# LAPORAN PRAKTIKUM INTERNET OF THINGS (IoT)

Fakultas Vokasi, Universitas Brawijaya

**Praktik Akses API Melalui Simulasi WOKWI**

*Eva Latifah*

*Fakultas Vokasi, Universitas Brawijaya*

*Email: ev.tifa@gmail.com*

**Abstrak**

Pada bab ini, dilakukan praktik akses API Laravel melalui simulasi menggunakan WOKWI dengan ESP32. Eksperimen ini bertujuan untuk memahami cara ESP32 berkomunikasi dengan server API menggunakan protokol HTTP dalam lingkungan simulasi. Proses dimulai dengan menjalankan server Laravel menggunakan perintah php artisan serve, kemudian menghubungkannya ke internet melalui NGROK untuk mendapatkan URL akses yang dapat digunakan oleh ESP32. Selanjutnya, ESP32 dikonfigurasi untuk terhubung ke jaringan WiFi WOKWI-GUEST dan melakukan permintaan HTTP ke API Laravel menggunakan metode GET dan POST. Hasil simulasi menunjukkan bahwa ESP32 berhasil mengakses API, menerima respons HTTP dengan kode status 200 (OK), serta mengirim data sensor suhu dan kelembaban ke server yang tersimpan dalam database MySQL. Implementasi ini membuktikan bahwa ESP32 dapat digunakan sebagai perangkat IoT untuk mengakses dan mengelola data melalui API dalam sistem berbasis web.

**Kata Kunci**—*ESP32, WOKWI, Laravel API, NGROK, HTTP, IoT, Sensor DHT22*

***Abstract***

*In this chapter, an experiment was conducted to access a Laravel API using WOKWI simulation with an ESP32. The objective of this experiment is to understand how the ESP32 communicates with a server API using the HTTP protocol in a simulated environment. The process begins by running the Laravel server with the php artisan serve command, followed by connecting it to the internet via NGROK to obtain an accessible URL for the ESP32. The ESP32 is then configured to connect to the WOKWI-GUEST WiFi network and perform HTTP requests to the Laravel API using GET and POST methods. The simulation results demonstrate that the ESP32 successfully accesses the API, receives an HTTP response with a 200 (OK) status code, and sends temperature and humidity sensor data to the server, which is stored in a MySQL database. This implementation proves that the ESP32 can be used as an IoT device to access and manage data through APIs in web-based systems.*

***Keywords****—ESP32, WOKWI, Laravel API, NGROK, HTTP, IoT, DHT22 Sensor*

**1. Introduction**

**1.1 Latar belakang**

Perkembangan Internet of Things (IoT) telah merevolusi cara perangkat berkomunikasi dan berinteraksi dengan sistem berbasis web. Salah satu mikrokontroler yang banyak digunakan dalam proyek IoT adalah ESP32, yang mendukung konektivitas WiFi dan dapat diintegrasikan dengan berbagai sensor untuk mengumpulkan serta mengirim data. Untuk mempermudah pengembangan dan pengujian aplikasi IoT, alat simulasi seperti WOKWI menyediakan lingkungan yang efisien untuk pengujian tanpa memerlukan perangkat keras fisik.

Laravel, sebagai salah satu framework PHP yang populer, sering digunakan untuk membangun API berbasis web yang berfungsi untuk mengelola dan menyimpan data. Namun, agar perangkat IoT dapat berinteraksi dengan API tersebut, diperlukan metode komunikasi yang andal melalui internet. NGROK memiliki peran penting dalam proses ini dengan menyediakan URL publik yang memungkinkan perangkat eksternal, seperti ESP32, mengakses server Laravel lokal.

Eksperimen ini bertujuan untuk mensimulasikan bagaimana perangkat ESP32 dapat berkomunikasi dengan API Laravel dalam lingkungan yang terkontrol menggunakan WOKWI. ESP32 diprogram untuk mengirim permintaan HTTP GET dan POST ke API, menunjukkan bagaimana perangkat IoT dapat berinteraksi dengan aplikasi web. Selain itu, sensor DHT22 juga diintegrasikan untuk mengukur suhu dan kelembaban, di mana data yang dikumpulkan akan disimpan dalam database MySQL.

**1.2 Tujuan eksperimen**

Praktikum ini bertujuan untuk:

1. Mensimulasikan akses API Laravel menggunakan ESP32 di WOKWI.
2. Menghubungkan ESP32 ke API Laravel melalui NGROK.
3. Mengirim dan menyimpan data sensor DHT22 ke database MySQL.
4. Menguji respons dan keandalan ESP32 dalam komunikasi HTTP.
5. Mengidentifikasi tantangan dalam integrasi IoT dengan web.
6. Dasar pengembangan sistem IoT untuk pemantauan dan otomatisasi.

**2. Methodology**

**2.1 Tools & Materials**

1. Komputer/Laptop dengan sistem operasi Windows/Linux/MacOS
2. Laravel 12
3. Composer
4. PHP
5. MySQL
6. Postman
7. Ngrok
8. WOKWI Simulator
9. ESP32
10. Sensor DHT22

**2.2 Implementation Steps**

#### **Menjalankan API Laravel**

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

#### **Membuat File Simulasi WOKWI**

* 1. Membuat proyek baru di PlatformIO.
  2. Menambahkan file main.cpp dengan script
  3. Menyesuaikan serverUrl dengan alamat URL yang diberikan oleh Ngrok.
  4. Meambahkan file wokwi.toml dan diagram.json untuk mendukung simulasi WOKWI.

#### **Menjalankan Ngrok**

1. Menjalankan perintah berikut untuk menghubungkan Laravel ke internet: > Wokwi Start Simulator
2. Menggunakan URL yang diberikan oleh Ngrok untuk menggantikan <URL\_NGROK> pada main.cpp. http://4f4d-175-45-191-9.ngrok-free.app

#### **Simulasi dengan Sensor DHT22**

1. Menambahkan sensor DHT22 ke dalam WOKWI Simulator.
2. Memodifikasi main.cpp untuk membaca suhu dan kelembaban lalu mengirimkan data ke API Laravel
3. Memodifikasi File platformio.ini
4. Menjalankan simulasi dengan perintah: > Wokwi Start Simulator
5. Memastikan data tersimpan di database MySQL melalui API Laravel.

**3. Results and Discussion**

**3.1 Experimental Results**

Pada praktik ini, dilakukan simulasi akses API Laravel menggunakan ESP32 melalui platform WOKWI. ESP32 dikonfigurasi untuk terhubung ke jaringan Wi-Fi dan mengirim permintaan HTTP ke API Laravel yang telah dijalankan menggunakan php artisan serve serta diakses secara publik melalui Ngrok. Simulasi ini memungkinkan pengujian komunikasi antara mikrokontroler dan server secara virtual tanpa perangkat fisik.

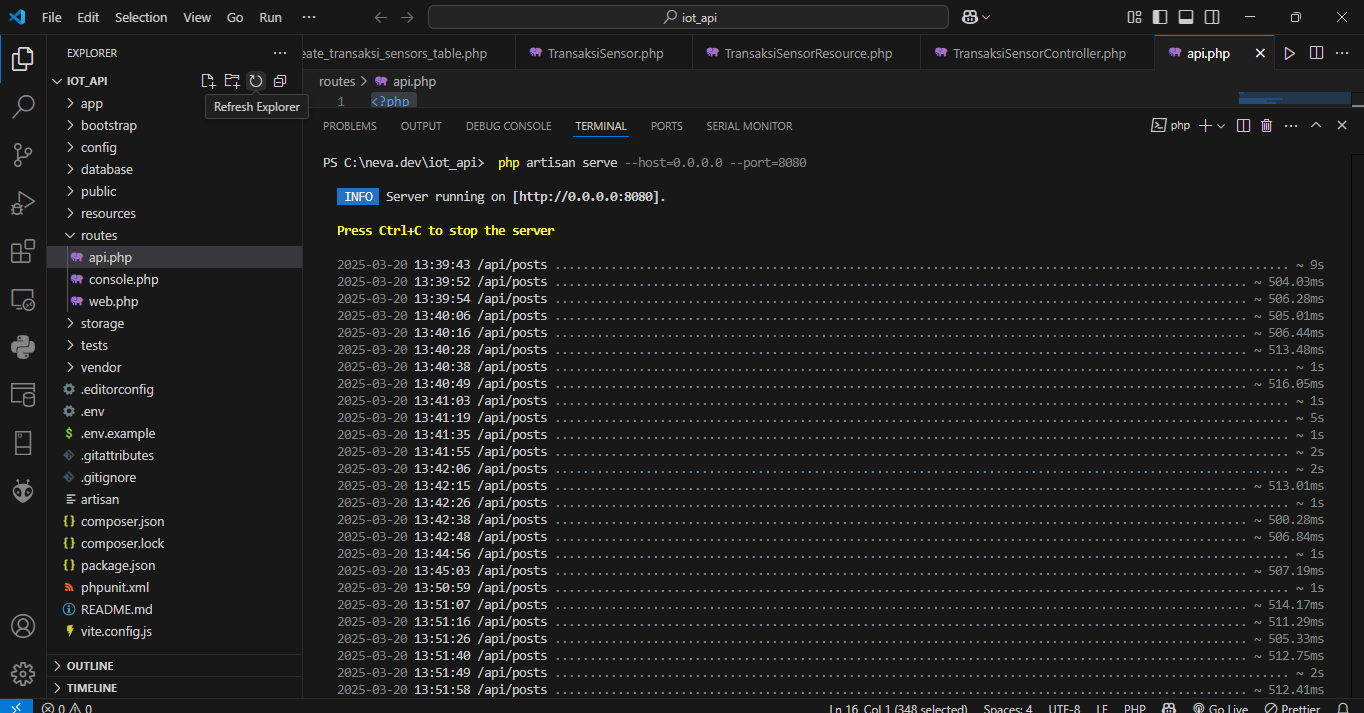
Proses implementasi mencakup instalasi dan konfigurasi Laravel, pembuatan database, serta pengujian API menggunakan Postman. Setelah itu, ESP32 dikodekan untuk mengambil data dari sensor suhu dan kelembaban (DHT22), lalu mengirimkan data tersebut ke API dalam format JSON. Hasil simulasi menunjukkan bahwa ESP32 dapat mengakses API dengan respons HTTP 200 (OK), menandakan komunikasi berhasil. Dengan metode ini, pengujian sistem berbasis IoT dapat dilakukan dengan lebih efisien sebelum diterapkan pada perangkat fisik.

Selain itu, penggunaan WOKWI sebagai simulator memberikan kemudahan dalam debugging dan pengujian tanpa memerlukan perangkat keras secara langsung. Dengan menghubungkan ESP32 ke Wi-Fi WOKWI-GUEST, data dari sensor DHT22 dapat dikirim ke server Laravel melalui metode HTTP POST. Data yang terkirim disimpan ke dalam database MySQL, memungkinkan pemantauan suhu dan kelembaban secara real-time.

Meskipun simulasi ini berjalan dengan baik, terdapat beberapa faktor yang dapat mempengaruhi keakuratan pengujian, seperti keterbatasan emulator dalam mereplikasi kondisi lingkungan nyata dan keterbatasan fitur ESP32 dalam mendukung HTTPS. Oleh karena itu, penggunaan Ngrok dengan skema HTTP menjadi solusi agar ESP32 tetap dapat mengakses API Laravel. Ke depannya, sistem ini dapat dikembangkan lebih lanjut dengan menambahkan fitur keamanan seperti autentikasi API dan enkripsi data agar lebih aman saat diimplementasikan dalam proyek IoT yang sesungguhnya

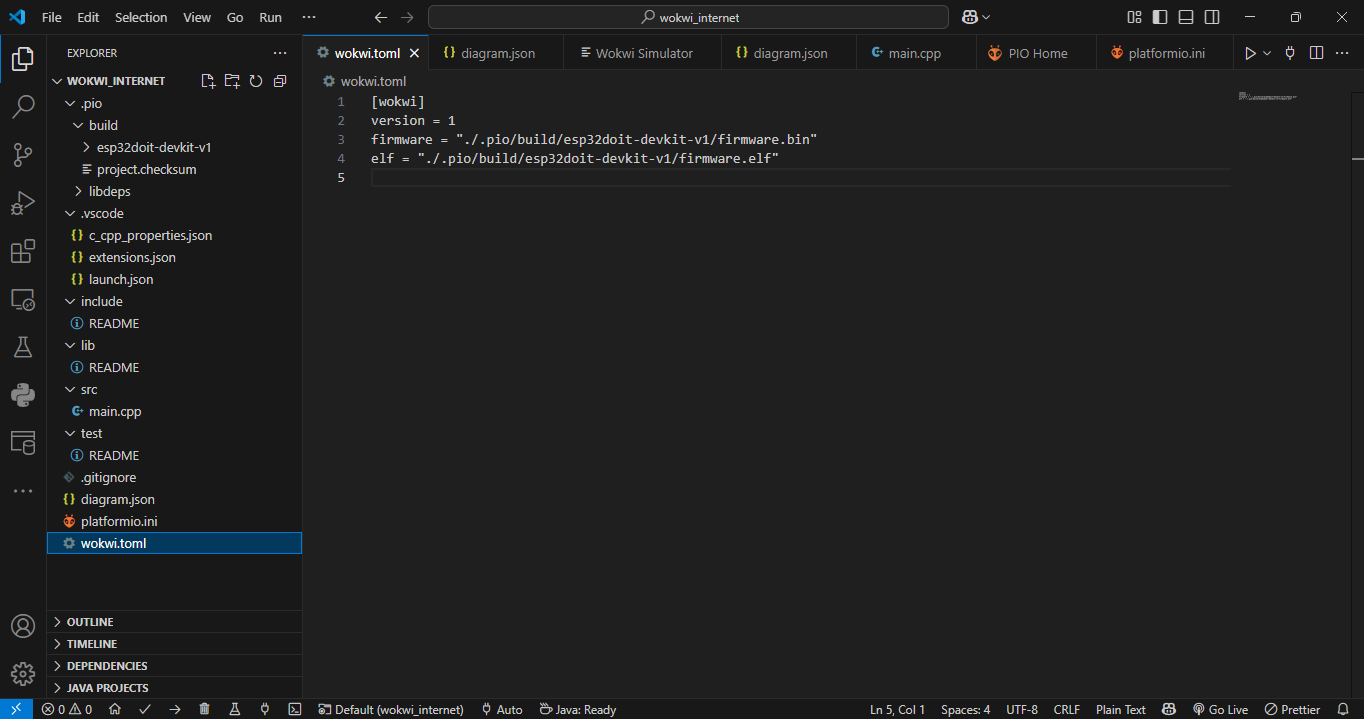
#### **Menjalankan API Laravel**

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



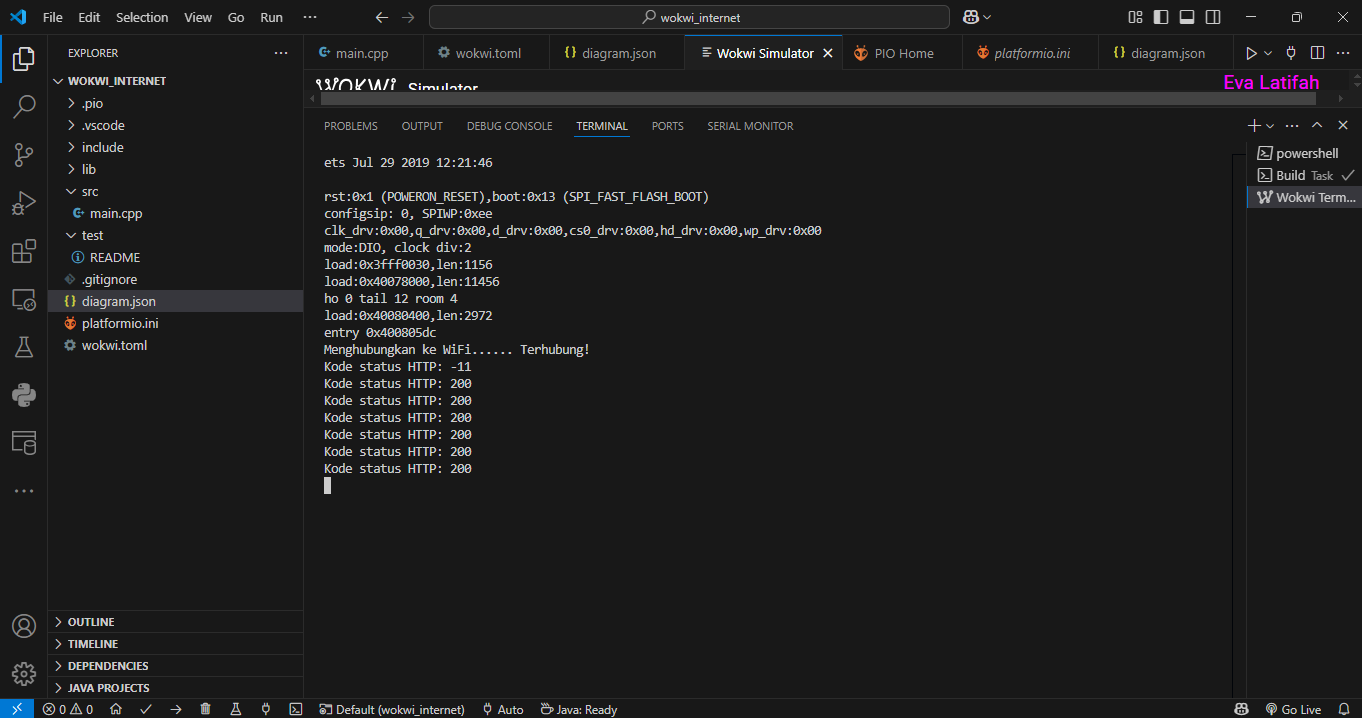
#### **Membuat File Simulasi WOKWI**

* 1. Membuat proyek baru di PlatformIO.
  2. Menambahkan file main.cpp dengan scrip
  3. Menyesuaikan serverUrl dengan alamat URL yang diberikan oleh Ngrok.
  4. Meambahkan file wokwi.toml dan diagram.json untuk mendukung simulasi WOKWI.



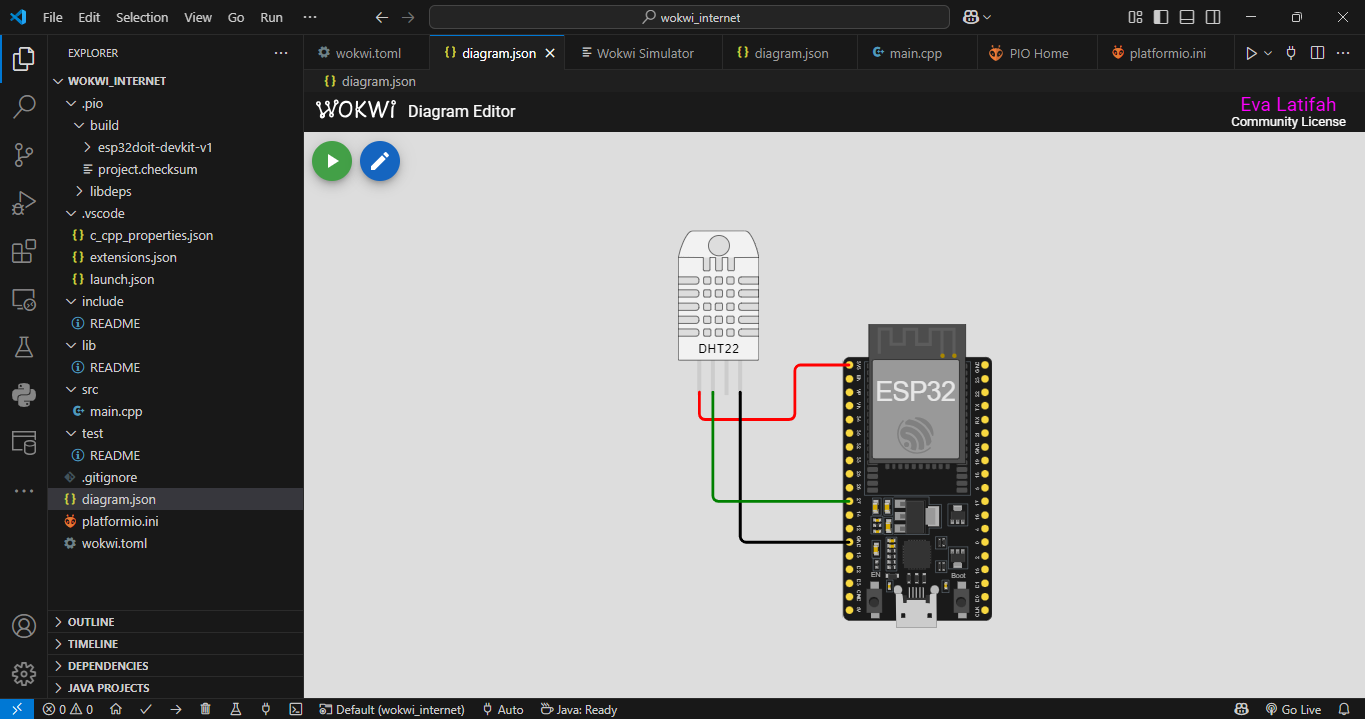
#### **Menjalankan Ngrok**

Menjalankan perintah berikut untuk menghubungkan Laravel ke internet: > Wokwi Start Simulator Menggunakan URL http://4f4d-175-45-191-9.ngrok-free.app

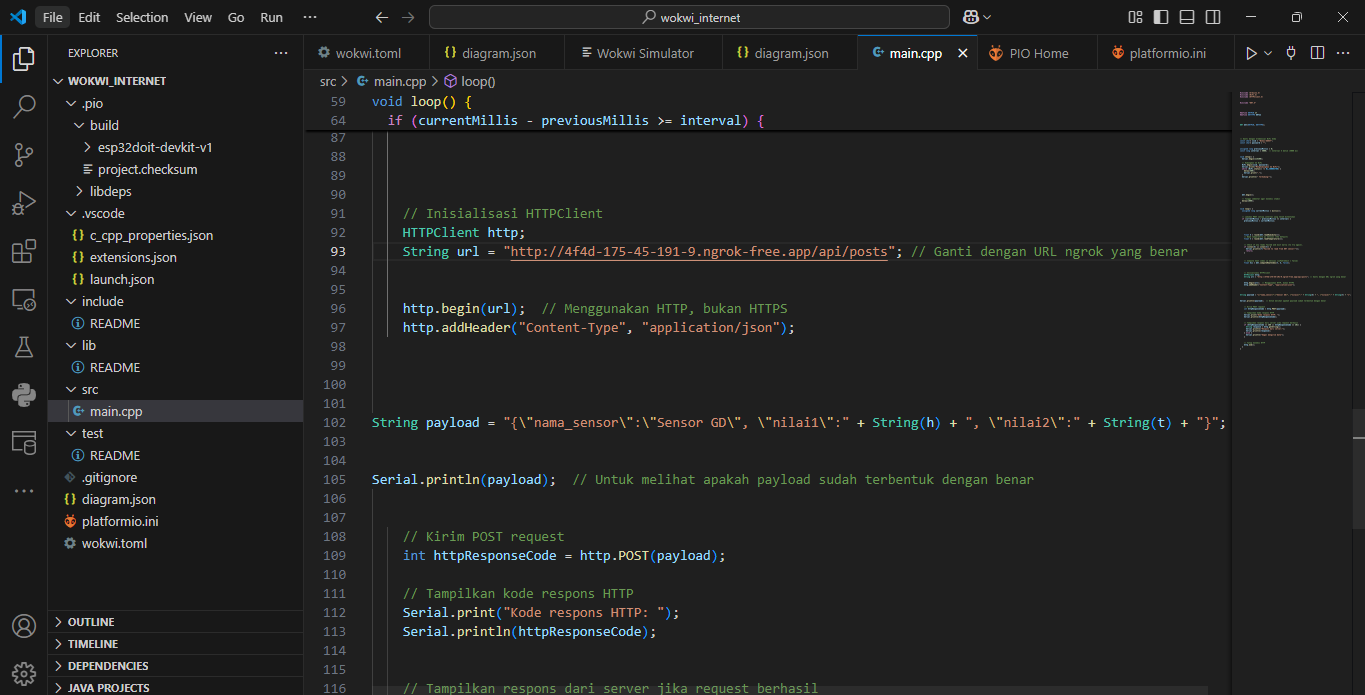
****

#### **Simulasi dengan Sensor DHT22**

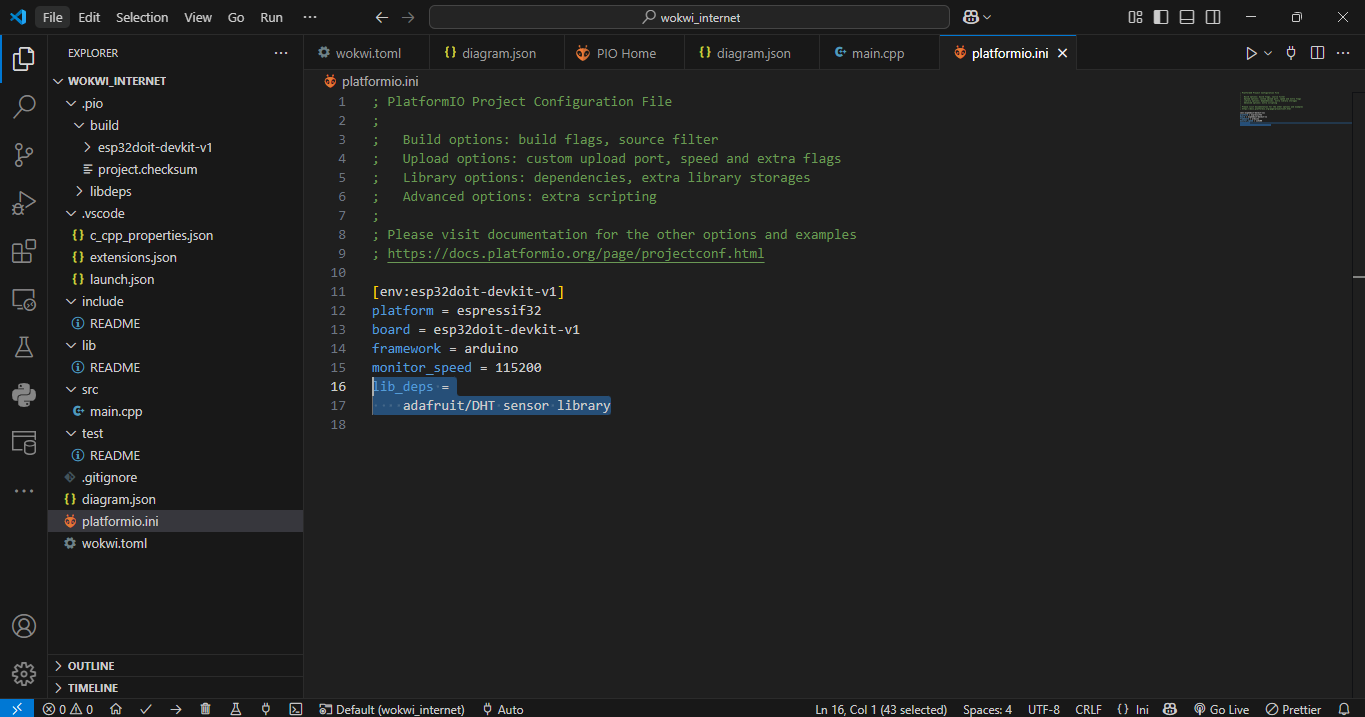
Menambahkan sensor DHT22 ke dalam WOKWI Simulator.



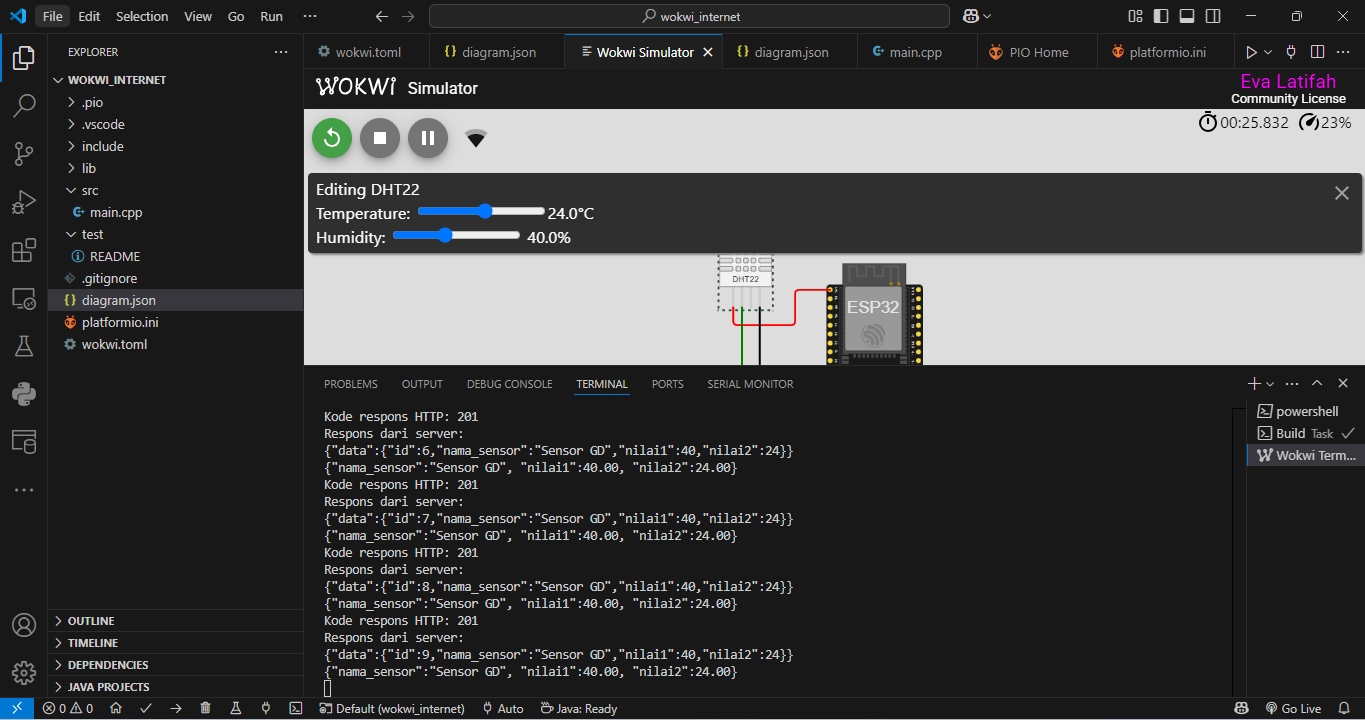
Memodifikasi main.cpp untuk membaca suhu dan kelembaban



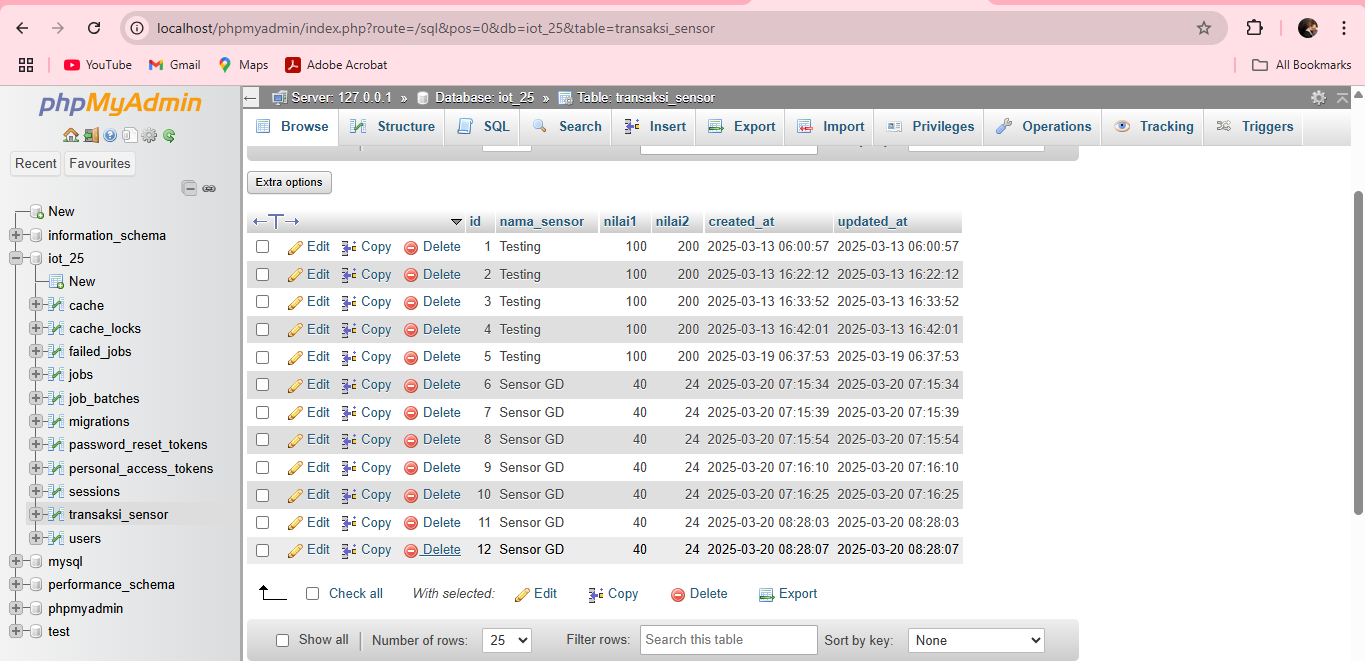
Memodifikasi File platformio.ini



Menjalankan simulasi dengan perintah: > Wokwi Start Simulator



Memastikan data tersimpan di database MySQL melalui API Laravel.

****

**4. Appendix**

1. main.cpp (TEST)

#include <WiFi.h>

#include <HTTPClient.h>

// Ganti dengan kredensial jaringan Wi-Fi Anda

// const char\* ssid = "Lab IT";

// const char\* password = "labit2024";

const char\* ssid = "Wokwi-GUEST";

const char\* password = "";

// URL lengkap server yang akan diakses

const char\* serverUrl = "http://4f4d-175-45-191-9.ngrok-free.app/api/posts";

// Interval waktu antara setiap permintaan (dalam milidetik)

const unsigned long interval = 5000;

unsigned long previousMillis = 0;

void setup() {

Serial.begin(115200);

WiFi.begin(ssid, password);

Serial.print("Menghubungkan ke WiFi...");

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

delay(500);

Serial.print(".");

}

Serial.println(" Terhubung!");

}

void loop() {

unsigned long currentMillis = millis();

// Periksa apakah interval waktu telah berlalu

if (currentMillis - previousMillis >= interval) {

previousMillis = currentMillis;

if (WiFi.status() == WL\_CONNECTED) {

HTTPClient http;

// Inisialisasi HTTPClient dengan URL server

http.begin(serverUrl);

// Mengirim permintaan HTTP GET

int httpResponseCode = http.GET();

// Menampilkan kode status HTTP

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

Serial.println(httpResponseCode);

// Menutup koneksi

http.end();

} else {

Serial.println("WiFi tidak terhubung."); }}}

1. diagram.json (TEST)

{

"version": 1,

"author": "Uri Shaked",

"editor": "wokwi",

"parts": [ { "type": "board-esp32-devkit-c-v4", "id": "esp", "top": 0, "left": 0, "attrs": {} } ],

"connections": [ [ "esp:TX", "$serialMonitor:RX", "", [] ], [ "esp:RX", "$serialMonitor:TX", "", [] ] ]

}

1. main.cpp (Simulasi Sensor)

#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 = "Wokwi-GUEST";

const char\* password = "";

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://4f4d-175-45-191-9.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. diagram.json (Simulasi Sensor)

{

"version": 1,

"author": "KAVITH BUDVIN",

"editor": "wokwi",

"parts": [

{ "type": "board-esp32-devkit-c-v4", "id": "esp", "top": 86.4, "left": 24.04, "attrs": {} },

{ "type": "wokwi-dht22", "id": "dht1", "top": 19.5, "left": -91.8, "attrs": {} }

],

"connections": [

[ "esp:TX", "$serialMonitor:RX", "", [] ],

[ "esp:RX", "$serialMonitor:TX", "", [] ],

[ "dht1:VCC", "esp:3V3", "red", [ "v19.2", "h67.2", "v-67.2" ] ],

[ "esp:GND.1", "dht1:GND", "black", [ "h0" ] ],

[ "dht1:SDA", "esp:27", "green", [ "v0" ] ]

],

"dependencies": {}

}

1. wokwi.toml

[wokwi]

version = 1

firmware = "./.pio/build/esp32doit-devkit-v1/firmware.bin"

elf = "./.pio/build/esp32doit-devkit-v1/firmware.elf"

1. platformio.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

monitor\_speed = 115200

lib\_deps =

adafruit/DHT sensor library