**LAPORAN PRAKTIKUM INTERNET OF THINGS (IoT)** Fakultas Vokasi, Universitas Brawijaya

### **BAB 11 Praktik Simulasi Sensor Jarak (Ultrasonic)**



*Cantika Kelana*  
 Fakultas Vokasi, Universitas Brawijaya  
 Email: [cantikakln@student.ub.ac.id](mailto:cantikakln@student.ub.ac.id) / [cantikakln@student.ub.ac.id](mailto:cantikakln@student.ub.ac.id)

## **Abstract (Abstrak)**

Praktikum ini bertujuan untuk memahami cara kerja sensor Ultrasonic HC-SR04 dengan ESP32 melalui simulasi di Wokwi. Eksperimen ini mencakup pemasangan sensor HC-SR04 ke ESP32, pemrograman menggunakan Arduino IDE, serta pemantauan data jarak melalui serial monitor. Hasil eksperimen menunjukkan bahwa ESP32 mampu membaca dan menampilkan data jarak dalam satuan cm dan inch secara real-time.

**Kata kunci**— Internet of Things, ESP32, HC-SR04, Wokwi, Simulasi

*Keywords— Internet of Things, ESP32, HC-SR04, Wokwi, Simulasi*

## **1. Introduction (Pendahuluan)**

### **1.1 Latar Belakang**

Internet of Things (IoT) semakin berkembang dan memungkinkan berbagai perangkat untuk saling terhubung dan bertukar data. ESP32 merupakan salah satu mikrokontroler yang banyak digunakan dalam proyek IoT karena memiliki konektivitas WiFi dan Bluetooth. Salah satu aplikasi IoT yang umum adalah pemantauan jarak menggunakan sensor Ultrasonic HC-SR04. Dalam praktikum ini, simulasi dilakukan menggunakan platform Wokwi untuk memberikan pengalaman langsung kepada mahasiswa dalam memahami cara kerja sensor tanpa perlu menggunakan perangkat fisik.

### **1.2 Tujuan**

1. Memahami cara menghubungkan sensor Ultrasonic HC-SR04 dengan ESP32.
2. Menggunakan Wokwi dan VSCode untuk melakukan simulasi pengukuran jarak.
3. Menampilkan hasil pembacaan jarak dalam cm dan inch di serial monitor secara real-time.

## **2. Methodology (Metodologi)**

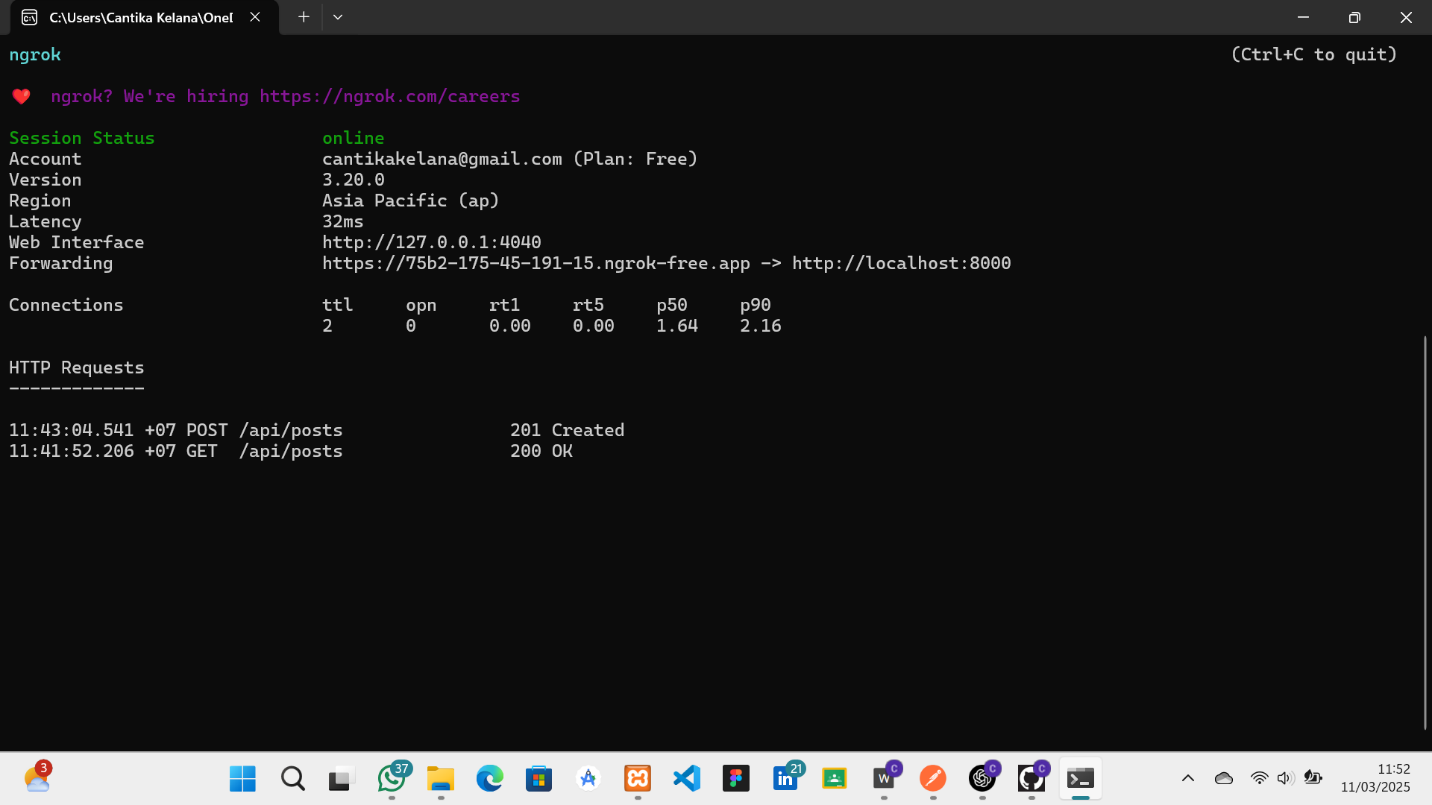
### **2.1 Tools & Materials (Alat dan Bahan)**

* **Framework :** Laravel 11
* **Database :** MySQL (phpMyAdmin)
* **Server :** Laravel built-in server (php artisan serve)
* **Testing Tools :** Postman
* **Ngrok :** Untuk menghubungkan API ke internet

### **2.2 Implementation Steps (Langkah Implementasi)**

* **Membuat Database**
  + Buat database di phpMyAdmin dengan nama iot\_25.
* **Membuat Model dan Migration**
  + Jalankan perintah berikut:
  + php artisan make:model TransaksiSensor -m
  + Edit file 2025\_02\_21\_074123\_create\_transaksi\_sensors\_table.php sesuai kebutuhan.
  + Jalankan perintah migrasi:
  + php artisan migrate
* **Membuat Resource Controller**
  + Jalankan perintah:
  + php artisan make:controller Api/TransaksiSensorController --resource
  + Implementasikan metode CRUD di controller.
* **Membuat Resource untuk API**
  + Jalankan perintah:
  + php artisan make:resource TransaksiSensorResource
  + Sesuaikan format respons JSON di file TransaksiSensorResource.php.
* **Menentukan Routes API**
  + Buka file routes/api.php dan tambahkan route berikut:
  + Route::apiResource('/posts', App\Http\Controllers\Api\TransaksiSensorController::class);
  + Periksa daftar routes dengan:
  + php artisan route:list
* **Menjalankan Server dan Menguji API dengan Postman**
  + Jalankan Laravel dengan:
  + php artisan serve
  + Gunakan Postman untuk mengakses http://127.0.0.1:8000/api/posts.
* **Menggunakan Ngrok untuk Publikasi API**
  + Install dan login ke Ngrok.
  + Jalankan perintah:
  + ngrok http http://localhost:8000
  + Gunakan URL publik yang diberikan oleh Ngrok untuk mengakses API dari luar jaringan lokal.

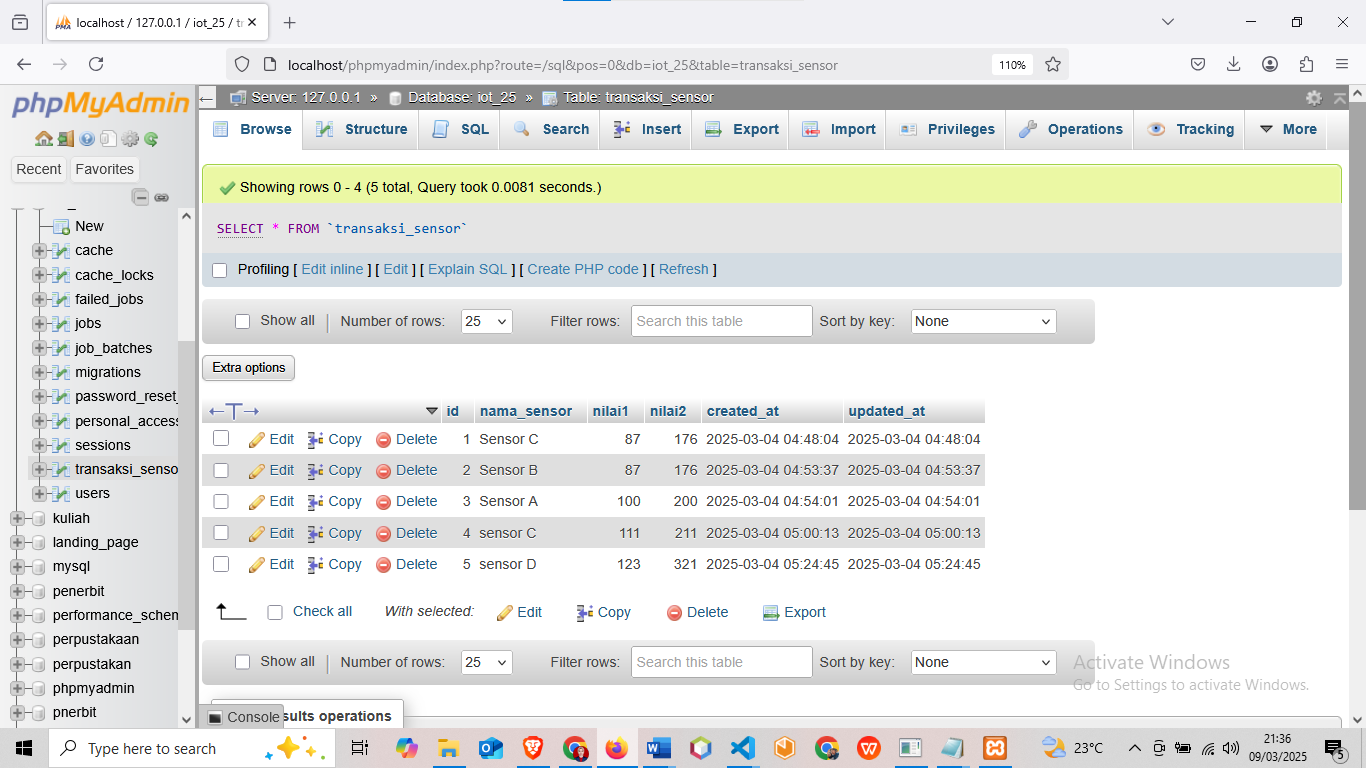
**Screenshot hasil simulasi:**



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

### **3.1 Experimental Results (Hasil Eksperimen)**.

1. API berhasil dibuat dan diuji menggunakan Postman.
2. Endpoint GET, POST, PUT, dan DELETE berfungsi dengan baik.
3. API dapat diakses secara publik menggunakan URL Ngrok.



## **4. Appendix (Lampiran)**

### **4.1 Kode Program**

{

"data": [

{

"id": 3,

"nama\_sensor": "Sensor B",

"nilai1": 87,

"nilai2": 176

},

{

"id": 2,

"nama\_sensor": "Sensor A",

"nilai1": 100,

"nilai2": 200

}

]

}