# LAPORAN PRAKTIKUM INTERNET OF THINGS (IoT)

Fakultas Vokasi, Universitas Brawijaya



# **Praktik Akses API Melalui Simulasi**

# **WOKWI via Visual Studio Code**

**Author :**

**Ivan AryaPutra Rachmadhani / 233140700111109**

**Fakultas Vokasi, Universitas Brawijaya**

**Email : ivanarya990@gmail.com**

**Abstract** (Abstrak)

Internet of Things (IoT) adalah konsep yang memungkinkan berbagai perangkat untuk saling berkomunikasi dan bertukar data melalui jaringan internet. Salah satu komponen penting dalam sistem IoT adalah akses terhadap layanan API (Application Programming Interface), yang memungkinkan perangkat untuk mendapatkan dan mengirim data ke server eksternal. Praktikum ini bertujuan untuk memahami bagaimana cara mengakses API melalui mikrokontroler ESP8266 menggunakan simulasi WOKWI di Visual Studio Code.

Dalam eksperimen ini, ESP8266 akan dikonfigurasi untuk mengakses API publik menggunakan protokol HTTP. Dengan memanfaatkan simulator WOKWI, proses pengujian dapat dilakukan tanpa perangkat keras fisik, sehingga lebih efisien dan fleksibel. Data yang diperoleh dari API dianalisis untuk menilai keberhasilan komunikasi antara ESP8266 dan server API.

Praktikum ini diharapkan dapat memberikan wawasan kepada mahasiswa mengenai penggunaan API dalam sistem IoT serta bagaimana simulasi dapat membantu dalam pengembangan dan pengujian perangkat lunak sebelum implementasi nyata.

**Keywords***:* *Internet of Things, MQTT, Smart Home, ESP8266, API*

**1. Introduction** (Pendahuluan)

**1.1 Latar belakang** praktikum IoT yang dilakukan

Dalam era digital saat ini, Internet of Things (IoT) menjadi salah satu teknologi yang berkembang pesat dan memiliki banyak aplikasi di berbagai bidang, termasuk smart home, smart city, dan industri 4.0. IoT memungkinkan perangkat elektronik seperti sensor dan mikrokontroler untuk berkomunikasi secara otomatis melalui jaringan internet. Salah satu aspek penting dalam pengembangan IoT adalah bagaimana perangkat dapat mengakses dan bertukar data dengan layanan berbasis cloud melalui API.

Mikrokontroler ESP8266 merupakan salah satu perangkat yang sering digunakan dalam proyek IoT karena memiliki modul WiFi terintegrasi yang memungkinkan komunikasi dengan server. Namun, pengujian dan debugging perangkat IoT secara langsung menggunakan perangkat keras sering kali memakan waktu dan biaya. Oleh karena itu, penggunaan simulator seperti WOKWI menjadi solusi yang efektif untuk menguji dan mengembangkan perangkat lunak sebelum implementasi di perangkat fisik.

Dengan menggunakan WOKWI, mahasiswa dapat memahami konsep akses API dalam sistem IoT tanpa perlu memiliki perangkat keras secara langsung. Simulasi ini juga memungkinkan pengujian berbagai skenario komunikasi antara ESP8266 dan layanan API, sehingga mempercepat proses pengembangan sistem IoT yang lebih efisien dan handal.

**1.2 Tujuan eksperimen**

1. Memahami cara mengakses API menggunakan ESP8266.
2. Menggunakan simulator WOKWI untuk menguji komunikasi dengan API.
3. Mengimplementasikan kode program di Visual Studio Code.

**2. Methodology (Metodologi)**

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

> **Hardware :** Simulasi berbasis WOKWI (ESP8266 virtual).

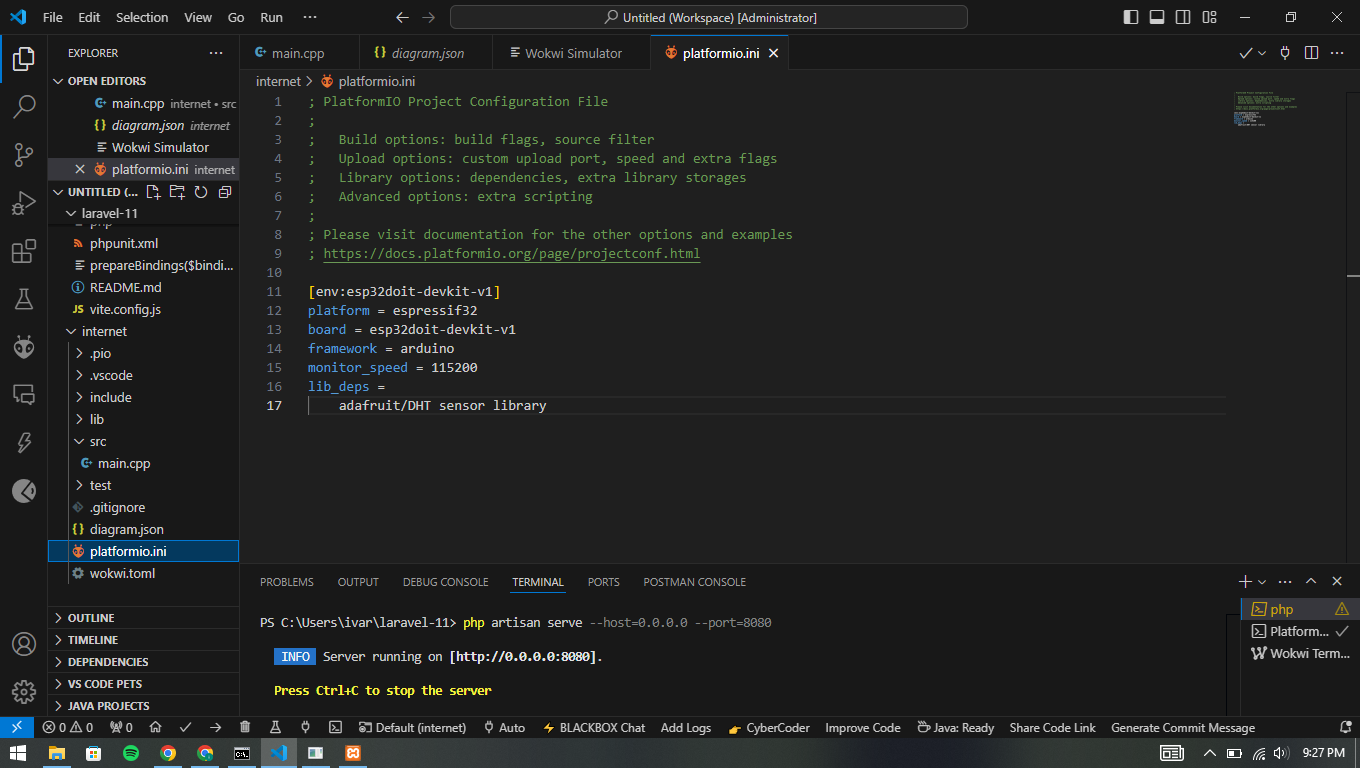
> **Software :**

* + WOKWI Simulator
  + Visual Studio Code
  + PlatformIO
  + Arduino Framework
  + API Public (misalnya OpenWeatherMap)

**2.2 Implementation Steps (Langkah Implementasi)**

> Langkah-langkah dalam menyusun sistem, pengkodean, dan pengujian diawali dengan menjalankan API menggunakan perintah berikut pada terminal

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



> Kemudian buka Visual Studio Code dan buat folder pada PlatformIO dengan menggunakan nama “internet”, untuk perangkat disesuaikan menggunakan ESP32 Devkit V1 DOIT dan framework Arduino

> Buka file **src > main.cpp,** lalu lakukan modifikasi file menggunakan syntax berikut,

#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://e6d3-2405-8740-6315-3520-5928-26b-7835-cd79.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.");

}

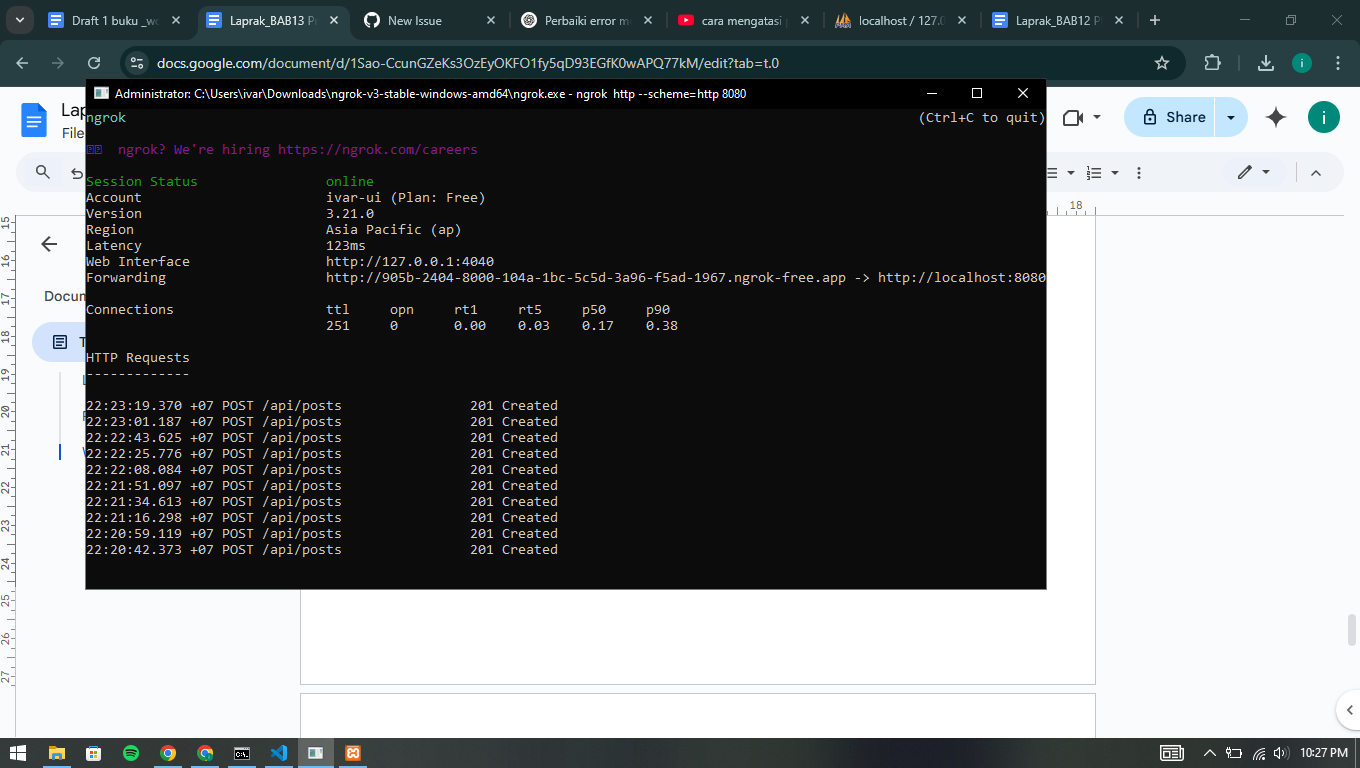
}

}

> sebelum build, lakukan konfigurasi pada URL dengan menyesuaikan URL yang didapat saat mengaktifkan NGROK pada terminal dengan perintah, **ngrok http --scheme=http 8080**,

sebagai contoh, lalu build dengan klik centang yang ada di sisi kanan atas.

**http://905b-2404-8000-104a-1bc-5c5d-3a96-f5ad-1967.ngrok-free.app/api/posts**



> Tambahkan file **wokwi.toml**

[wokwi]

version = 1

firmware = '.pio\build\esp32doit-devkit-v1\firmware.bin'

elf = '.pio\build\esp32doit-devkit-v1\firmware.elf'

> Tambahkan file **diagram.json,** menggunakan syntax berikut,

{

"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", "", [] ] ]

}

**>** Kemudian lakukan start dengan perintah, **> Wokwi Start Simulator**

> Simulasi diatas menunjukkan, ESP32 berhasil terhubung we WIFI wokwi-GUEST dan berhasil mengakses API laravel yang sudah dibuat pada BAB sebelumnya.  
  
> Pastikan muncul **Kode Status HTTP:200** pada terminal saat berjalan**,** yang artinya **“OK”** yang artinya request dari client telah berhasil untuk diproses oleh server.

> Lakukan modifikasi pada diagram.json dengan syntax berikut,

{

"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": {}

}

> Ubah konfigurasi pada platformIO.ini seperti berikut,

; 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

> Kembali buka **src > main.cpp,** lalu modifikasi file dengan sybntax berikut,  
#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://905b-2404-8000-104a-1bc-5c5d-3a96-f5ad-1967.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();

}

}

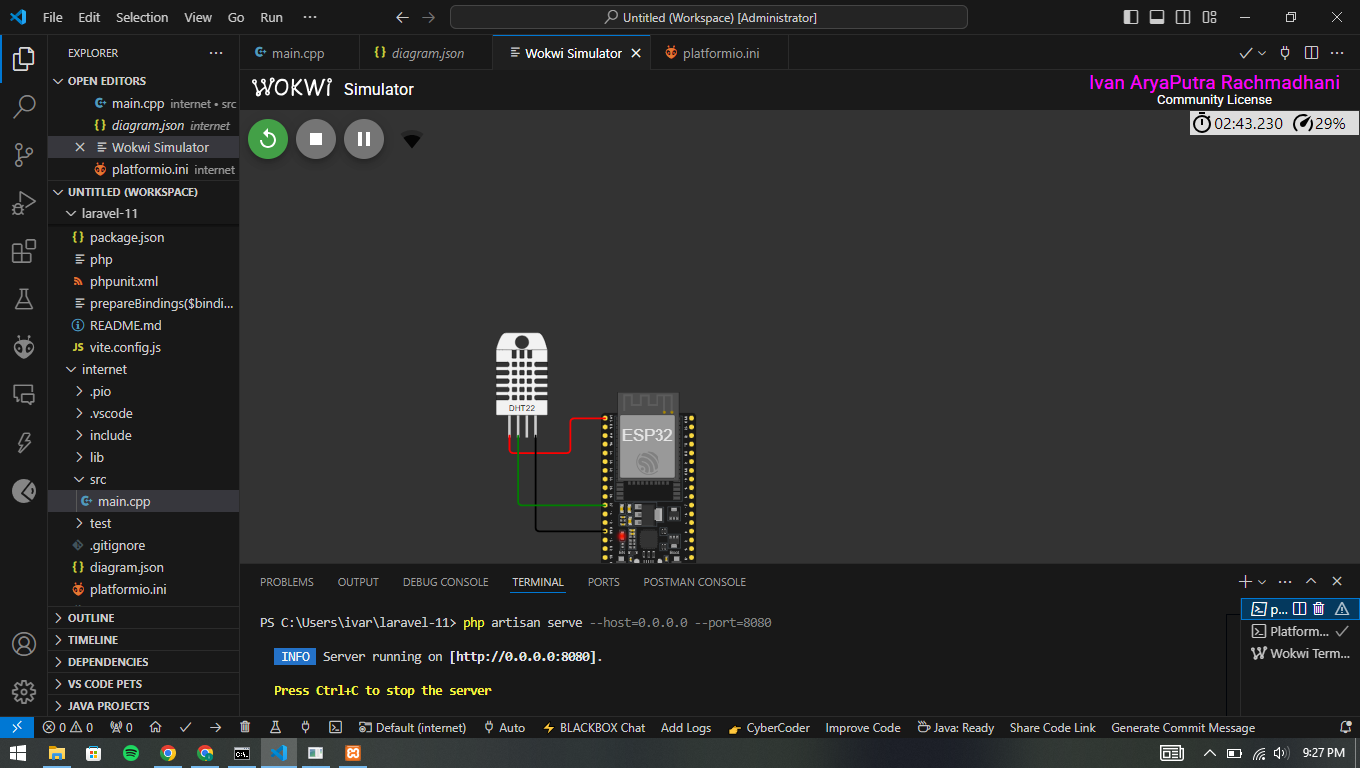
> Jangan lupa konfigurasi HTTP, sesuai dengan yang didapat pada saat aktivasi NGROK, sebagai contoh,

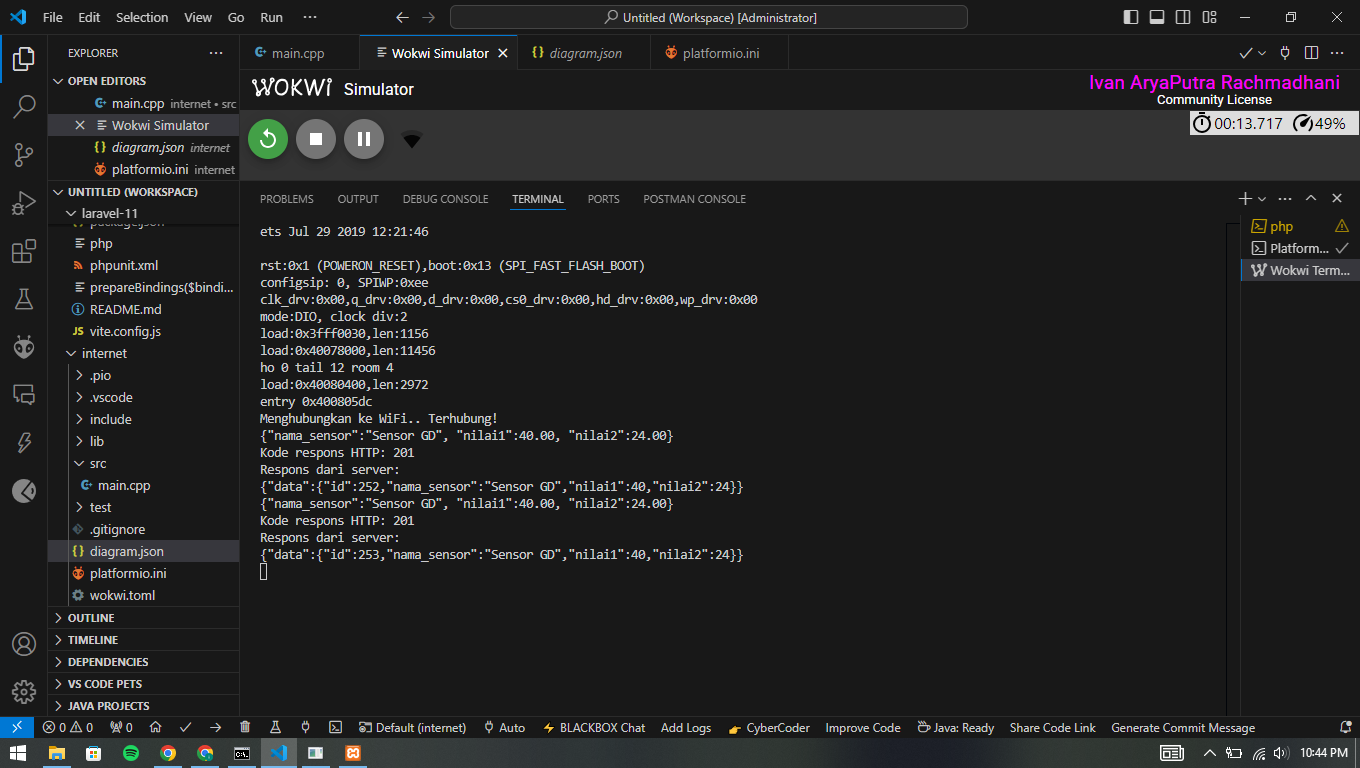
**http://905b-2404-8000-104a-1bc-5c5d-3a96-f5ad-1967.ngrok-free.app/api/posts**

> Jangan lupa menambahkan Endpoint **api/posts**

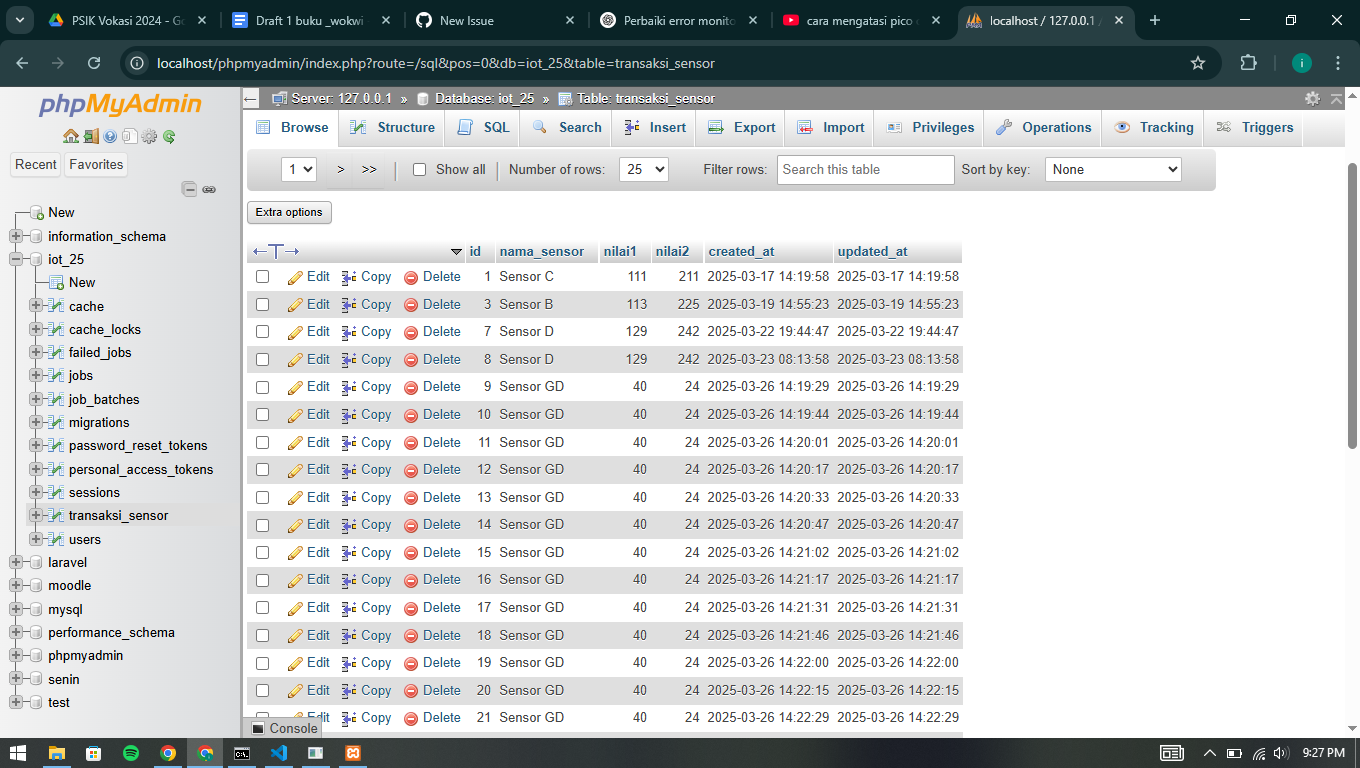
> Lakukan simulasi dengan perintah   
 **> Wokwi Start Simulator**

> Pastikan tampilan setelah menjalankan simulasi seperti berikut,





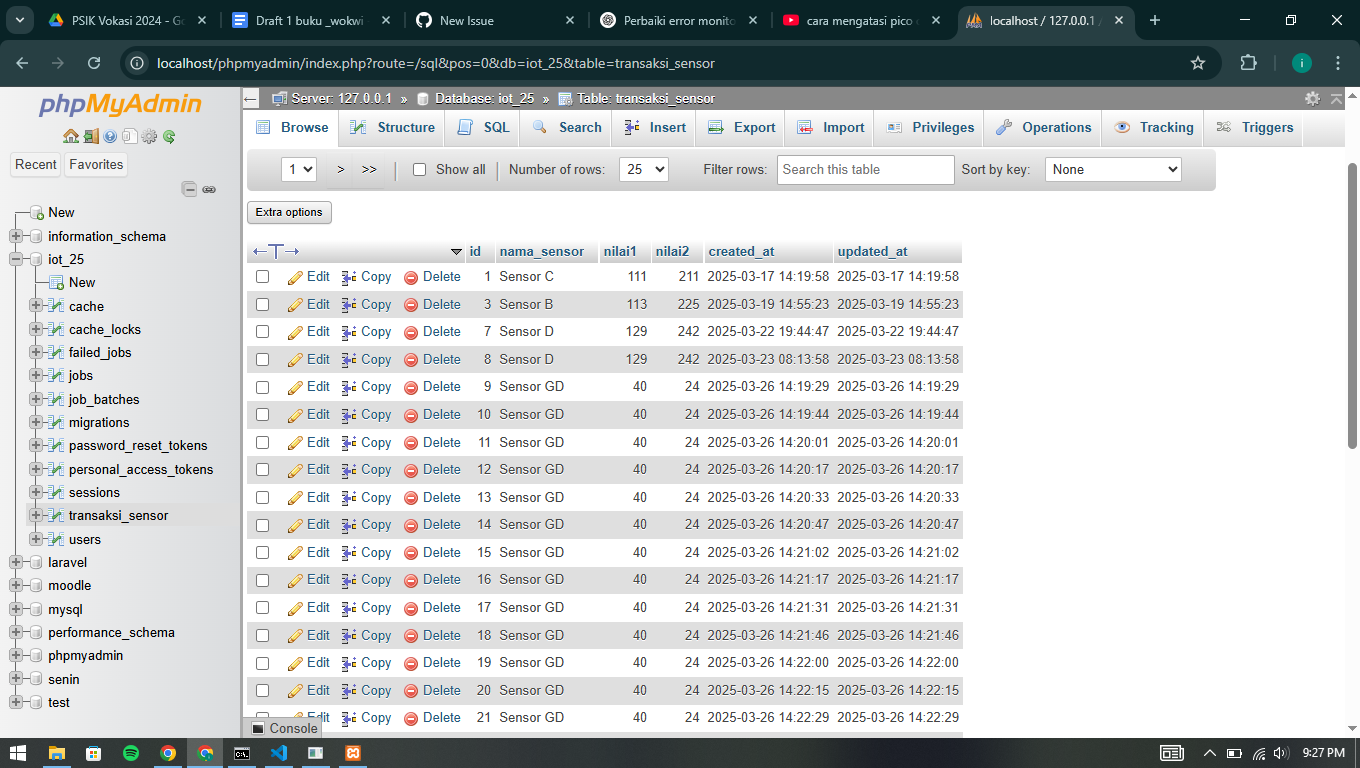
> Pastikan pada database muncul value yang muncul pada terminal Simulasi



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

**3.1 Experimental Results (Hasil Eksperimen)**

> Penyajian data pada praktikum,



**4. Appendix (Lampiran, jika diperlukan)**

> Laporan berupa dokumentasi lainnya pada praktikum BAB ini,

