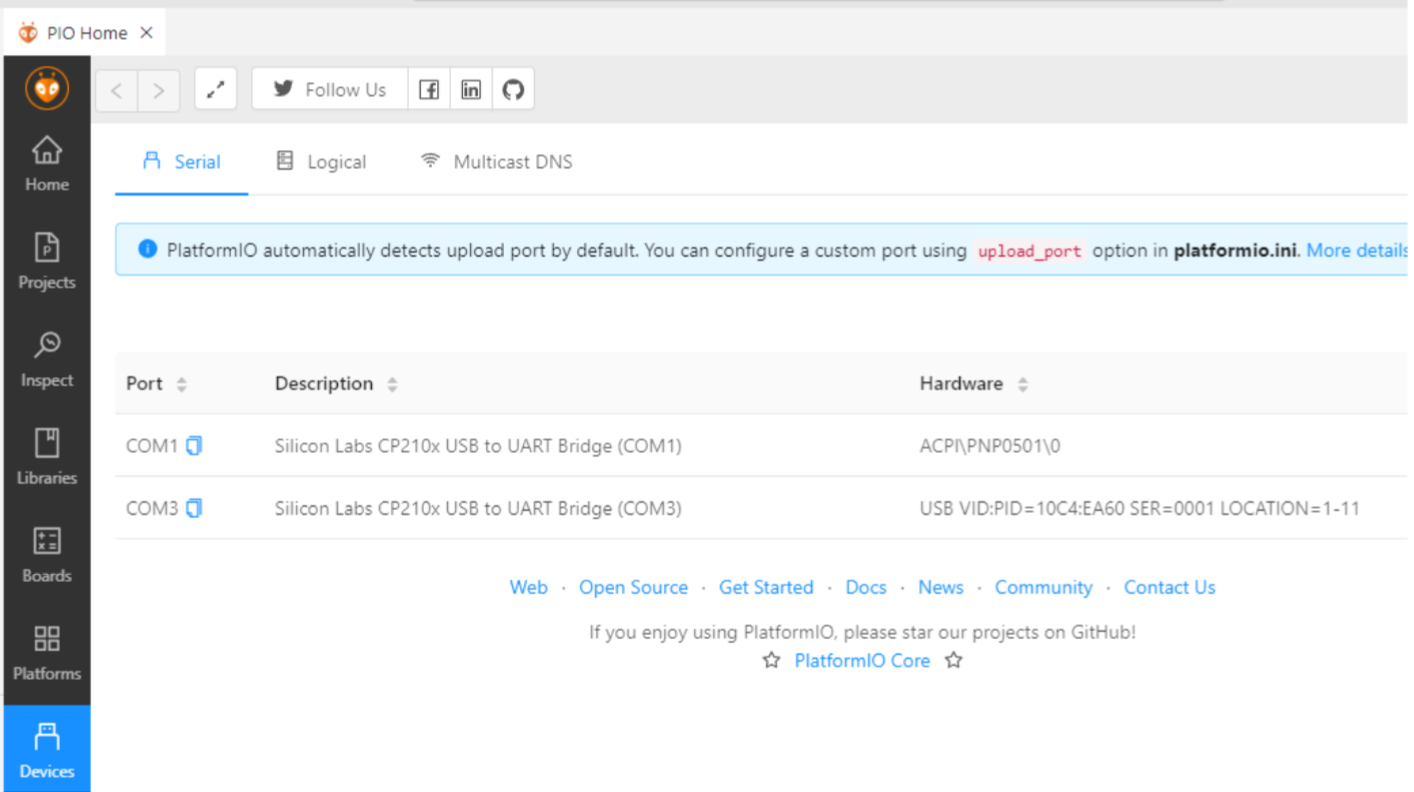
* **Perangkat Lunak:**
  + Visual Studio Code (VSCode) dengan extension PlatformIO
  + Laravel 12 (untuk backend API)
  + Ngrok
  + phpMyAdmin (untuk pengelolaan database)
  + Postman (untuk pengujian API)
* **Perangkat Keras:**
  + Mikrokontroler ESP32 (misalnya, DOIT ESP32 DEVKIT V1)
  + Kabel USB untuk koneksi ke komputer
  + Breadboard dan kabel jumper (untuk wiring sensor dan LED)
  + Sensor DHT22
* **Driver:**
  + Silicon Labs CP210x USB-to-UART Bridge Driver (dapat diunduh di: [Silicon Labs CP210x](https://www.silabs.com/developer-tools/usb-to-uart-bridge-vcp-drivers?tab=downloads))

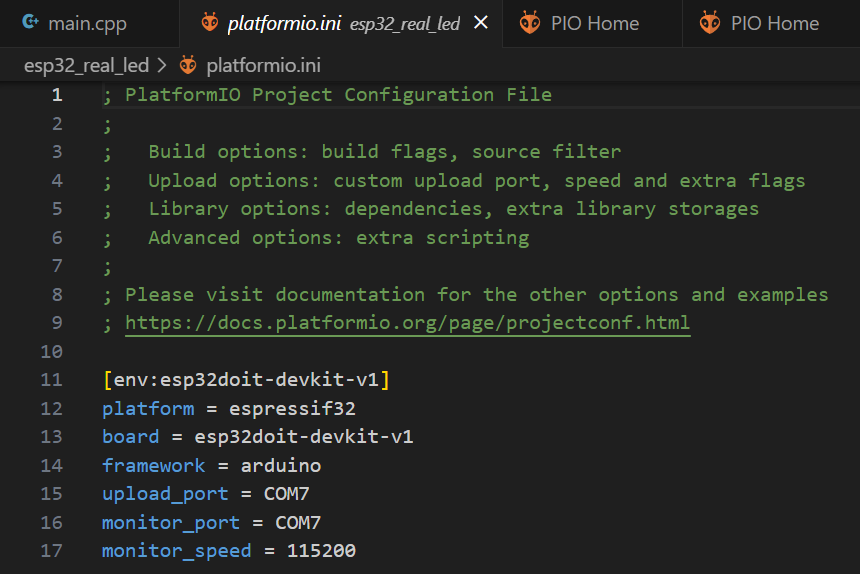
**2.2 Implementation Steps (Langkah Implementasi)**

1. Install driver esp32 terlebih dahulu dapat diunduh di: [Silicon Labs CP210x](https://www.silabs.com/developer-tools/usb-to-uart-bridge-vcp-drivers?tab=downloads)
2. Percobaan praktik real hardware 2 LED dengan membuat project baru di PlatformIO dengan ketentuan seperti berikut

\

1. Tambahkan `upload\_port` dan `monitor\_port` pada platform.ini agar program bisa men-detect lokasi port ESP32, sesuaikan COM dengan yang terdetect seperti COM3





1. Codingan src/main.cpp

#include <Arduino.h> // Wajib untuk PlatformIO + ESP32

// Deklarasi pin LED

int lampu = 26;

int lampu2 = 33;

void setup() {

Serial.begin(115200); // Inisialisasi komunikasi Serial

Serial.println("ESP32 Blinking LED");

// Atur pin sebagai OUTPUT

pinMode(lampu, OUTPUT);

pinMode(lampu2, OUTPUT);

}

void loop() {

// Nyalakan kedua LED

digitalWrite(lampu, HIGH);

digitalWrite(lampu2, HIGH);

Serial.println("LED ON");

delay(1000); // Tunggu 1 detik

// Matikan kedua LED

digitalWrite(lampu, LOW);

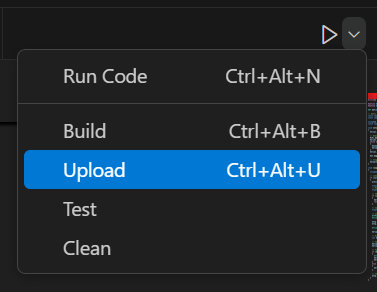
digitalWrite(lampu2, LOW);

Serial.println("LED OFF");

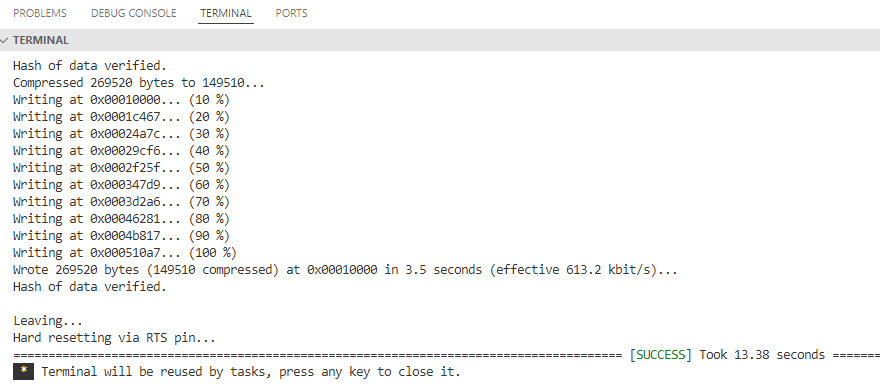
delay(1000); // Tunggu 1 detik sebelum mengulang

}

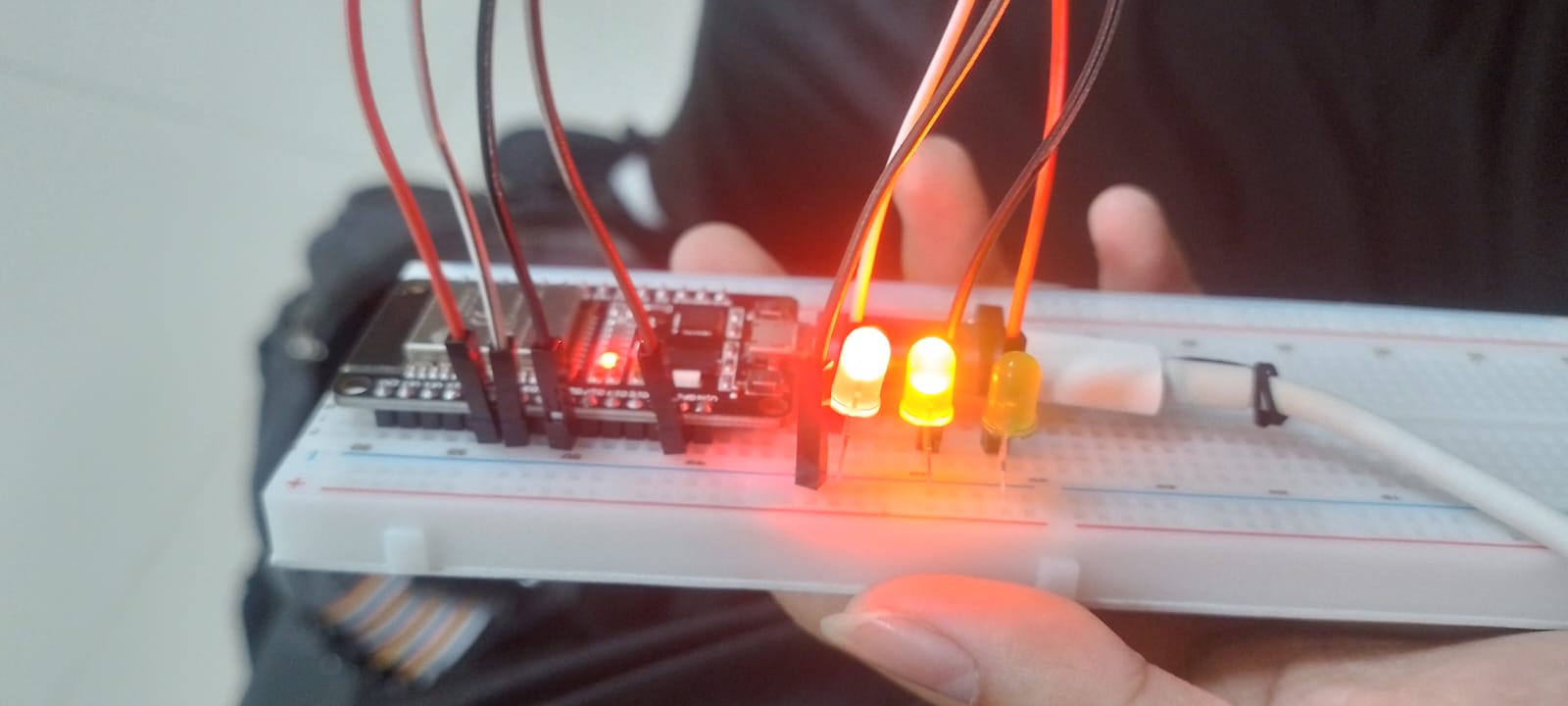
1. Sesuaikan wiring hardware dengan kodingan
2. Klik upload



Proses compiling dan upload akan berjalan dan pastikan berhasil seperti tampilan berikut



Jika seperti ini maka berhasil



1. Mengecek koneksi wifi yang dapat terhubung pada ESP32

src/main.cpp

#include <WiFi.h>

void setup()

{

Serial.begin(115200);

WiFi.mode(WIFI\_STA);

WiFi.disconnect();

delay(100);

Serial.println("Pemindaian Jaringan Wi-Fi Dimulai...");

}

void loop()

{

int n = WiFi.scanNetworks();

Serial.println("Pemindaian Selesai");

if (n == 0)

{

Serial.println("Tidak ada jaringan Wi-Fi yang ditemukan.");

}

else

{

Serial.print(n);

Serial.println(" jaringan Wi-Fi ditemukan:");

for (int i = 0; i < n; ++i)

{

Serial.print(i + 1);

Serial.print(": ");

Serial.print(WiFi.SSID(i));

Serial.print(" (");

Serial.print(WiFi.RSSI(i));

Serial.print("dBm)");

Serial.println((WiFi.encryptionType(i) == WIFI\_AUTH\_OPEN) ? " " : "\*");

delay(10);

}

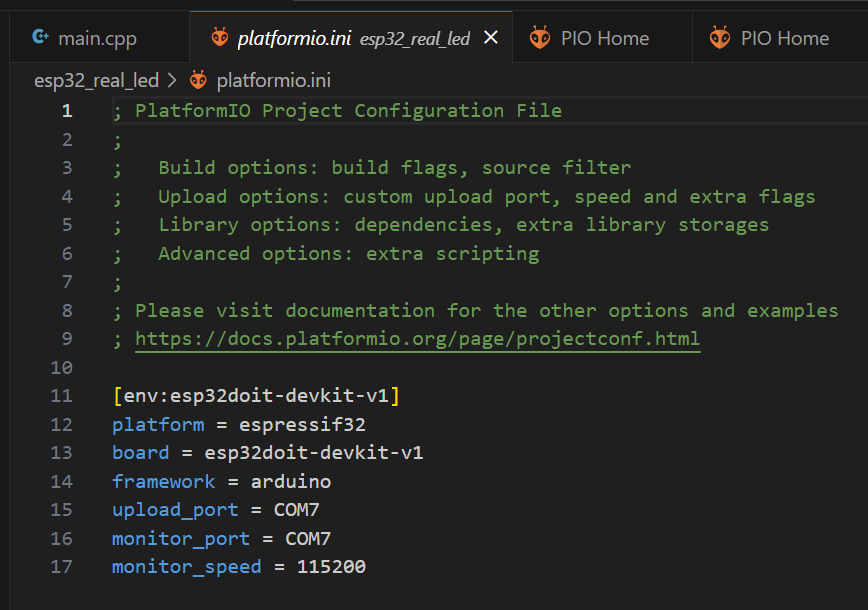
}

Serial.println("");

delay(5000); // Lakukan pemindaian setiap 5 detik

}

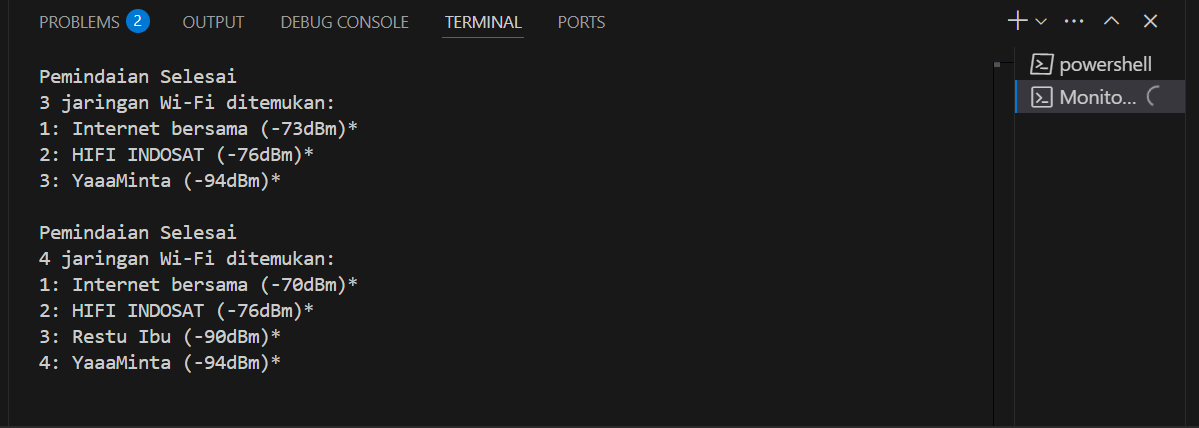
1. Tambahkan monitor\_speed di platform.ini untuk preview serial monitor



1. Klik upload
2. Klik monitoring di bagian atas untuk melihat hasil printout serial



1. Berhasil



1. ESP32 mengirim data sensor kelembapan dan suhu dari DHT11 ke database melalui API backendnya yang sebelunnya sudah dibuat
2. Buat project baru dengan ketentuan sama seperti sebelumnya
3. Jalankan API laravel  **php artisan serve --host=0.0.0.0 --port=8080**
4. Kemudian jalankan Ngrok **ngrok http --scheme=http 8080**
5. Proses wiring cable sesuai dengan src/main.cpp

#include <Arduino.h>

#include <WiFi.h>

#include <HTTPClient.h>

#include "DHT.h"

#define DHTPIN 27

#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

// Ganti dengan kredensial WiFi Anda

const char \*ssid = "Uas";

const char \*password = "1234567890";

unsigned long previousMillis = 0;

const long interval = 5000; // Interval 5 detik (5000 ms)

void setup()

{

Serial.begin(115200);

// Hubungkan ke WiFi

WiFi.begin(ssid, password);

Serial.print("Menghubungkan ke WiFi");

while (WiFi.status() != WL\_CONNECTED)

{

delay(500);

Serial.print(".");

}

Serial.println(" Terhubung!");

dht.begin();

// Tunggu sebentar agar koneksi stabil

delay(1000);

}

void loop()

{

unsigned long currentMillis = millis();

// Lakukan POST setiap interval yang telah ditentukan

if (currentMillis - previousMillis >= interval)

{

previousMillis = currentMillis;

float h = round(dht.readHumidity());

// Read temperature as Celsius (the default)

float t = round(dht.readTemperature());

// Check if any reads failed and exit early (to try again).

if (isnan(h) || isnan(t))

{

Serial.println(F("Failed to read from DHT sensor!"));

return;

}

// Compute heat index in Celsius (isFahreheit = false)

float hic = dht.computeHeatIndex(t, h, false);

// Inisialisasi HTTPClient

HTTPClient http;

String url = "http://7aa2-114-10-46-166.ngrok-free.app/api/posts"; // Ganti dengan URL ngrok yang benar

http.begin(url); // Menggunakan HTTP, bukan HTTPS

http.addHeader("Content-Type", "application/json");

String payload = "{\"nama\_sensor\":\"Sensor GD\", \"nilai1\":" + String(h) + ", \"nilai2\":" + String(t) + "}";

Serial.println(payload); // Untuk melihat apakah payload sudah terbentuk dengan benar

// Kirim POST request

int httpResponseCode = http.POST(payload);

// Tampilkan kode respons HTTP

Serial.print("Kode respons HTTP: ");

Serial.println(httpResponseCode);

// Tampilkan respons dari server jika request berhasil

if (httpResponseCode == 200 || httpResponseCode == 201)

{

String response = http.getString();

Serial.println("Respons dari server:");

Serial.println(response);

}

else

{

Serial.println("Gagal mengirim data");

}

// Tutup koneksi HTTP

http.end();

}

}

1. Ganti kredensial wifi dengan wifi pribadi dan salin link Ngrok yang sudah dirun sebelumnya, pastikan menggunakan HTTP jangan HTTPS
2. Tambahkan library sensor DHT pada platform.ini

; PlatformIO Project Configuration File

;

; Build options: build flags, source filter

; Upload options: custom upload port, speed and extra flags

; Library options: dependencies, extra library storages

; Advanced options: extra scripting

;

; Please visit documentation for the other options and examples

; https://docs.platformio.org/page/projectconf.html

[env:esp32doit-devkit-v1]

platform = espressif32

board = esp32doit-devkit-v1

framework = arduino

upload\_port = COM7

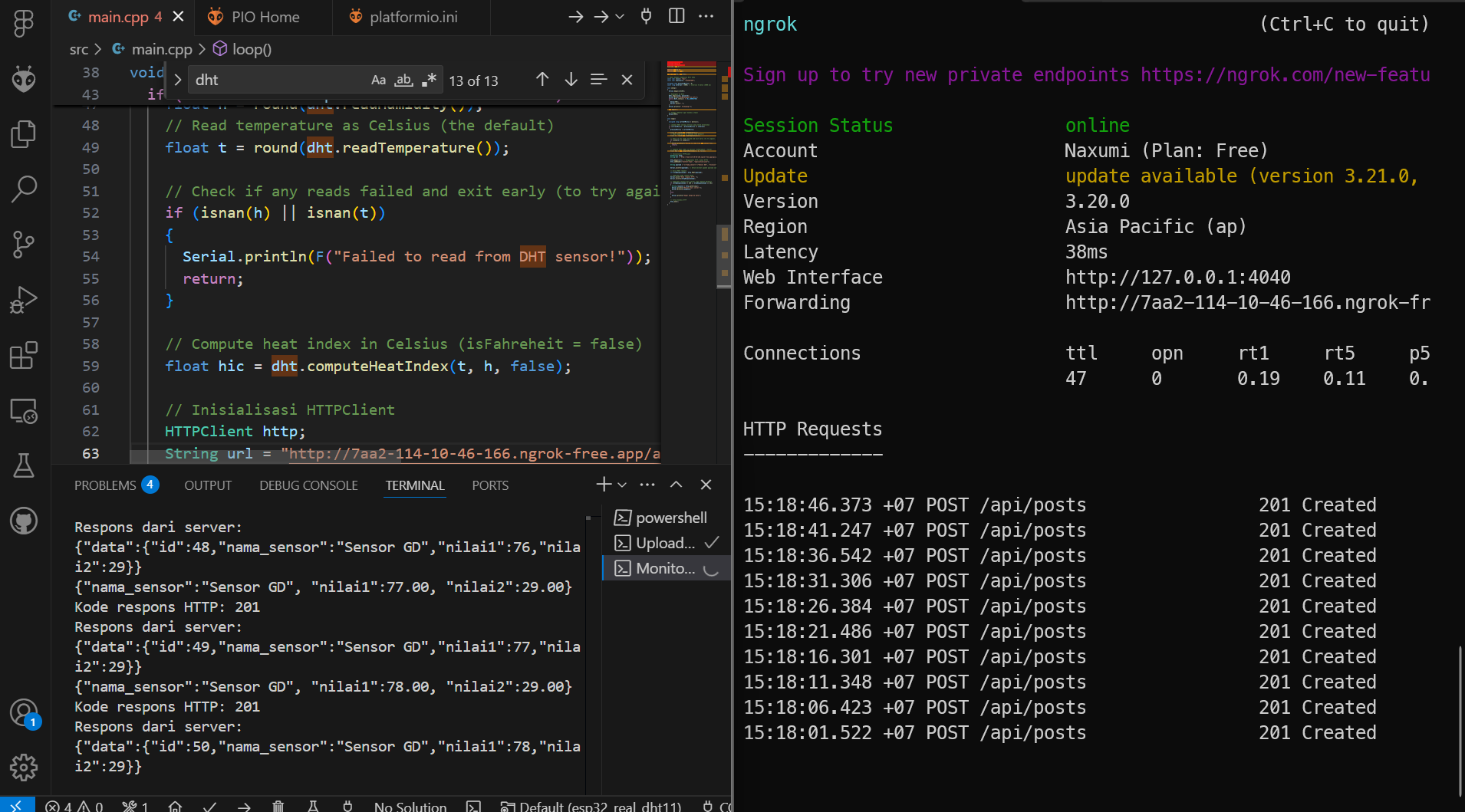
monitor\_port = COM7

monitor\_speed = 115200

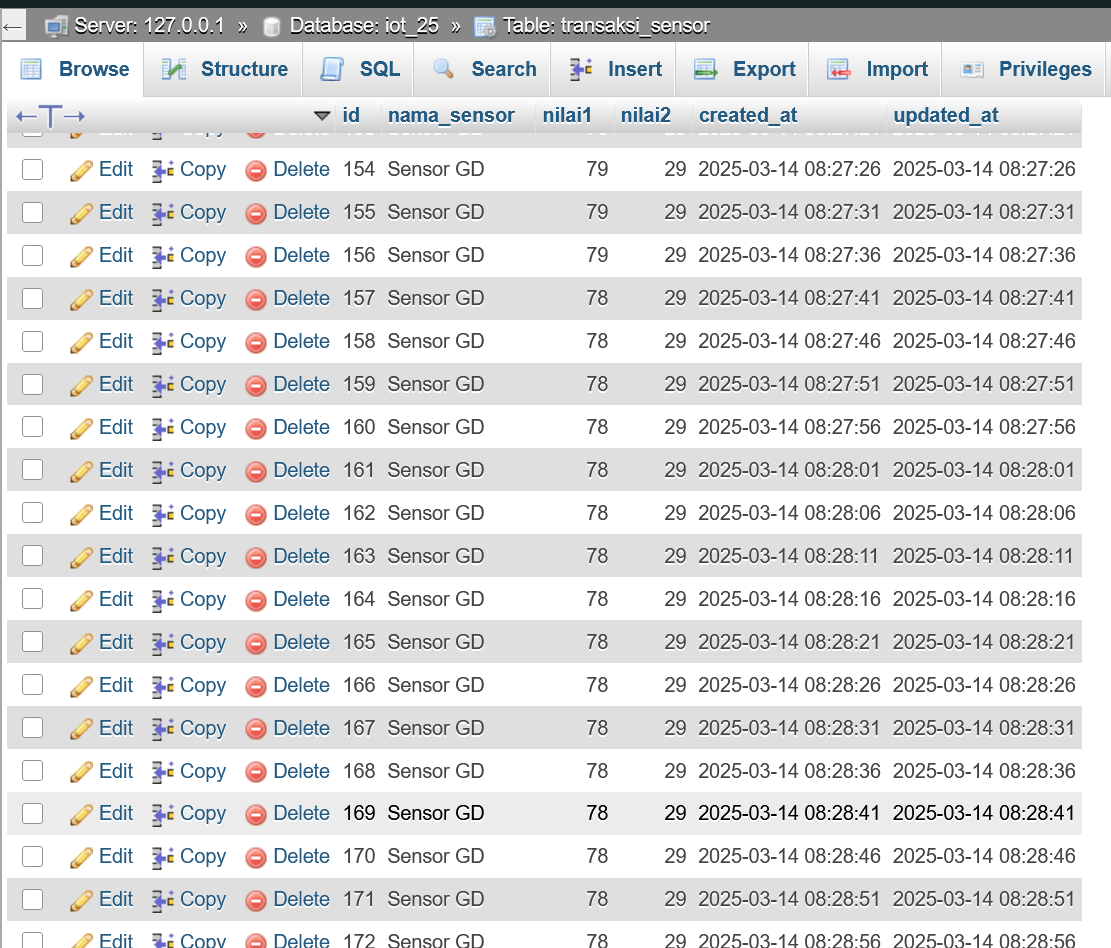
lib\_deps = adafruit/DHT sensor library@^1.4.4

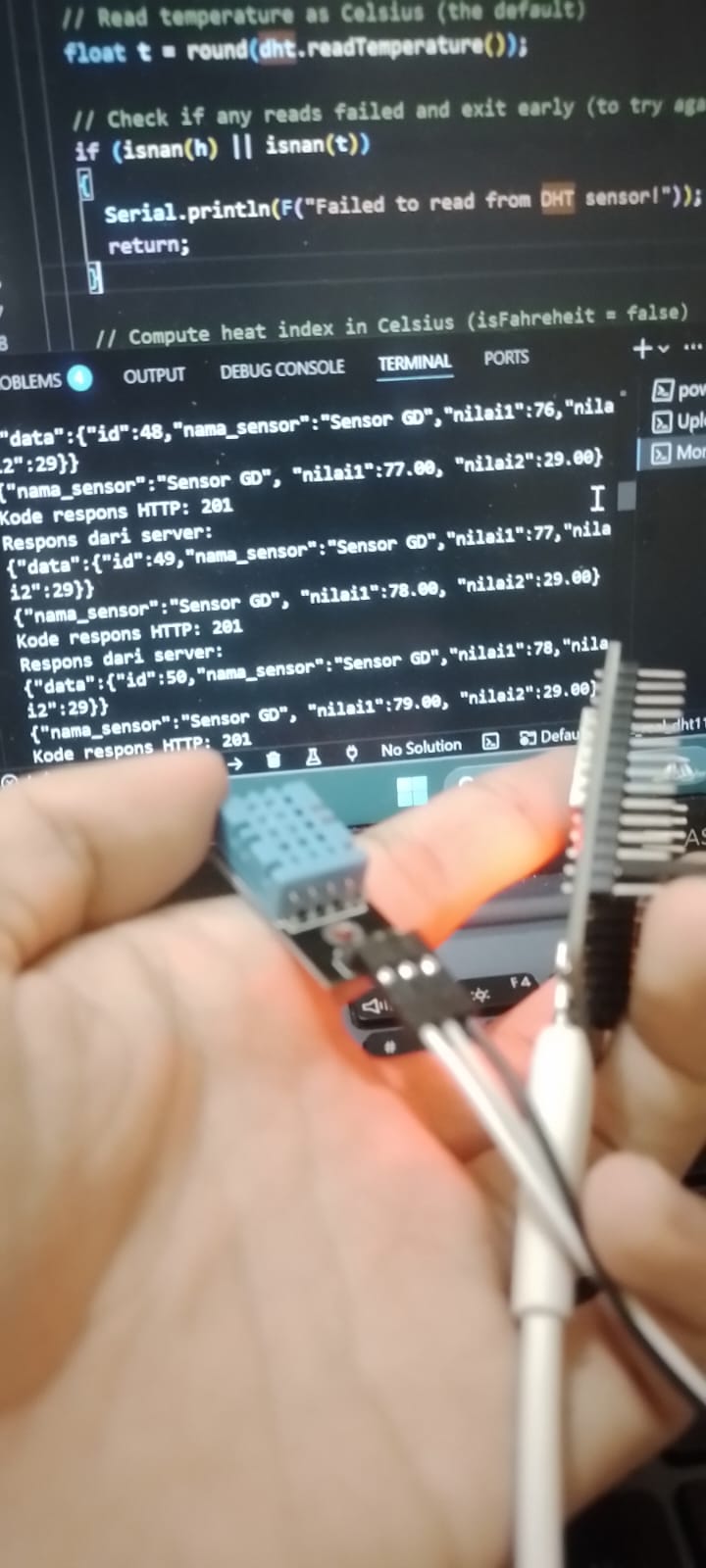
adafruit/Adafruit Unified Sensor@^1.1.14

1. Klik upload, setelah selesai cek serial monitor untuk melihat printout
2. Data berhasil terkirim



1. Cek database juga apakah berhasil terkirim





**3. Results and Discussion (Hasil dan Pembahasan)**

* **Verifikasi Hardware:** Proses instalasi driver CP210x berhasil, dan ESP32 terdeteksi di Device Manager (misalnya, pada COM3).
* **Upload Program Dasar:** Kode LED blinking dan WiFi scanning berhasil di-upload ke hardware ESP32, yang terbukti dengan output pada serial monitor.
* **Pengiriman Data Sensor:** Kode untuk mengirim data DHT22 melalui HTTP POST berhasil dijalankan. Serial monitor menunjukkan kode respons HTTP 200/201, yang menandakan data sensor berhasil dikirim ke API Laravel. Data tersebut juga dapat diverifikasi masuk ke database.
* **Integrasi Sistem:** Penggunaan Ngrok memungkinkan API diakses secara publik, dan hardware ESP32 dapat berkomunikasi dengan API melalui jaringan eksternal. Hal ini membuktikan sistem IoT yang terintegrasi mulai dari sensor hingga backend.

**4. Kesimpulan**

Praktik Real Hardware ESP32 berhasil menunjukkan bahwa perangkat nyata dapat digunakan dalam implementasi sistem IoT. Mulai dari instalasi driver, pengunggahan kode melalui PlatformIO, hingga pengiriman data sensor ke API melalui Ngrok, semua langkah berjalan sesuai rencana. Integrasi antara hardware ESP32 dan API backend memastikan data sensor dapat dikirim dan disimpan secara real-time, membuka peluang aplikasi IoT yang lebih kompleks di masa depan.