Laporan Praktikum Simulasi Relay, Button & LED, Simulasi Sensor Jarak (Ultrasonic), Pembuatan API Menggunakan Laravel 11 dan Ngrok

Riyanti Teresa Br Situmeang
Fakultas Vokasi, Universitas Brawijaya
Email: riyantiteresa14@gmail.com

1. PENDAHULUAN

1.1 Latar Belakang

Internet of Things (IoT) adalah konsep di mana objek fisik (perangkat, sensor, atau alat) diintegrasikan dengan kemampuan komunikasi dan pemrosesan data sehingga dapat saling berinteraksi melalui jaringan internet. Tujuan utamanya adalah untuk mengumpulkan, mengirim, dan memproses data secara otomatis sehingga menghasilkan informasi yang dapat membantu dalam pengambilan keputusan secara efisien.

Pada praktikum ini, dilakukan tiga eksperimen utama yang berfokus pada implementasi dan simulasi perangkat IoT:

1. Simulasi Relay, Button & LED – Praktikum ini bertujuan untuk memahami bagaimana cara kerja relay, button, dan LED dalam sebuah sistem berbasis ESP32. Relay digunakan untuk mengontrol perangkat listrik, sedangkan button dan LED berfungsi sebagai input dan output dalam sistem kendali.
2. Simulasi Sensor Jarak (Ultrasonic) – Eksperimen ini bertujuan untuk memahami cara kerja sensor jarak ultrasonik dalam mengukur jarak suatu objek. Sensor ini sering digunakan dalam sistem otomatisasi seperti robotika dan kendaraan pintar.
3. Pembuatan API Menggunakan Laravel 11 dan Ngrok – Praktikum ini berfokus pada pengembangan antarmuka pemrograman aplikasi (API) menggunakan framework Laravel 11 serta penggunaan Ngrok untuk mengakses API secara publik. API yang dibuat berfungsi sebagai penghubung antara sistem IoT dan server berbasis web.

1.2 Tujuan Eksperimen

1. Memahami konsep dasar dan implementasi relay, button, dan LED dalam sistem berbasis ESP32.
2. Mengaplikasikan sensor ultrasonik untuk mengukur jarak objek dalam suatu sistem IoT.
3. Mengembangkan API menggunakan Laravel 11 sebagai backend yang dapat digunakan untuk menghubungkan perangkat IoT dengan sistem berbasis web.
4. Menggunakan Ngrok untuk memungkinkan akses API secara publik dan menguji komunikasi antara perangkat IoT dan server berbasis cloud.
5. Mengintegrasikan berbagai komponen IoT dengan pemrograman berbasis mikrocontroller dan sistem berbasis web untuk mendukung berbagai kebutuhan otomatisasi.

2. METODOLOGI

2.1 Alat dan Bahan

2.1.1 Simulasi Relay, Button & LED

- Mikrokontroler ESP32
- Relay, LED, dan Push Button
- Wokwi Simulator
- VsCode

2.1.2 Simulasi Sensor Jarak (Ultrasonic)

- Mikrokontroler ESP32
- Sensor Ultrasonik HC-SR04
- Wokwi Simulator
- VsCode

2.1.3 Pembuatan API Menggunakan Laravel 11 dan Ngrok

- Postman untuk pengujian API
- Laravel 11
- Ngrok
- phpMyAdmin dan MySQL
- VsCode

2.2 Langkah Implementasi

2.2.1 Simulasi Relay, Button & LED

- Buka web wokwi.com dan buat diagram yang berisi ESP32, Relay, LED dan Push Botton
- Buat Codingan di sketch.ino yang ada di wokwi.com
- Buka VsCode, dan buat project baru di platform io
- Isi Parameter Name nya sesuai keinginan sendiri, Parameter Board isi DOIT ESP32 DEVKIT V1 dan parameter Frameworknya adalah Arduino.
- Lalu salin coding yang sudah dibuat di platform wokwi.com ke file main.cpp
- Edit file platformio.ini menjadi seperti ini :
  [env:esp32doit-devkit-v1]
  platform = espressif32
  board = esp32doit-devkit-v1
  framework = arduino
- Buat file baru diagram.json di VsCode , dan copy paste diagram.json pada platform online wokwi.com ke diagram.json di VsCode
- Buat file baru wokwi.toml, dan isikan file tersebut dengan koding sebagai berikut :
  [wokwi]
  version = 1
  firmware = '.pio\build\esp32doit-devkit-v1\firmware.bin'
  elf = '.pio\build\esp32doit-devkit-v1\firmware.elf'
- Langkah berikutnya lakukan compile pada file main.cpp
- Langkah berikutnya lakukan request license ke wokwi.com
- Klik tombol Get Your License
- Klik Open
- Langkah terakhir jalankan simulasi dengan mengetik perintah: >Wokwi: Start Simulator

2.2.2 Simulasi Sensor Jarak (Ultrasonic)

- Buka web wokwi.com dan buat diagram yang berisi ESP32, Sensor Ultrasonik HC-SR04
- Buat Codingan di sketch.ino yang ada di wokwi.com
- Buka VsCode, dan buat project baru di platform io
- Isi Parameter Name nya sesuai keinginan sendiri, Parameter Board isi DOIT ESP32 DEVKIT V1 dan parameter Frameworknya adalah Arduino.
- Lalu salin coding yang sudah dibuat di platform wokwi.com ke file main.cpp
- Edit file platformio.ini menjadi seperti ini :
  [env:esp32doit-devkit-v1]
  platform = espressif32
  board = esp32doit-devkit-v1
  framework = arduino
- Buat file baru diagram.json di VsCode , dan copy paste diagram.json pada platform online wokwi.com ke diagram.json di VsCode
- Buat file baru wokwi.toml, dan isikan file tersebut dengan koding sebagai berikut :
  [wokwi]
  version = 1
  firmware = '.pio\build\esp32doit-devkit-v1\firmware.bin'
  elf = '.pio\build\esp32doit-devkit-v1\firmware.elf'
- Langkah berikutnya lakukan compile pada file main.cpp
- Langkah berikutnya lakukan request license ke wokwi.com
- Klik tombol Get Your License
- Klik Open
- Langkah terakhir jalankan simulasi dengan mengetik perintah: >Wokwi: Start Simulator

2.2.3 Pembuatan API Menggunakan Laravel 11 dan Ngrok

- Menginstal dan mengkonfigurasi Laravel 11
- Mengaktifkan mysql di xampp
- Buat database di phpmyadmin dengan nama **iot_25**
- Ubah isi konfigurasi file .env

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=iot_25
DB_USERNAME=root
DB_PASSWORD=
```

- Buat file model **TransaksiSensor.php** dengan cara menjalankan perintah berikut di terminal :
  php artisan make:model TransaksiSensor -m
- Ubah file 2025_03_07_131008_create_personal_access_tokens_table.php

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('personal_access_tokens', function (Blueprint $table) {
            $table->id();
            $table->morphs('tokenable');
            $table->string('name');
            $table->string('token', 64)->unique();
            $table->text('abilities')->nullable();
            $table->timestamp('last_used_at')->nullable();
            $table->timestamp('expires_at')->nullable();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('personal_access_tokens');
    }
};
```

- ubah isi file **app/Models/TransaksiSensor.php**

```php
<?php


namespace App\Models;


use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;


class TransaksiSensor extends Model
{
    use HasFactory;

    /**
     * The table associated with the model.
     *
     * @var string
     */
    protected $table = 'transaksi_sensor';


    /**
     * The attributes that are mass assignable.
     *
     * @var array
     */
    protected $fillable = [
        'nama_sensor',
        'nilai1',
        'nilai2',
    ];


    /**
     * The attributes that should be hidden for arrays.
     *
     * @var array
     */
    protected $hidden = [];


    /**
     * The attributes that should be cast.
     *
     * @var array
     */
    protected $casts = [];
```

```
}
```

- Kemudian jalankan perintah **php artisan migrate** untuk membuat tabel
- Buat Resource dengan menjalankan perintah **php artisan make:resource TransaksiSensorResource**
- Ubah isi file **TransaksiSensorResource.php** yang ada di folder app-Http-Resources

```php
<?php


namespace App\Http\Resources;
use Illuminate\Http\Request;
use Illuminate\Http\Resources\Json\JsonResource;


class TransaksiSensorResource extends JsonResource
{
    /**
     * Transform the resource into an array.
     *
     * @param  \Illuminate\Http\Request  $request
     * @return array
     */
    public function toArray($request)
    {
        return [
            'id' => $this->id,
            'nama_sensor' => $this->nama_sensor,
            'nilai1' => $this->nilai1,
            'nilai2' => $this->nilai2,
        ];
    }
}
```

- Buat API controller dengan menjalankan perintah **php artisan make:controller Api/TransaksiSensorController**
- Ubah isi file **app/Http/Controllers/Api/TransaksiSensorController.php**

```php
<?php


namespace App\Http\Controllers\Api;
use Illuminate\Http\Request;


use App\Models\TransaksiSensor;
use App\Http\Controllers\Controller;
use App\Http\Resources\TransaksiSensorResource;


class TransaksiSensorController extends Controller
{
```

```php
    /**
     * index
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        // Get all transactions from TransaksiSensor model, paginated
        $transaksiSensors = TransaksiSensor::latest()->paginate(5);


        // Return a collection of transactions as a resource
        return TransaksiSensorResource::collection($transaksiSensors);
    }


    /**
 * Store a newly created resource in storage.
 *
 * @param  \Illuminate\Http\Request  $request
 * @return \Illuminate\Http\Response
 */
public function store(Request $request)
{
    $validatedData = $request->validate([
        'nama_sensor' => 'required|string|max:255',
        'nilai1' => 'required|integer',
        'nilai2' => 'required|integer',
    ]);


    $transaksiSensor = TransaksiSensor::create($validatedData);


    return new TransaksiSensorResource($transaksiSensor);
}
/**
 * Display the specified resource.
 *
 * @param  int  $id
 * @return \Illuminate\Http\Response
 */
public function show($id)
{
    $transaksiSensor = TransaksiSensor::findOrFail($id);


    return new TransaksiSensorResource($transaksiSensor);
```

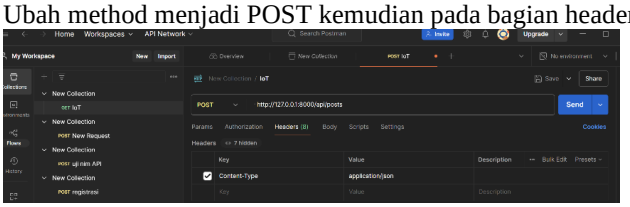
```
}

/**
 * Update the specified resource in storage.
 *
 * @param  \Illuminate\Http\Request  $request
 * @param  int  $id
 * @return \Illuminate\Http\Response
 */
public function update(Request $request, $id)
{
    $validatedData = $request->validate([
        'nama_sensor' => 'required|string|max:255',
        'nilai1' => 'required|integer',
        'nilai2' => 'required|integer',
    ]);


    $transaksiSensor = TransaksiSensor::findOrFail($id);
    $transaksiSensor->update($validatedData);


    return new TransaksiSensorResource($transaksiSensor);
}
/**
 * Remove the specified resource from storage.
 *
 * @param  int  $id
 * @return \Illuminate\Http\Response
 */
public function destroy($id)
{
    $transaksiSensor = TransaksiSensor::findOrFail($id);
    $transaksiSensor->delete();


    return response()->json(['message' => 'Deleted successfully'], 204);
}


}
```

- Buat route khusus API dengan menjalankan perintah **php artisan install:api**
- Buka file **routes/api.php** dan ubah isi file menjadi

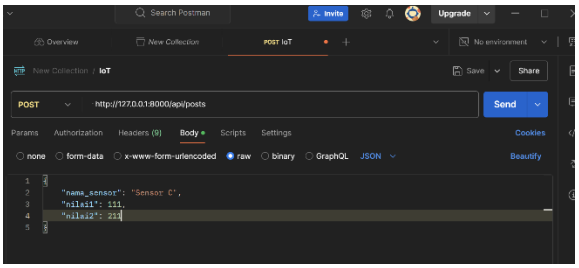
```
<?php

use Illuminate\Auth\Middleware\Authenticate;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Route;
```

```php
Route::get('/user', function (Request $request) {
    return $request->user();
})->middleware(Authenticate::using('sanctum'));

//posts
Route::apiResource('/posts', App\Http\Controllers\Api\TransaksiSensorController::class);
```

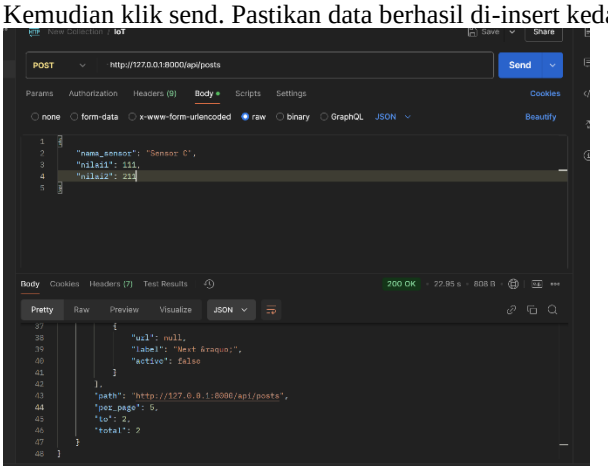
- Kemudian pastikan routes telah terbentuk dengan menjalankan perintah **php artisan route:list**
- Download dan installasi Postman
- Untuk melakukan percobaan akses api, pastikan aplikasi laravel dijalankan dengan perintah **php artisan serve**
- Pastikan ada beberapa data yang dimasukkan kedalam tabel di database iot_25.
- Untuk mengambil data diatas melalui aplikasi postman, jalankan prosedur berikut :
    - Pada bagian URL masukkan alamat server laravel **http://127.0.0.1:8000/api/posts** Atau bisa diakses melalui url : **http://localhost:8000/api/posts**
    - Pilih method GET untuk mengambil data dari database , kemudian klik tombol SEND
    - Pastikan data dikembalikan dalam bentuk json
- Selanjutnya melakukan percobaan insert data ke tabel di database menggunakan API. Ubah method menjadi POST kemudian pada bagian header ubah menjadi



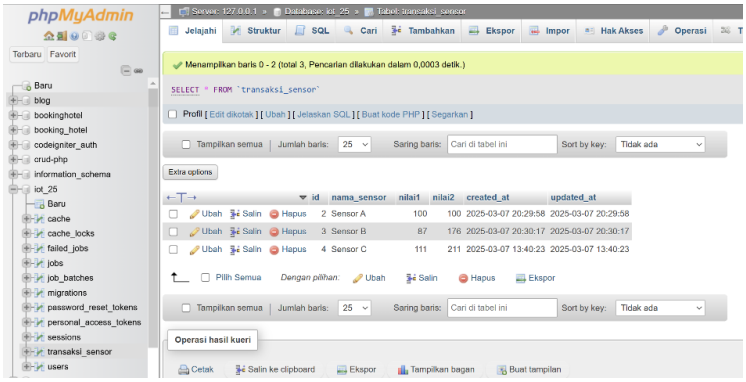
- Pada bagian body ubah menjadi sebagai berikut



- Kemudian klik send. Pastikan data berhasil di-insert kedatabase seperti tampilan ini



- Cek manual di phpmyadmin, pastikan data baru masuk

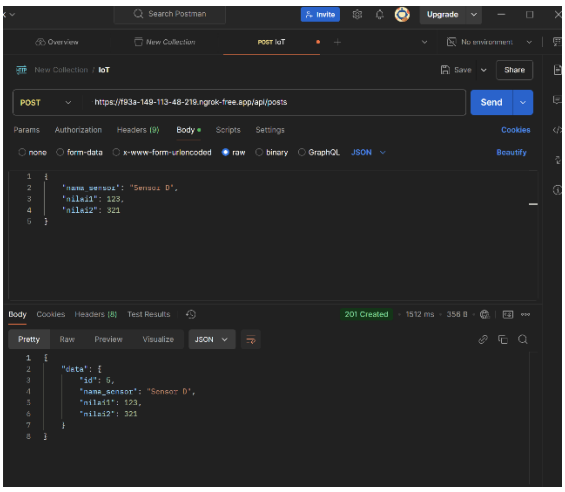


- Download dan install aplikasi ngrok
- Login ke web ngrok, kemudian download aplikasi ngrok sesuai sistem operasi
- Lakukan ekstraksi
- Buka command prompt dari alamat folder ekstraksi
- Kemudian jalankan perintah sesuai yang ada di akun ngrok

```
C:\Users\ASUS\Downloads\ngrok-v3-stable-windows-amd64>ngrok config add-authtoken 2tzVXX473YTrgTfqvWz169NRYAc_37zxDam9TbkLzBzPoaCU
Authtoken saved to configuration file: C:\Users\ASUS\AppData\Local/ngrok/ngrok.yml

C:\Users\ASUS\Downloads\ngrok-v3-stable-windows-amd64>
```

- Kemudian jalankan perintah berikut **ngrok http http://localhost:8000** untuk mengonline kan laravel melalui port 8000

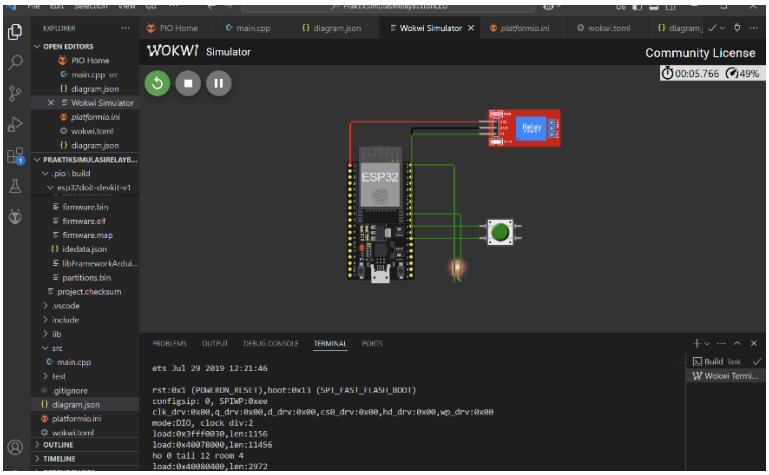
- Untuk melakukan percobaan GET api , maka URL harus ditambahkan alamat endpoint menjadi sebagai berikut
  https://f93a-149-113-48-219.ngrok-free.app/**api/posts**
- Ubah method menjadi POST dan parameter header dan body sesuaikan



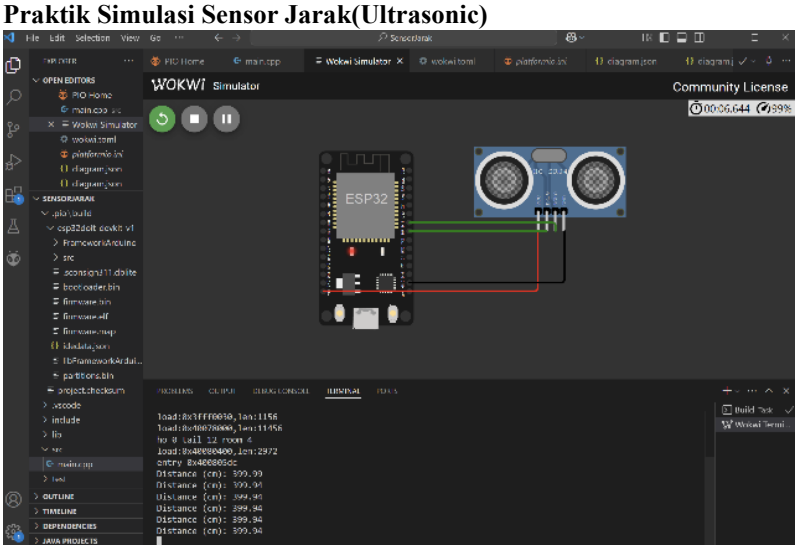
- Sampai disini API yang dibangun menggunakan laravel sudah dapat berjalan dengan baik dan dapat diakses melalui URL public

3. HASIL EKSPERIMEN

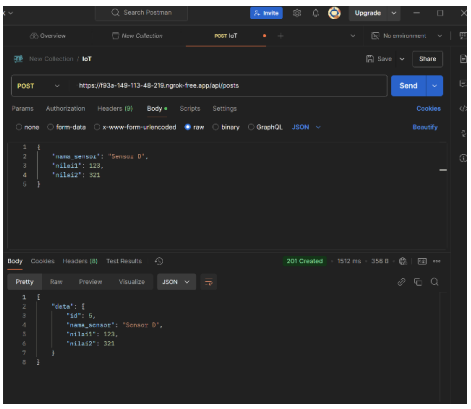
3.1 Praktik Simulasi Relay, Button & LED

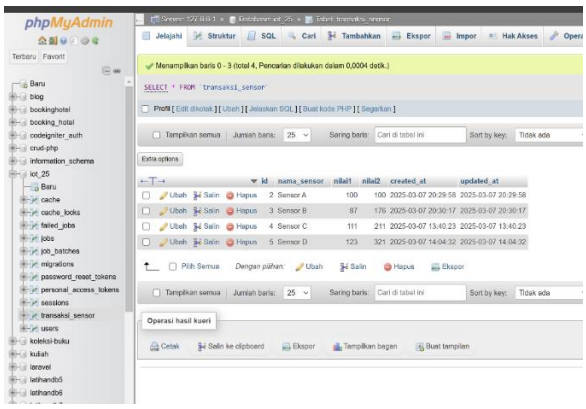


3.2 Praktik Simulasi Sensor Jarak(Ultrasonic)



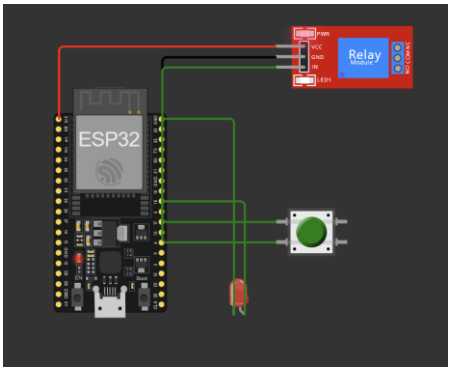
3.3 Praktik Pembuatan API Menggunakan Laravel 11 dan Ngrok





4. LAMPIRAN

4.1 Lampiran Praktik Simulasi Relay, Button & LED

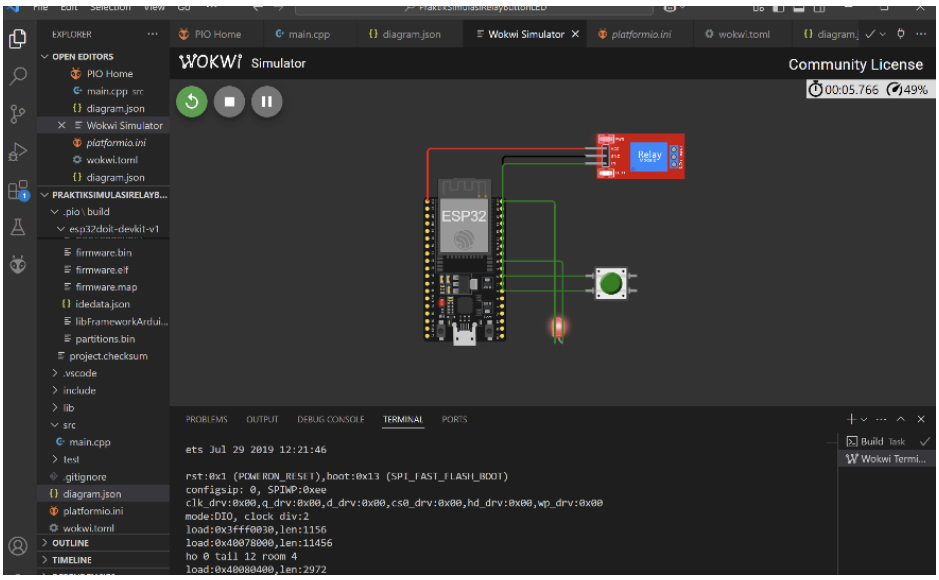


- **main.cpp**

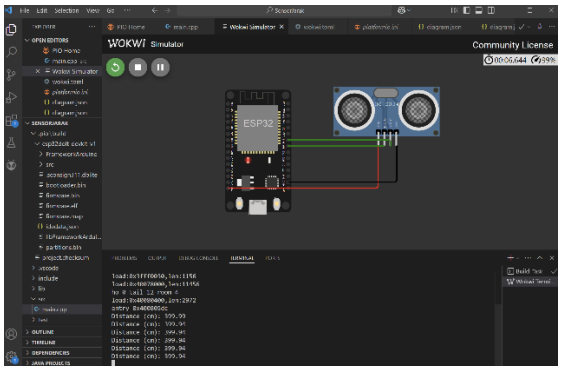
```
#include <Arduino.h>

// Define pin numbers
const int ButtonPin = 19;  // GPIO19 connected to the pushbutton
const int LedPin = 18;     // GPIO18 connected to the LED
const int RelayPin = 23;   // GPIO23 connected to the relay module

void setup() {
  // Set pin modes
  pinMode(ButtonPin, INPUT_PULLUP);  // Set the button pin as an input with an
internal pull-up resistor
  pinMode(LedPin, OUTPUT);          // Set the LED pin as an output
  pinMode(RelayPin, OUTPUT);         // Set the relay pin as an output

  // Initialize the outputs to be OFF
  digitalWrite(LedPin, LOW);
  digitalWrite(RelayPin, LOW);
}

void loop() {
  // Read the state of the button
  int buttonState = digitalRead(ButtonPin);

  // Check if the button is pressed
  // Since the button is wired to pull the pin LOW when pressed, we check for LOW
  if (buttonState == LOW) {
    digitalWrite(LedPin, HIGH);     // Turn on the LED
    digitalWrite(RelayPin, HIGH);   // Turn on the relay
  } else {
    digitalWrite(LedPin, LOW);      // Turn off the LED
    digitalWrite(RelayPin, LOW);    // Turn off the relay
  }
}
```

- **diagram.json**

```
{
  "version": 1,
  "author": "Anonymous maker",
  "editor": "wokwi",
  "parts": [
    { "type": "board-esp32-devkit-c-v4", "id": "esp", "top": -9.6, "left": -110.36, "attrs": {} },
    { "type": "wokwi-relay-module", "id": "relay1", "top": -67, "left": 96, "attrs": {} },
    {
      "type": "wokwi-pushbutton",
      "id": "btn1",
      "top": 102.2,
      "left": 96,
      "attrs": { "color": "green", "xray": "1" }
    },
    { "type": "wokwi-led", "id": "led1", "top": 159.6, "left": 42.2, "attrs": { "color": "red" } }
  ],
  "connections": [
    [ "esp:TX", "$serialMonitor:RX", "", [] ],
    [ "esp:RX", "$serialMonitor:TX", "", [] ],
    [ "relay1:VCC", "esp:3V3", "red", [ "h0" ] ],
    [ "relay1:GND", "esp:GND.2", "black", [ "h0" ] ],
    [ "relay1:IN", "esp:23", "green", [ "h0" ] ],
    [ "btn1:2.l", "esp:GND.2", "green", [ "h0" ] ],
    [ "btn1:1.l", "esp:19", "green", [ "h0" ] ],
    [ "led1:C", "esp:GND.2", "green", [ "v0" ] ],
    [ "led1:A", "esp:18", "green", [ "v0" ] ]
  ],
  "dependencies": {}
}
```



4.2 Praktik Simulasi Sensor Jarak (Ultrasonic)



- **main.cpp**

```
#include <Arduino.h>

const int trigPin = 5;
const int echoPin = 18;


//define sound speed in cm/uS
#define SOUND_SPEED 0.034
#define CM_TO_INCH 0.393701


long duration;
float distanceCm;
float distanceInch;


void setup() {
 Serial.begin(115200); // Starts the serial communication
 pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
 pinMode(echoPin, INPUT); // Sets the echoPin as an Input
}


void loop() {
 // Clears the trigPin
 digitalWrite(trigPin, LOW);
 delayMicroseconds(2);
 // Sets the trigPin on HIGH state for 10 micro seconds
 digitalWrite(trigPin, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigPin, LOW);
  // Reads the echoPin, returns the sound wave travel time in microseconds
 duration = pulseIn(echoPin, HIGH);
  // Calculate the distance
 distanceCm = duration * SOUND_SPEED/2;
  // Convert to inches
```

```
distanceInch = distanceCm * CM_TO_INCH;
 // Prints the distance in the Serial Monitor
Serial.print("Distance (cm): ");
Serial.println(distanceCm);
// Serial.print("Distance (inch): ");
// Serial.println(distanceInch);
 delay(1000);
}
```

- **diagram.json**

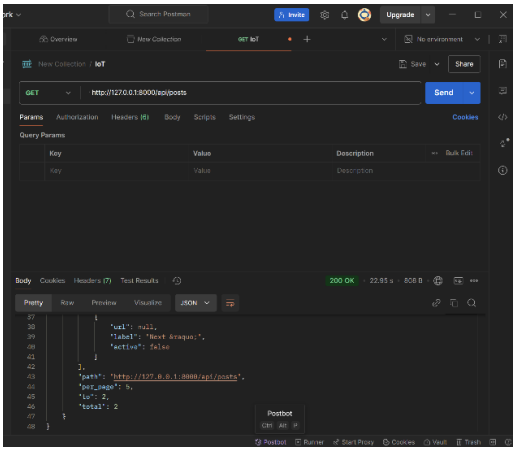
```
{
  "version": 1,
  "author": "Uri Shaked",
  "editor": "wokwi",
  "parts": [
    { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": -52.9, "left": -5, "attrs": {} },
    { "type": "wokwi-hc-sr04", "id": "ultrasonic1", "top": -56.1, "left": 168.7, "attrs": {} }
  ],
  "connections": [
    [ "esp:TX0", "$serialMonitor:RX", "", [] ],
    [ "esp:RX0", "$serialMonitor:TX", "", [] ],
    [ "ultrasonic1:VCC", "esp:VIN", "red", [ "v0" ] ],
    [ "ultrasonic1:TRIG", "esp:D5", "green", [ "v0" ] ],
    [ "ultrasonic1:ECHO", "esp:D18", "green", [ "v0" ] ],
    [ "ultrasonic1:GND", "esp:GND.1", "black", [ "v0" ] ]
  ],
  "dependencies": {}
}
```

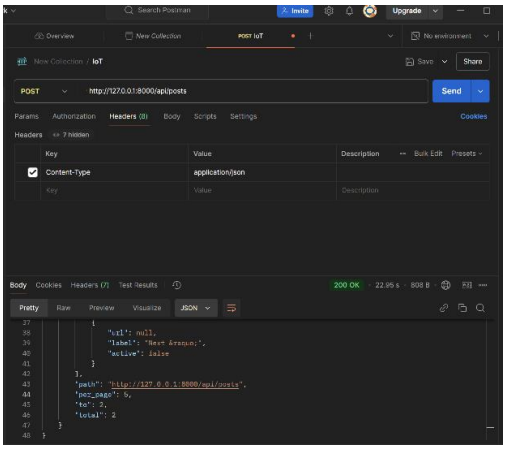
4.3 Praktik Pembuatan API Menggunakan Laravel 11 dan Ngrok

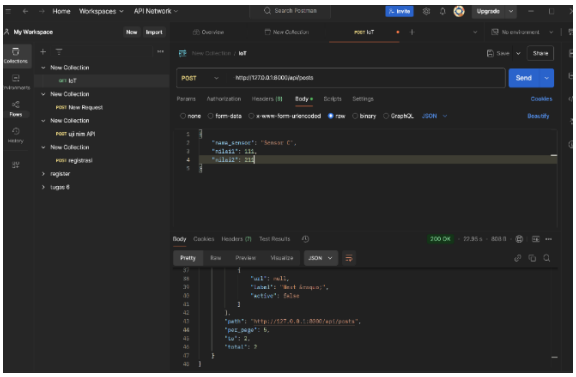
```
PS C:\xampp\htdocs\pembuatan_api> php artisan migrate
PHP Warning:  Module "openssl" is already loaded in Unknown on line 0

Warning: Module "openssl" is already loaded in Unknown on line 0

  INFO  Preparing database.

  Creating migration table ..... 17.57ms DONE

  INFO  Running migrations.

  0001_01_01_000000_create_users_table ..... 102.01ms DONE
  0001_01_01_000001_create_cache_table ..... 17.35ms DONE
  0001_01_01_000002_create_jobs_table ..... 88.38ms DONE
  2025_03_07_125414_create_transaksi_sensors_table ..... 49.97ms DONE
```

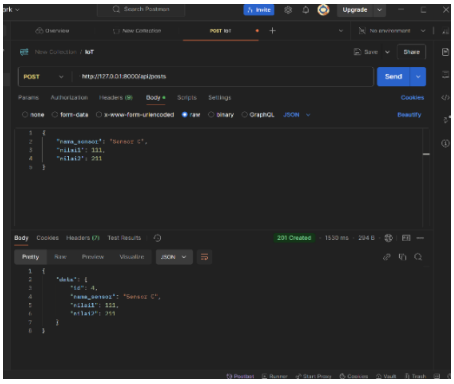
```
PS C:\xampp\htdocs\pembuatan_api> php artisan make:resource TransaksiSensorResource
PHP Warning:  Module "openssl" is already loaded in Unknown on line 0

Warning: Module "openssl" is already loaded in Unknown on line 0

  INFO  Resource [C:\xampp\htdocs\pembuatan_api\app\Http\Resources\TransaksiSensorResource.php] created successfully.
```

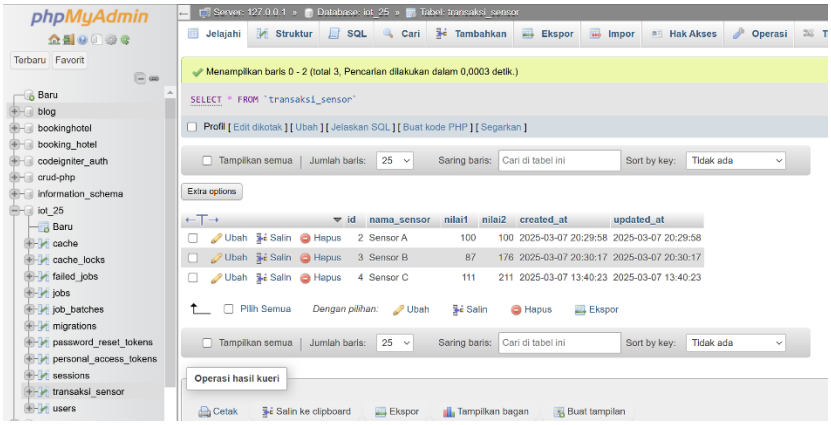
```
GET|HEAD        / ..... 
GET|HEAD        api/posts ..... posts.index › Api\TransaksiSensorController@index
POST            api/posts ..... posts.store › Api\TransaksiSensorController@store
GET|HEAD        api/posts/{post} ..... posts.show › Api\TransaksiSensorController@show
PUT|PATCH       api/posts/{post} ..... posts.update › Api\TransaksiSensorController@update
DELETE          api/posts/{post} ..... posts.destroy › Api\TransaksiSensorController@destroy
GET|HEAD        api/user ..... 
GET|HEAD        sanctum/csrf-cookie ..... sanctum.csrf-cookie › Laravel\Sanctum › CsrfCookieController@show
GET|HEAD        storage/{path} ..... storage.local
GET|HEAD        up ..... 
```





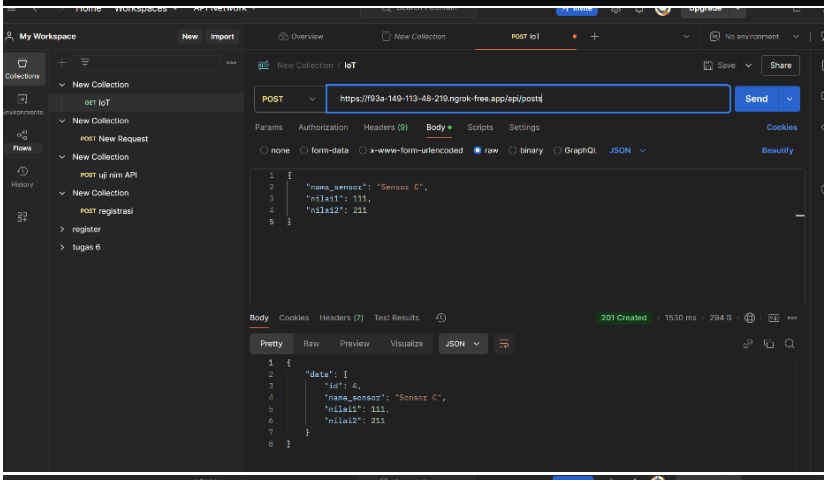


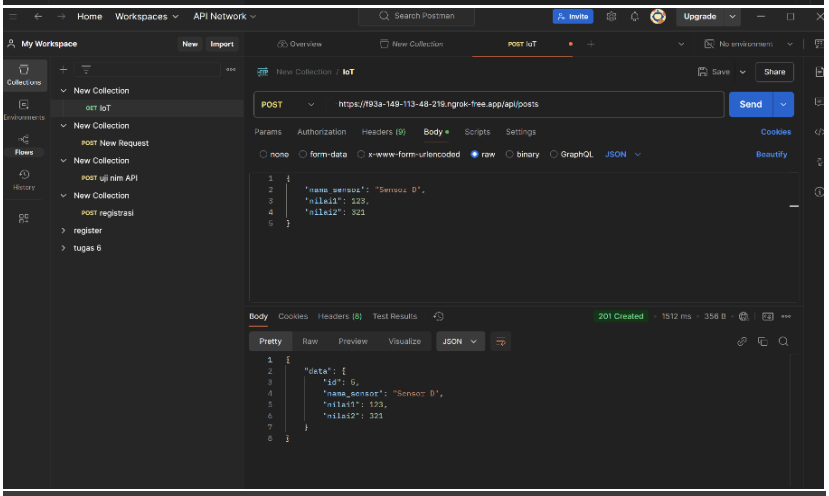


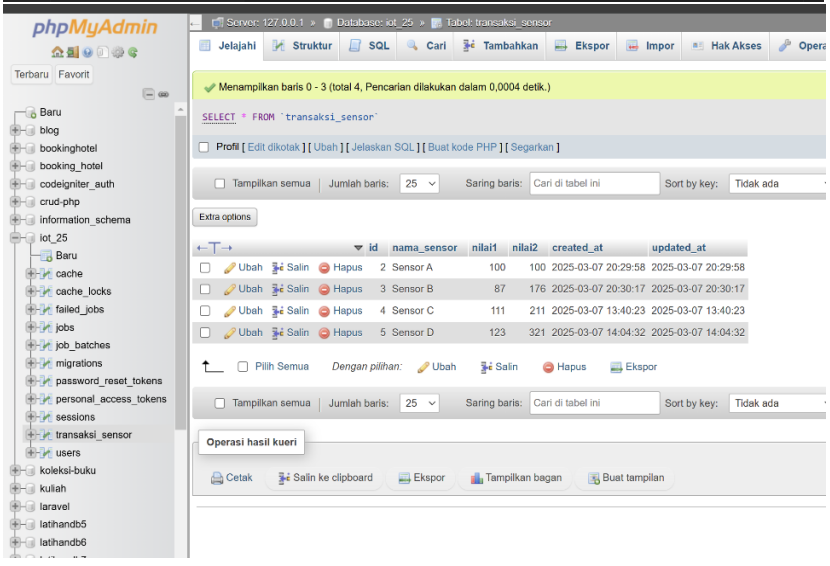



```
C:\Users\ASUS\Downloads\ngrok-v3-stable-windows-amd64>ngrok config add-authtoken 2tzVXX4Y3YTrgTfqvWz169NRYAc_37zxDam9TbkLzBzPoaCU
Authtoken saved to configuration file: C:\Users\ASUS\AppData\Local/ngrok/ngrok.yml

C:\Users\ASUS\Downloads\ngrok-v3-stable-windows-amd64>
```


Command Prompt - ngrok ht
ngrok
(Ctrl+C to quit)
Goodbye tunnels, hello Agent Endpoints: https://ngrok.com/r/aep
Session Status online
Account riyanti14 (Plan: Free)
Version 3.20.0
Region Asia Pacific (ap)
Web Interface http://127.0.0.1:4040
Forwarding https://f93a-149-113-48-219.ngrok-free.app -> http://localhost:8000
Connections ttl opn rt1 rt5 p50 p90
0 0 0.00 0.00 0.00 0.00


POST
https://f93a-149-113-48-219.ngrok-free.app/api/posts
Send
{
"nama_sensor": "Sensor C",
"nilai1": 111,
"nilai2": 211
}
201 Created
{
"data": {
"id": 4,
"nama_sensor": "Sensor C",
"nilai1": 111,
"nilai2": 211
}
}


POST
https://f93a-149-113-48-219.ngrok-free.app/api/posts
Send
{
"nama_sensor": "Sensor D",
"nilai1": 123,
"nilai2": 321
}
201 Created
{
"data": {
"id": 6,
"nama_sensor": "Sensor D",
"nilai1": 123,
"nilai2": 321
}
}


phpMyAdmin
Server: 127.0.0.1 » Database: iot_25 » Tabel: transaksi_sensor
Menampilkan baris 0 - 3 (total 4, Pencarian dilakukan dalam 0,0004 detik.)
SELECT * FROM `transaksi_sensor`
id nama_sensor nilai1 nilai2 created_at updated_at
2 Sensor A 100 100 2025-03-07 20:29:58 2025-03-07 20:29:58
3 Sensor B 87 176 2025-03-07 20:30:17 2025-03-07 20:30:17
4 Sensor C 111 211 2025-03-07 13:40:23 2025-03-07 13:40:23
5 Sensor D 123 321 2025-03-07 14:04:32 2025-03-07 14:04:32