**LAPORAN PRAKTIKUM INTERNET OF THINGS (IoT)**

**Pemindaian Jaringan WiFi dan Pengiriman Data Sensor DHT22 Menggunakan ESP32**

*Author : Nelly Fananda Melani*

*Fakultas Vokasi, Universitas Brawijaya*

*Email:* [nellyfananda15@student.ub.ac.id](mailto:nellyfananda15@student.ub.ac.id)

**Abstract**

Praktikum ini bertujuan untuk menguji koneksi WiFi pada mikrokontroler ESP32 sebagai bagian dari pemahaman dasar sistem Internet of Things (IoT). ESP32 digunakan untuk memindai jaringan WiFi di sekitarnya dan menampilkan informasi berupa nama access point (SSID) dan kekuatan sinyal (RSSI). Pemrograman dilakukan melalui Visual Studio Code dengan ekstensi PlatformIO IDE. Hasil pemindaian ditampilkan secara waktu nyata melalui Serial Monitor. Setelah proses pemindaian, ESP32 dihubungkan ke jaringan WiFi, kemudian mengirimkan data dari sensor suhu dan kelembapan (DHT22) ke server API menggunakan metode HTTP POST. Data yang dikirimkan berhasil diterima dan ditampilkan melalui backend Laravel yang terhubung dengan ngrok. Praktikum ini memberikan pemahaman menyeluruh mengenai konektivitas WiFi serta alur pengiriman data dari mikrokontroler ke server berbasis internet.

**Kata kunci** — Internet of Things, ESP32, WiFi, HTTP POST, Laravel

**1. Introduction**

**1.1 Latar Belakang**

Perkembangan teknologi telah mempengaruhi berbagai aspek kehidupan, salah satunya adalah dalam bidang komunikasi dan pengolahan data melalui perangkat yang saling terhubung. Salah satu inovasi terbesar dalam bidang ini adalah Internet of Things (IoT), di mana perangkat fisik dapat terhubung dengan internet dan saling berkomunikasi untuk berbagi data secara real-time. Dalam konteks IoT, salah satu komponen penting adalah konektivitas antar perangkat, yang sering kali dihubungkan melalui jaringan nirkabel. WiFi menjadi salah satu teknologi komunikasi yang paling banyak digunakan dalam sistem IoT karena kecepatan, kemudahan, dan jangkauannya yang luas. Keuntungan utama dari WiFi adalah kemampuannya untuk menyediakan konektivitas internet secara stabil tanpa memerlukan kabel, yang sangat memudahkan penggunaan dalam berbagai aplikasi berbasis IoT. Salah satu mikrokontroler yang sangat populer untuk aplikasi IoT adalah ESP32. Mikrokontroler ini dilengkapi dengan modul WiFi dan Bluetooth internal, memungkinkan perangkat untuk terhubung langsung ke jaringan internet. Keunggulan lainnya adalah kemampuannya dalam menangani proses data secara cepat dan efisien, menjadikannya pilihan utama untuk pengembangan perangkat IoT yang memerlukan konektivitas dan pemrosesan data secara real-time.

Pada praktik ini, ESP32 digunakan untuk melakukan pemindaian jaringan WiFi yang ada di sekitar perangkat. Proses ini memungkinkan pengguna untuk melihat berbagai jaringan WiFi yang tersedia beserta informasi terkait seperti SSID (nama jaringan) dan RSSI (kekuatan sinyal). Selain itu, ESP32 juga akan digunakan untuk menghubungkan ke jaringan WiFi yang dipilih dan mengirimkan data sensor suhu dan kelembapan (DHT22) ke server menggunakan metode HTTP POST. Data yang terkirim akan diproses dan ditampilkan melalui backend Laravel untuk dianalisis lebih lanjut. Eksperimen ini bertujuan untuk memberikan pemahaman dasar tentang penggunaan ESP32 dalam mengelola konektivitas WiFi serta pengiriman data dalam sistem IoT, yang merupakan langkah pertama menuju pengembangan aplikasi IoT yang lebih kompleks.

**1.2 Tujuan Eksperimen**

Eksperimen ini bertujuan untuk:

1. Mempelajari cara kerja pemindaian jaringan WiFi menggunakan ESP32.
2. Menampilkan informasi SSID dan RSSI melalui Serial Monitor.
3. Menghubungkan ESP32 ke jaringan WiFi yang terdeteksi.
4. Mengirimkan data dari sensor suhu dan kelembapan (DHT22) ke server menggunakan HTTP POST.
5. Menampilkan data yang berhasil dikirim melalui backend Laravel.

**2. Methodology (Metodologi)**

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

1. Mikrokontroler: ESP32 DevKit v1
2. Sensor: DHT22 (sensor suhu dan kelembaban)
3. Software:
   1. PlatformIO (dalam Visual Studio Code)
   2. Ngrok
   3. Laravel (untuk API Backend)
   4. Postman
   5. MySQL (Database)
   6. Serial Monitor (via PlatformIO)
4. Jaringan WiFi: Hotspot smartphone sebagai access point
5. Perlengkapan tambahan: Kabel jumper, breadboard

**2.2 Implementation Steps (Langkah Implementasi)**

Berikut adalah langkah-langkah implementasi sistem Internet of Things menggunakan ESP32 dan Laravel API:

**1. Persiapan Proyek di PlatformIO**

1. Membuat proyek baru di PlatformIO dengan board esp32doit-devkit-v1 pada file main.cpp

#include <Arduino.h>

#include <WiFi.h>

#include <HTTPClient.h>

#include "DHT.h"

#define DHTPIN 27

#define DHTTYPE DHT22

DHT dht(DHTPIN, DHTTYPE);

// Ganti dengan kredensial WiFi Anda

const char\* ssid = "iPhone";

const char\* password = "izindulurek";

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://b917-175-45-190-3.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. Menambahkan dependensi library sensor DHT pada file platformio.ini:

lib\_deps =

    adafruit/DHT sensor library@^1.4.4

    adafruit/Adafruit Unified [Sensor@^1.1.14](mailto:Sensor@%5e1.1.14)

1. Menambahkan konfigurasi port dan kecepatan serial pada file platformio.ini:

upload\_port = COM3

monitor\_port = COM3

monitor\_speed = 115200

**2. Pengujian Koneksi WiFi**

1. Membuat kode awal untuk mengecek apakah ESP32 dapat mendeteksi jaringan WiFi di sekitarnya.
2. Upload kode ke ESP32 dan buka Serial Monitor untuk melihat daftar access point dan kekuatan sinyalnya.

**3. Menyiapkan API Laravel**

1. Menjalankan server Laravel lokal menggunakan perintah:

php artisan serve --host=0.0.0.0 --port=8080

1. Membuka server Laravel untuk akses publik menggunakan Ngrok:

ngrok http --scheme=http 8080

1. Mencatat URL endpoint yang diberikan oleh Ngrok untuk digunakan dalam kode ESP32.

**4. Menyambungkan Sensor DHT22 ke ESP32**

Melakukan wiring kabel antara pin DHT22 dan ESP32:

* 1. VCC → 3V3
  2. GND → GND
  3. Data → GPIO 27

**5. Menulis dan Mengunggah Program utama ke ESP32**

1. Mengisi file main.cpp dengan program yang:
   1. Menghubungkan ESP32 ke WiFi
   2. Membaca data dari sensor DHT22
   3. Mengirim data ke server Laravel menggunakan HTTP POST dengan format JSON
2. Contoh payload yang dikirim:

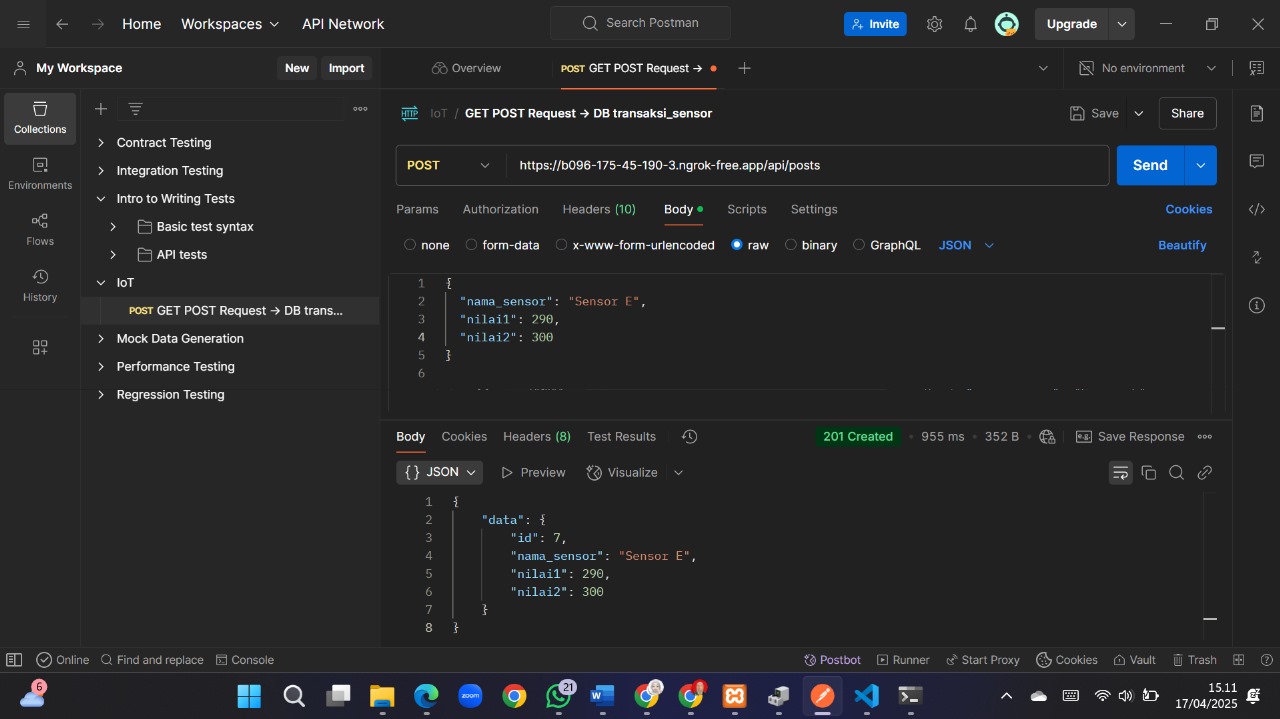
{

"nama\_sensor": "Sensor E",

"nilai1": 290,

"nilai2": 300

}



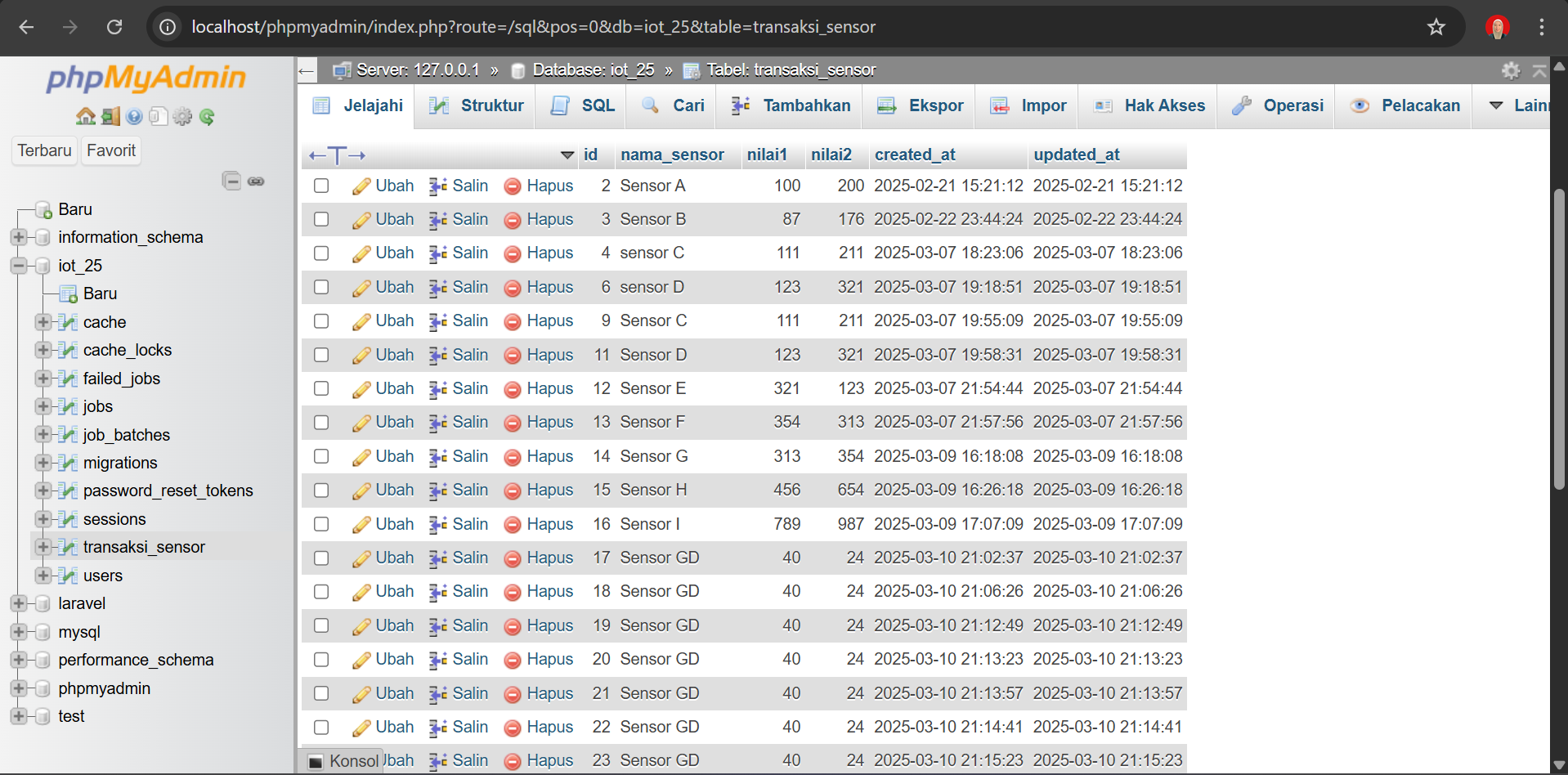
**6. Melihat Hasil di Serial Monitor**

Membuka Serial Monitor untuk memantau:

* 1. Koneksi WiFi
  2. Pembacaan sensor
  3. Payload yang dikirim
  4. HTTP response dari server

**7. Memastikan Data Masuk ke Database**

1. Mengecek database Laravel untuk memastikan bahwa data dari sensor berhasil tersimpan.
2. Alternatifnya, bisa juga mengecek lewat route API Laravel (/api/posts) untuk melihat data JSON yang masuk.

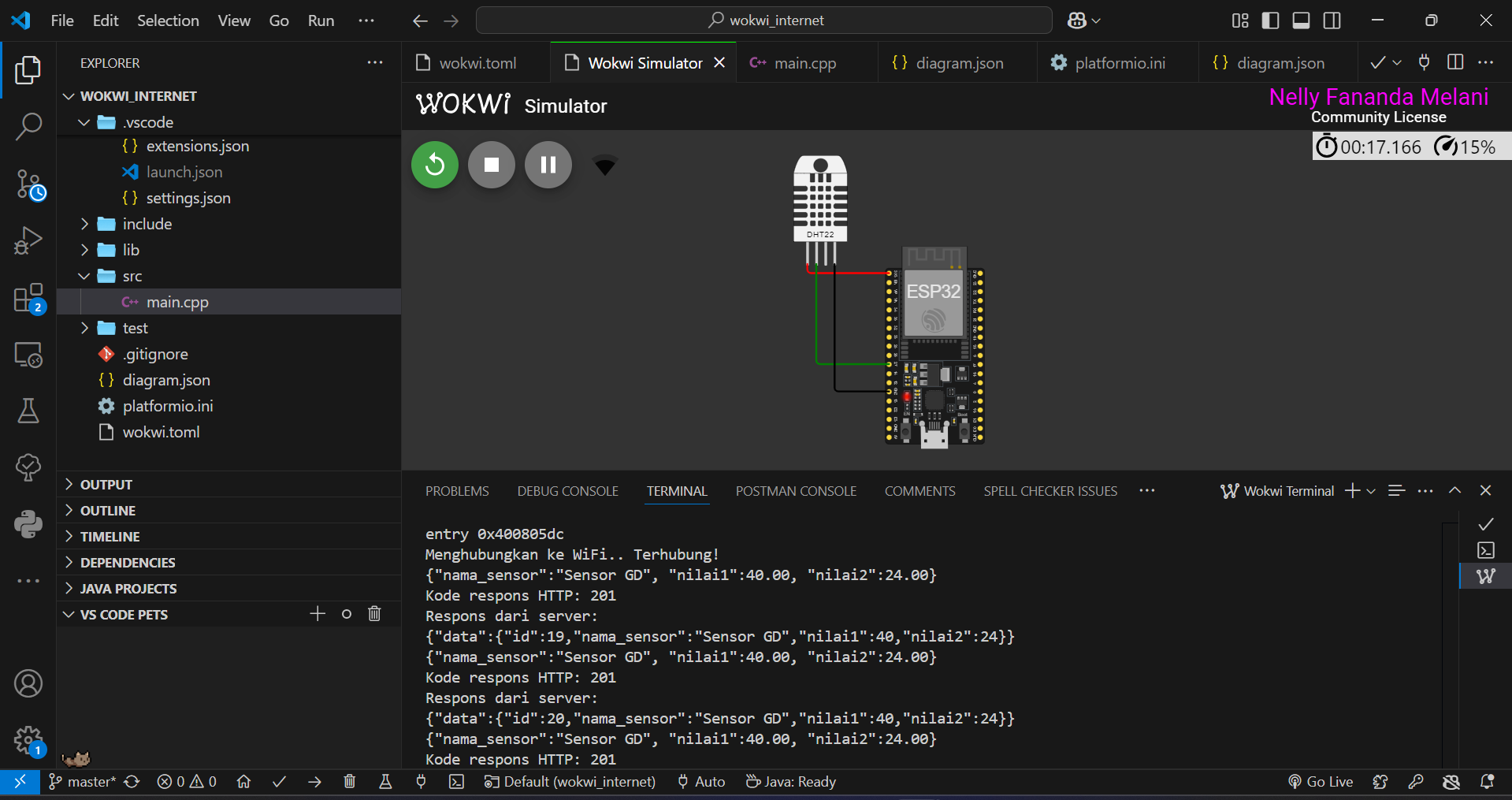


**3. Results and Discussion (Hasil dan Pembahasan)**

**3.1 Experimental Results (Hasil Eksperimen)**

Pada tahap ini dilakukan eksperimen dengan menggunakan mikrokontroler ESP32 yang telah terhubung dengan sensor DHT22 untuk membaca suhu dan kelembaban lingkungan. Data yang diperoleh dari sensor akan dikirimkan ke server melalui metode HTTP POST menuju endpoint API Laravel yang telah dibuat dan diakses melalui NGROK.

Hasil dari eksperimen ditampilkan melalui Serial Monitor pada Visual Studio Code (VSCode). Berikut adalah hasil yang berhasil ditampilkan:



Tampilan menunjukkan bahwa data berhasil dikirim ke server dengan kode respons HTTP 201, menandakan permintaan berhasil dibuat. Setiap 5 detik, data suhu dan kelembaban baru dikirim dan diterima oleh server.

Contoh Data JSON Dikirim:

|  |
| --- |
| {  "nama\_sensor": "Sensor GD",  "nilai1": 30,  "nilai2": 60  } |

**Tabel Hasil Eksperimen**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Nama Sensor | Suhu (°C) | Kelembaban (%) | Kode Respons | ID Data |
| 1 | Sensor GD | 40.00 | 24.00 | 201 | 19 |
| 2 | Sensor GD | 40.00 | 24.00 | 201 | 20 |
| 3 | Sensor GD | 40.00 | 24.00 | 201 | 21 |

**3.2 Performance Evaluation (Evaluasi Kerja)**

Eksperimen ini mengonfirmasi beberapa pengamatan berikut:

1. Koneksi WiFi yang Stabil: ESP32 berhasil terhubung ke jaringan WiFi dan mempertahankan koneksi selama proses pengiriman data berlangsung.
2. Pembacaan Sensor yang Konsisten: Sensor DHT22 mampu memberikan nilai suhu dan kelembaban yang akurat dan stabil selama beberapa siklus pembacaan.
3. Pengiriman Data ke Server Berhasil: Data berhasil dikirimkan ke endpoint API menggunakan metode HTTP POST, ditandai dengan kode respons HTTP 201 setiap kali data dikirim.
4. Pemanfaatan Simulasi Efektif: Penggunaan Wokwi Simulator mempercepat proses debug dan pengujian logika program, sehingga tidak tergantung langsung pada perangkat keras fisik di tahap awal pengembangan.

**3.3 Discussion (Pembahasan)**

Eksperimen ini menunjukkan bagaimana mikrokontroler seperti ESP32 dapat digunakan untuk membangun sistem pemantauan lingkungan berbasis IoT. Berikut beberapa poin penting dari pembahasan:

1. Integrasi Sensor DHT22:

Sensor ini terbukti handal dalam membaca parameter suhu dan kelembaban. Nilai yang dihasilkan dapat langsung diproses dan dikirim ke server untuk dianalisis lebih lanjut.

1. Pemrograman Interval Waktu:

Dengan memanfaatkan fungsi millis(), sistem dapat mengirim data setiap 5 detik tanpa mengganggu proses lainnya. Ini mencerminkan logika pemrograman non-blocking yang efisien.

1. Penggunaan HTTP dan API:

Komunikasi antara ESP32 dan server backend dilakukan menggunakan protokol HTTP POST. Payload dalam format JSON menunjukkan fleksibilitas ESP32 dalam bekerja dengan sistem modern berbasis web/API.

**4. Appendix (Lampiran)**

