**LAPORAN PRAKTIKUM INTERNET OF THINGS**

**PRAKTIK PEMBUATAN API Laravel**

Fakultas Vokasi, Universitas Brawijaya

**Autors**

Fadhilah Azkia

Fakultas Vokasi, Universitas Brawijaya

Email : [fadhilahazkia95@gmail.com](mailto:fadhilahazkia95@gmail.com)

**Abstract**

Percoobaan ini bertujuan untuk merancang API dan Ngrok dengan media dari Postman. Diguunakannya Ngrok adalah sebagai tunnel yang memungkinkan API yang berjalan dilocalhost dapat di akses public melaluli internet. Praktik ini sangat efektif untuk mengetahui bagaimana API bekerja dan perantara dari Ngrok.

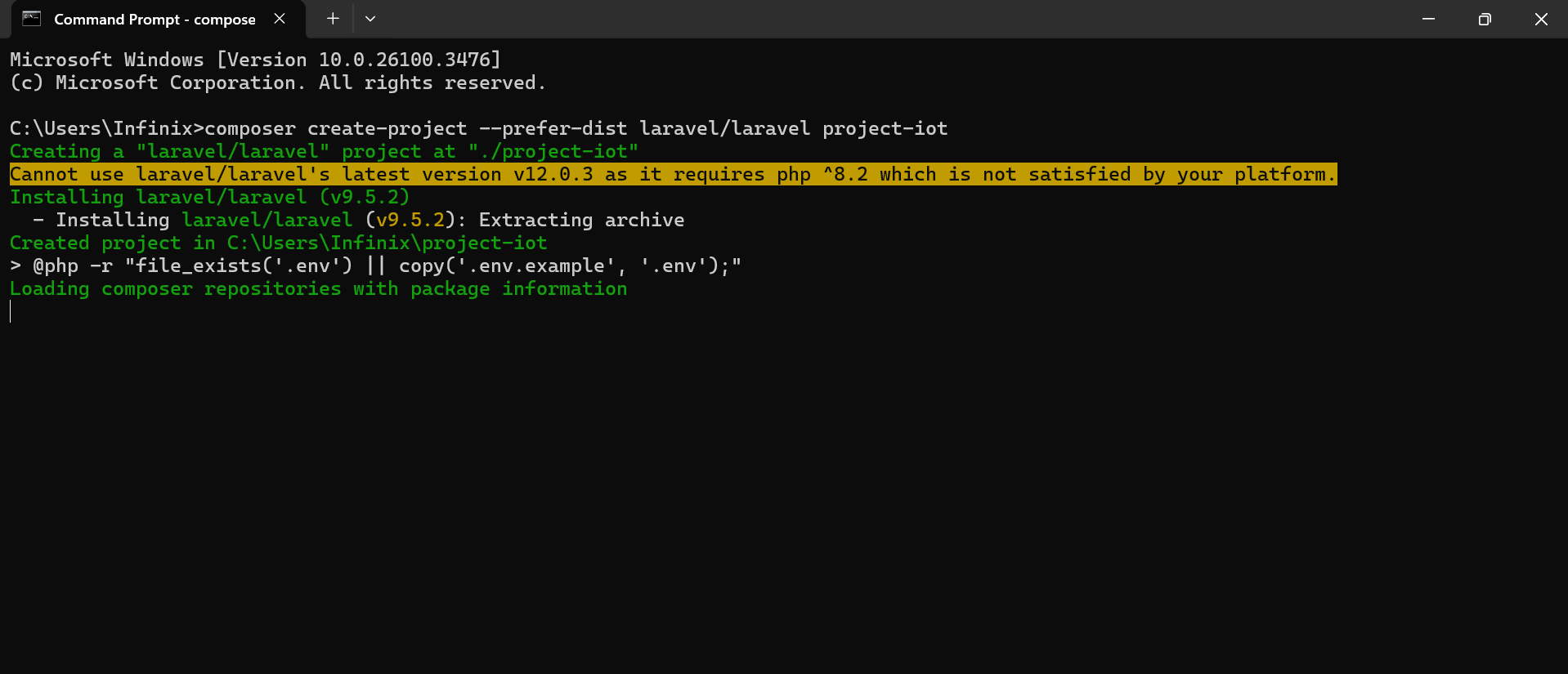
1. **Introduction**  
   1. **Latar Belakang**  
      API adalah mekanisme yang memungkinkan dua komponen perangkat lunak untuk saling berkomunikasi menggunakan serangkaian definisi dan protokol. Pembuatan API menggunakan Laravel 11 diakses melalui jaringan local, kemudian di bantu dengan Ngrok agar dapat diakses public.
   2. **Tujuan Eksperimen**

* Membuat API menggunakan Laravel.
* Mengaplikasikan POST dan GET menggunakan Postman.
* Mengaplikasikan Ngrok agar API dapat diakses public.

1. **Metodologi**
   1. **Alat dan Bahan**

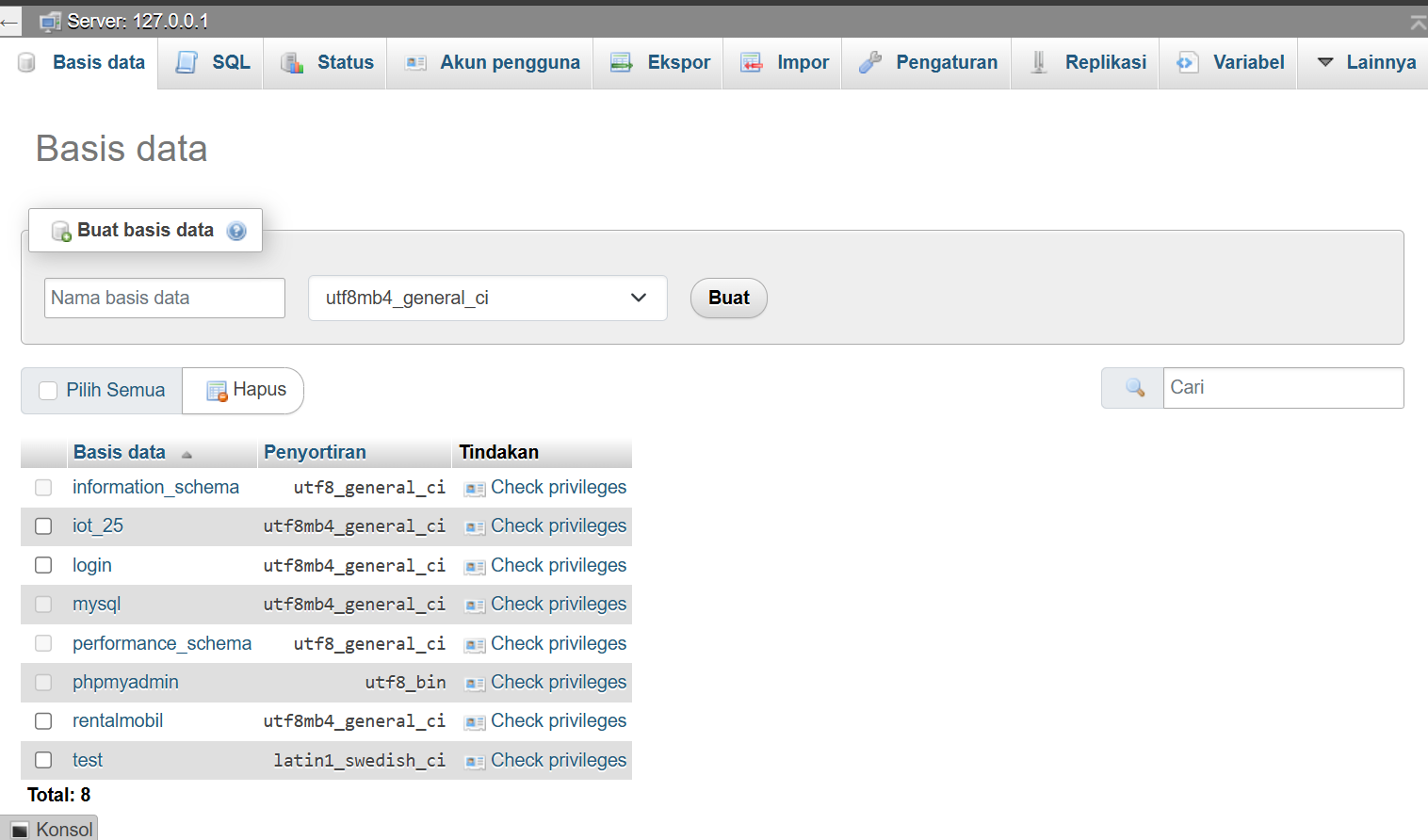
* Hardware : Laptop
* Laravel
* Visual Studio Code
* XAMPP
* Ngrok
  1. **Langkah Implementasi**

1. Membuka Visual Studio Code
2. Install Laravel menggunakan command prompt



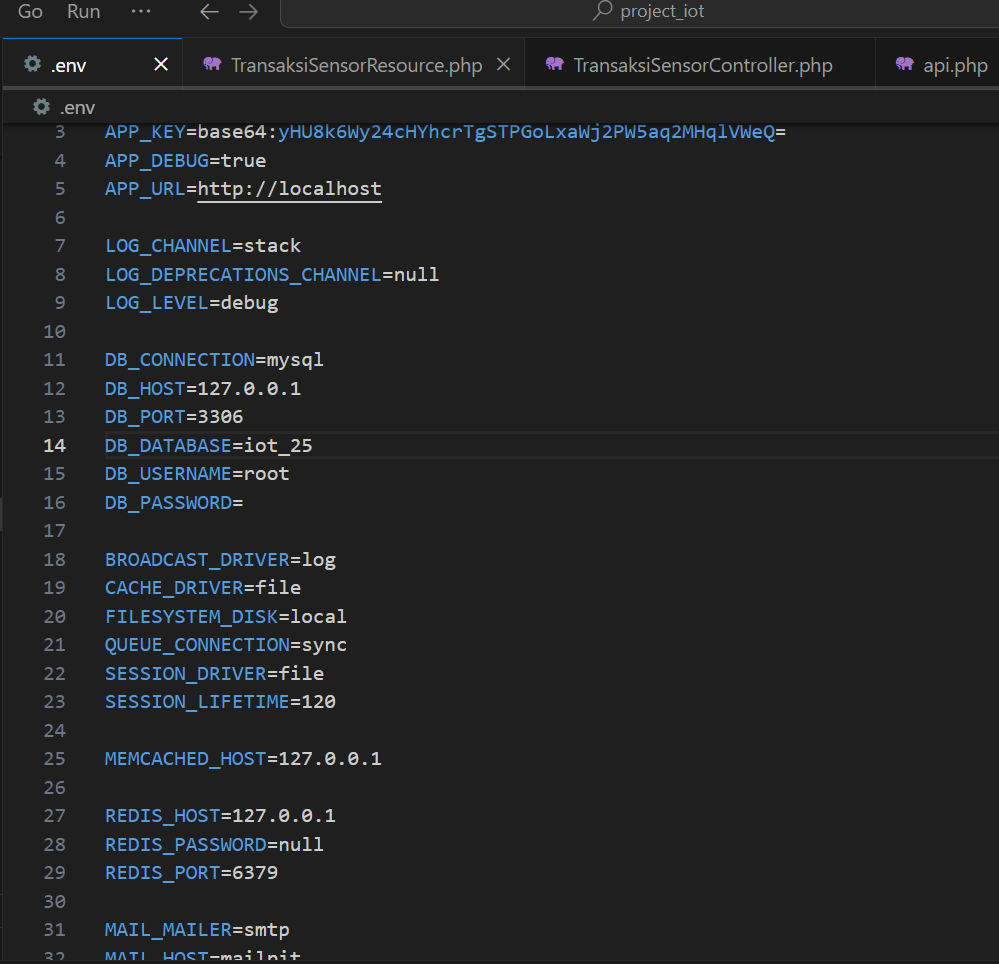
**Gambar 2.2.1** Install Laravel menggunakan command prompt.

1. Selanjutnya membuka phpmyAdmin untuk membuat database.



**Gambar 2.2.2** Halaman phpmyAdmin.

1. Selanjutnya masuk ke VSCode, kemudian sesuaikan nama database kedalam file .env.



**Gambar 2.2.3** file .env.

1. Membuat file model TransaksiSensor.php dengan cara menambahkan perintah berikut di terminal :

**php artisan make:model TransaksiSensor -m**

Kemudian ubah **2025\_03\_14\_080958\_create\_transaksi\_sensors\_table.php** yang ada di folder migrations :

<?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('transaksi\_sensor', function (Blueprint $table) {

            $table->id('id')->startingValue(1); // Menetapkan AUTO\_INCREMENT dimulai dari 1

            $table->string('nama\_sensor', 255); // varchar(255)

            $table->integer('nilai1', false)->length(255); // int(255)

            $table->integer('nilai2', false)->length(255); // int(255)

            $table->timestamps(); // Menambahkan created\_at dan updated\_at

        });

    }

    /\*\*

     \* Reverse the migrations.

     \*/

    public function down(): void

    {

        Schema::dropIfExists('transaksi\_sensors');

    }

};

**Gambar 2.2.4** Gambar device setelah di pasangkan.

1. Kemudian ubah isi file **app/Models/TransaksiSensor.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 = [];

}

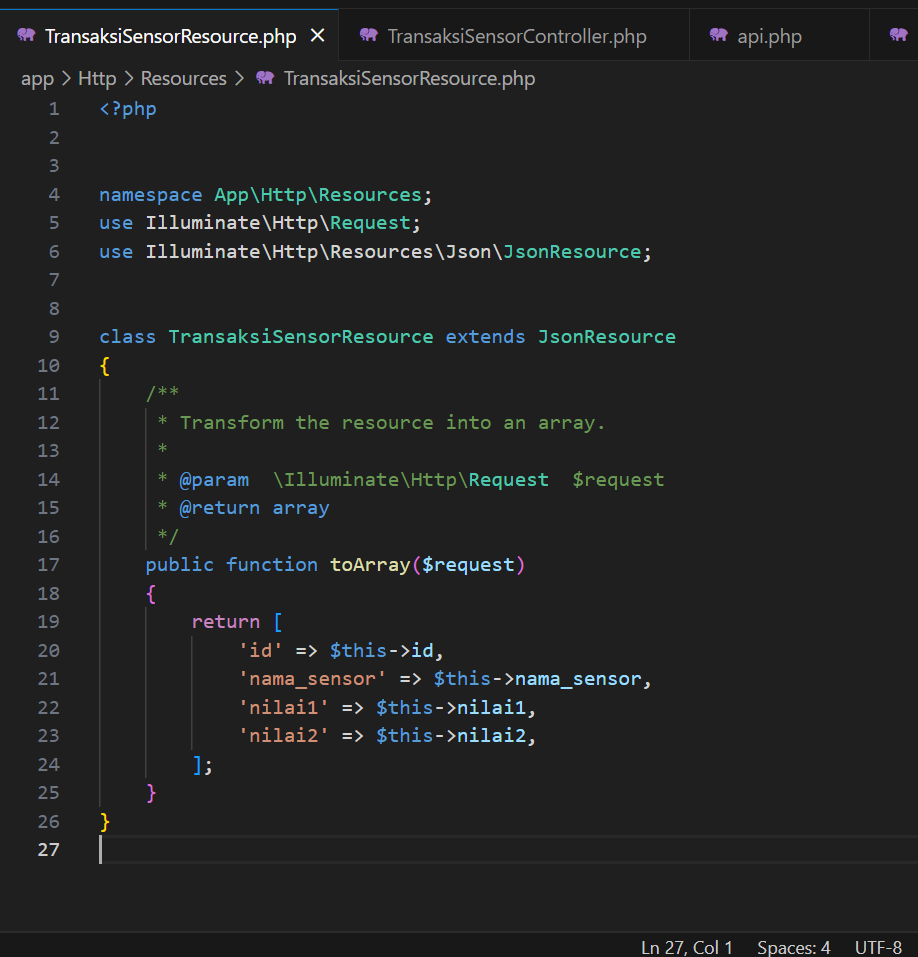
1. Jalankan perintah berikut untuk membuat tabel dari file yang sudah kita buat tadi.



1. Membuat file **App/Http/Resources/TransaksiSensorResource** menggunakan perintah berikut **:**



Dan diubah isinya seperti dibawah ini:



**Gambar 2.2.5** App/Http/Resource/TransaksiSensorResource.

1. Membuat file API controller dengan perintah :



Dan mengubah isinya sesuai dengan yang sudah dituliskan dibawah ini :

<?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

{

    /\*\*

     \* 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)

{

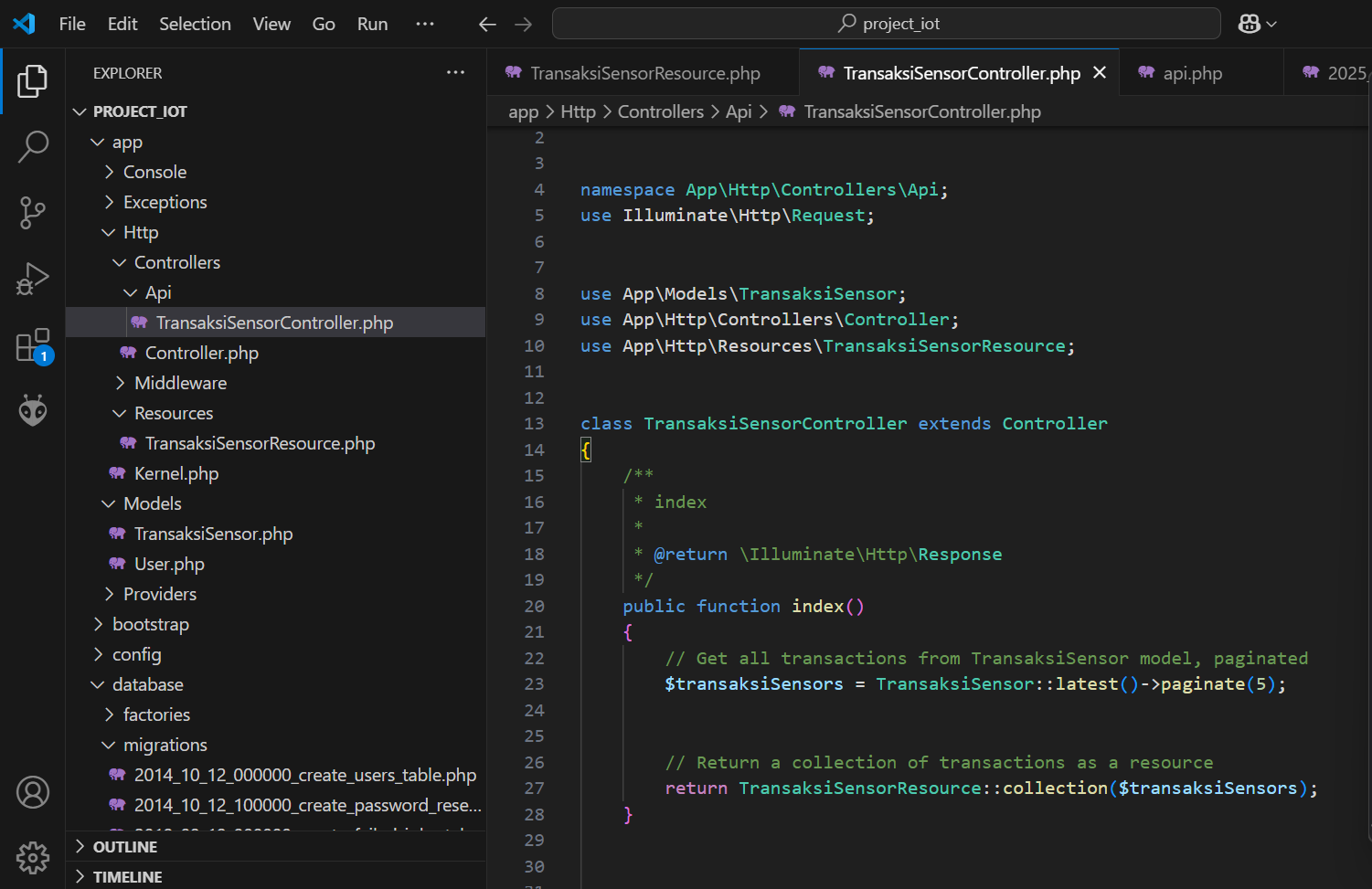
    $transaksiSensor = TransaksiSensor::findOrFail($id);

    $transaksiSensor->delete();

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

}

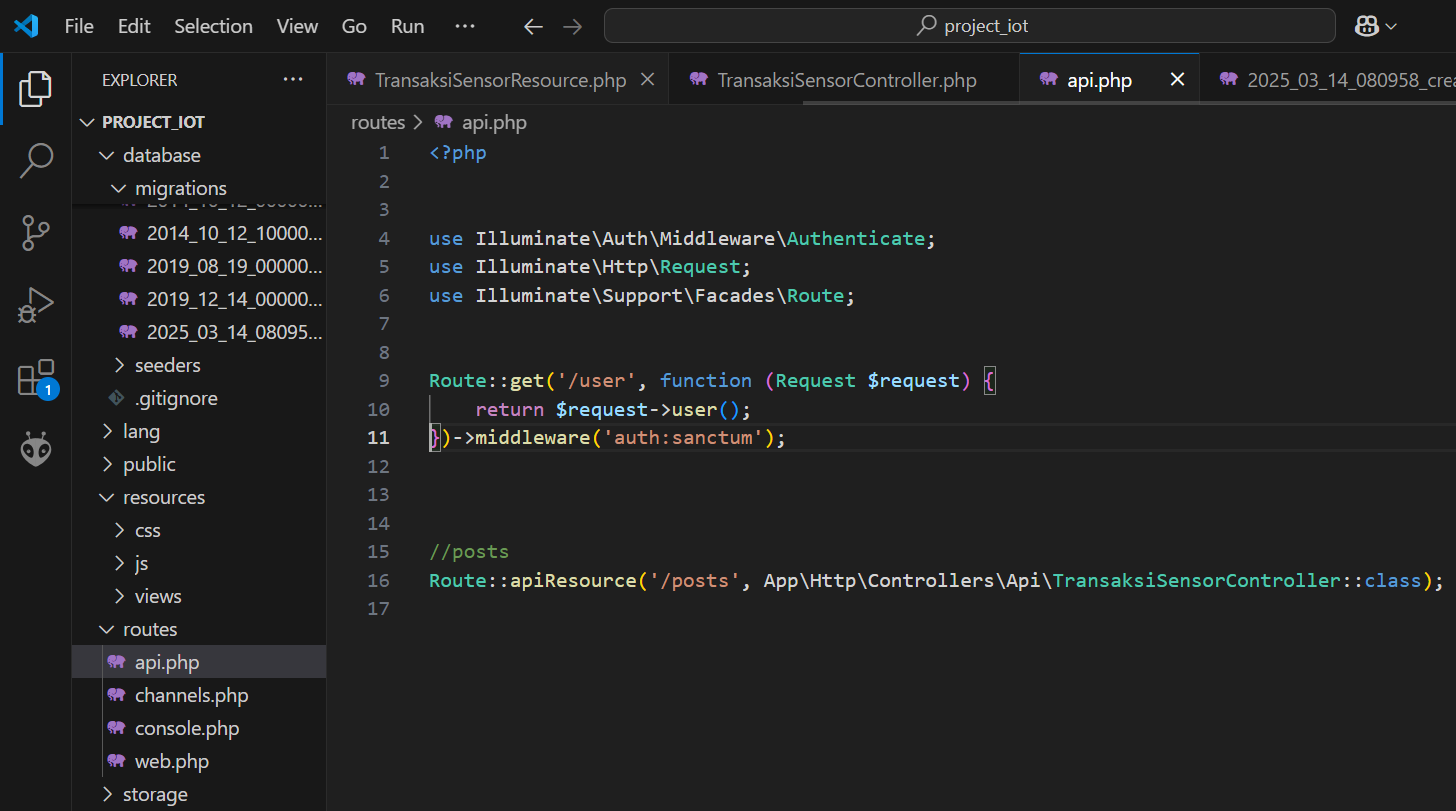
}



**Gambar 2.2.6** App//Controllers/Api/TransaksiSensorController.php.

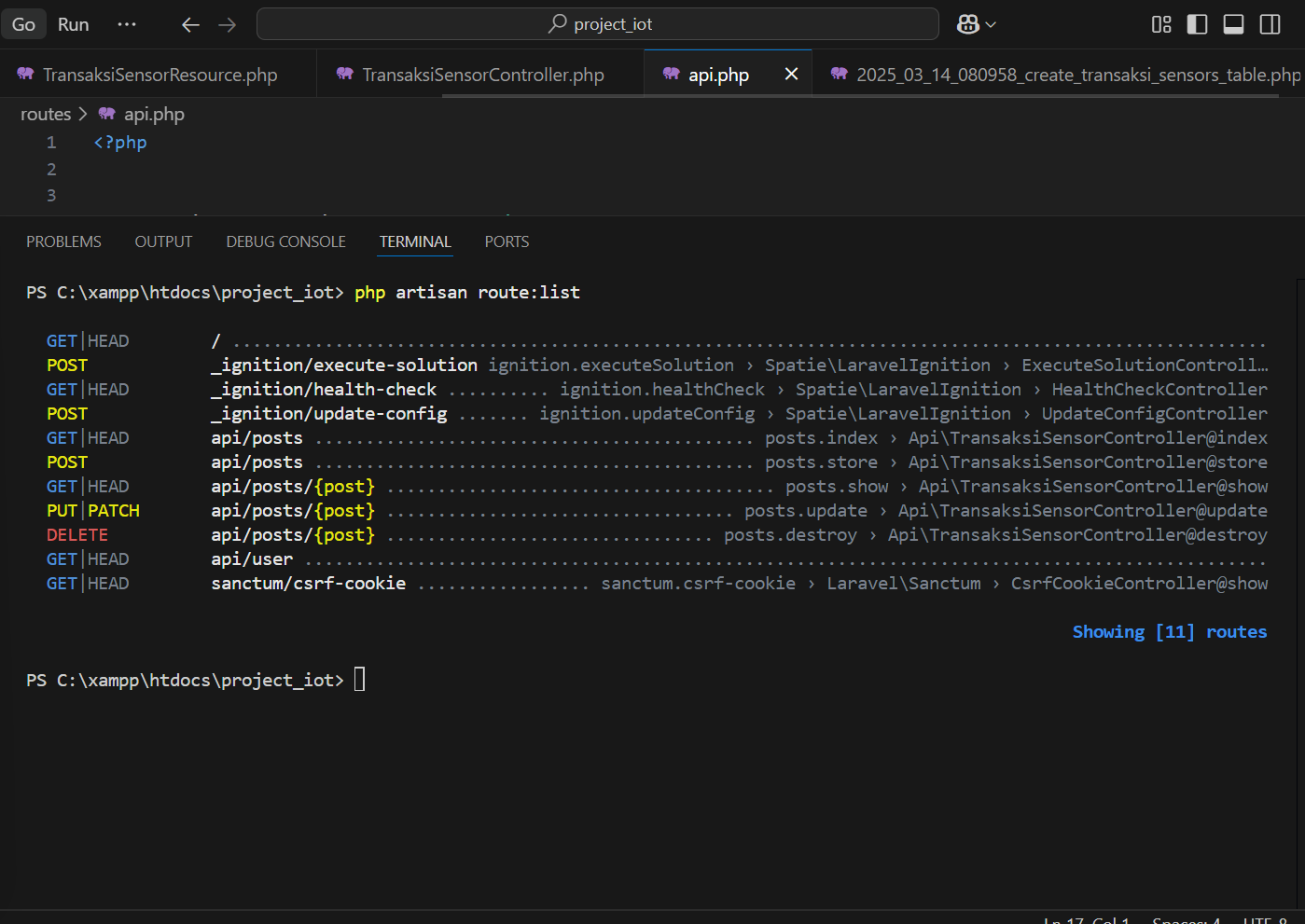
1. Membuat Route khusus API dengan perintah :

****

****

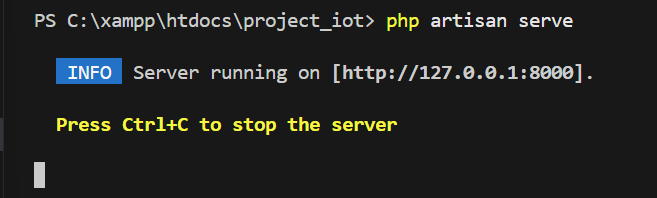
**Gambar 2.2.7** Routes/api.php.

1. Pastikan route telah terbentuk dengan menggunakan perintah seperti berikut :

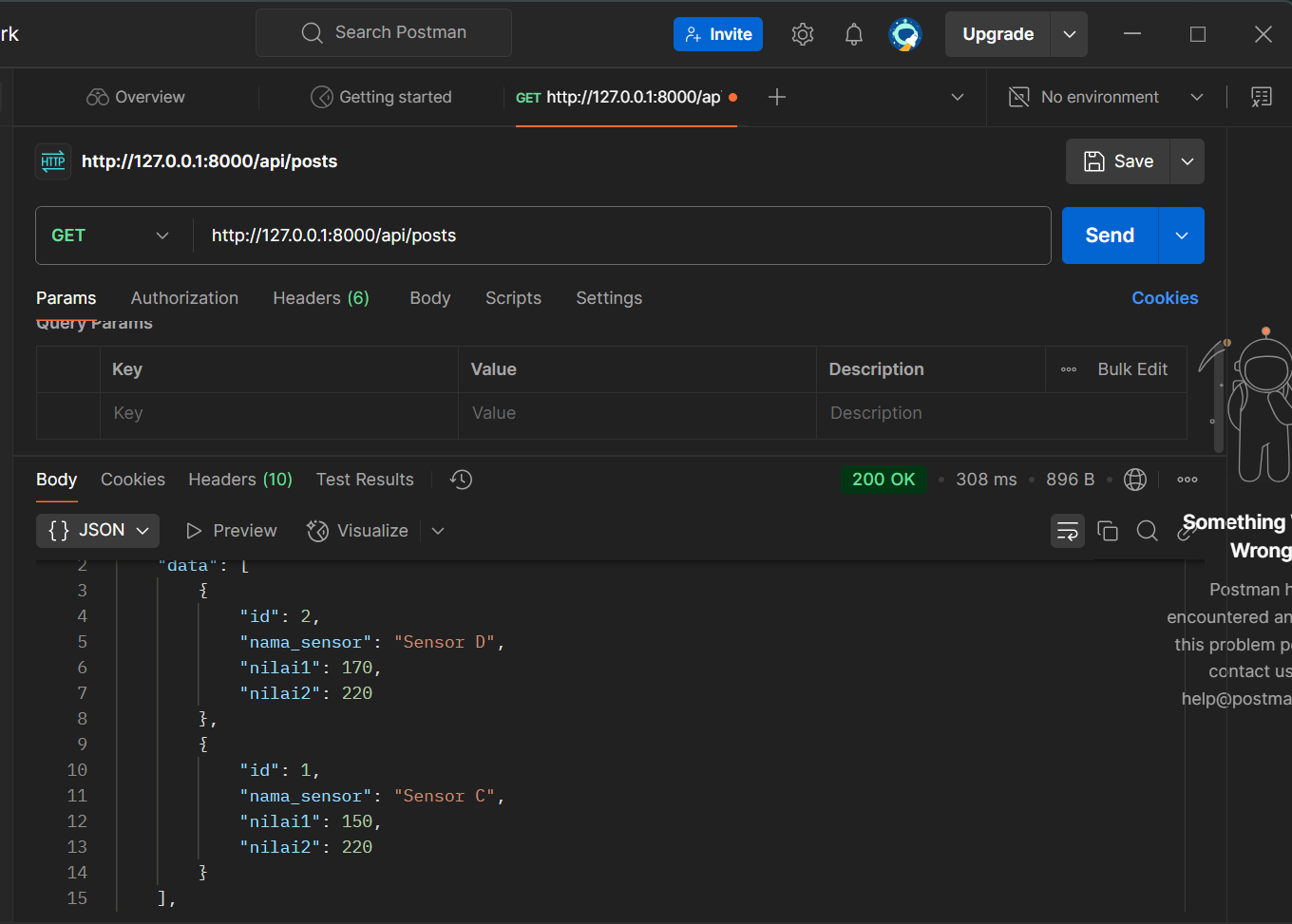


**Gambar 2.2.7** Route List.

1. Untuk menjalankan mengatur postman usahakan folder API kita berjalan dengan memasukkan perintah :



1. Pastikan telah data yang dimasukkan kedalam tabel di database.
2. Untuk mengambil data melalui aplikasi postman, jalankan prosedur berikut :

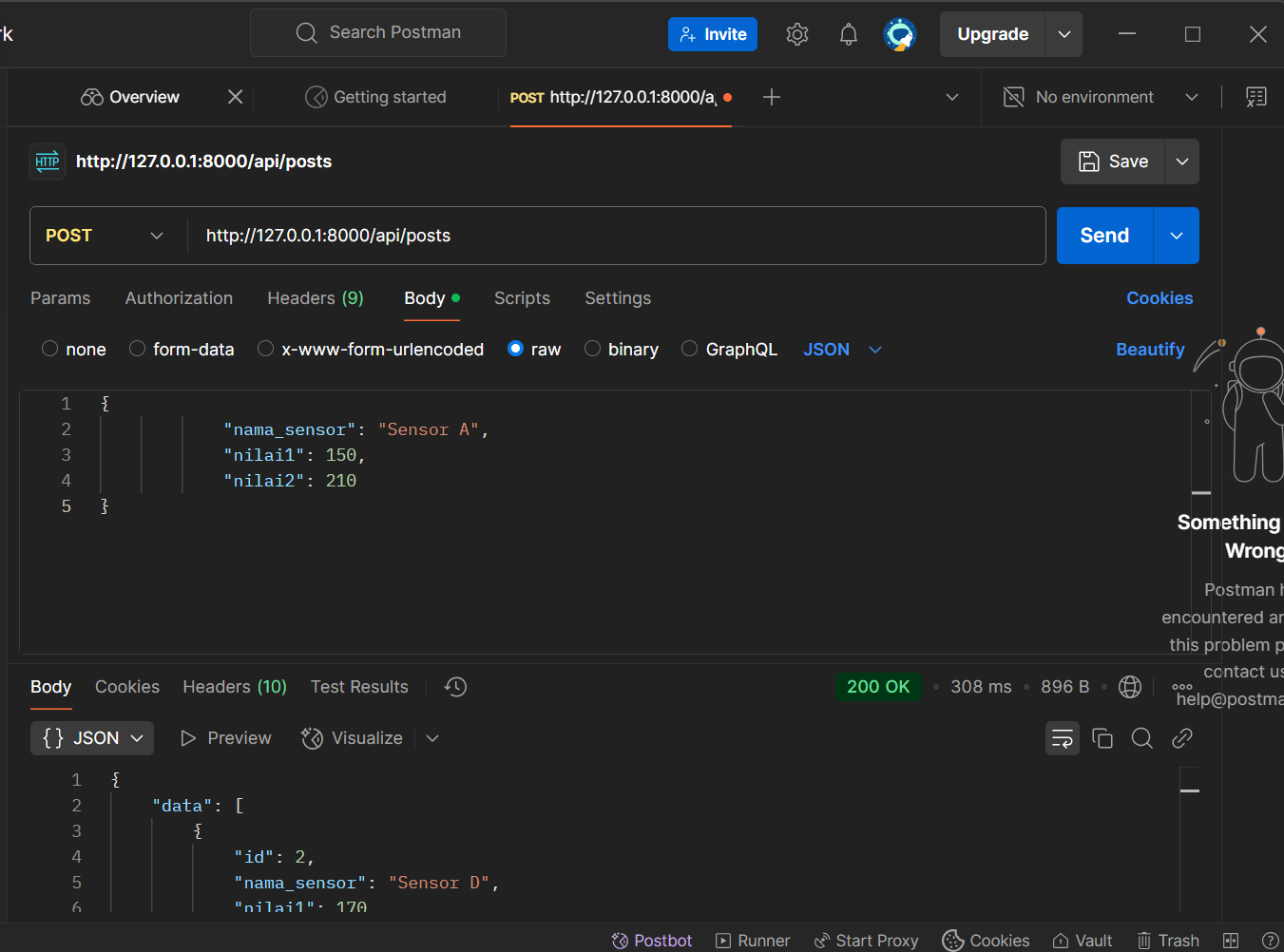
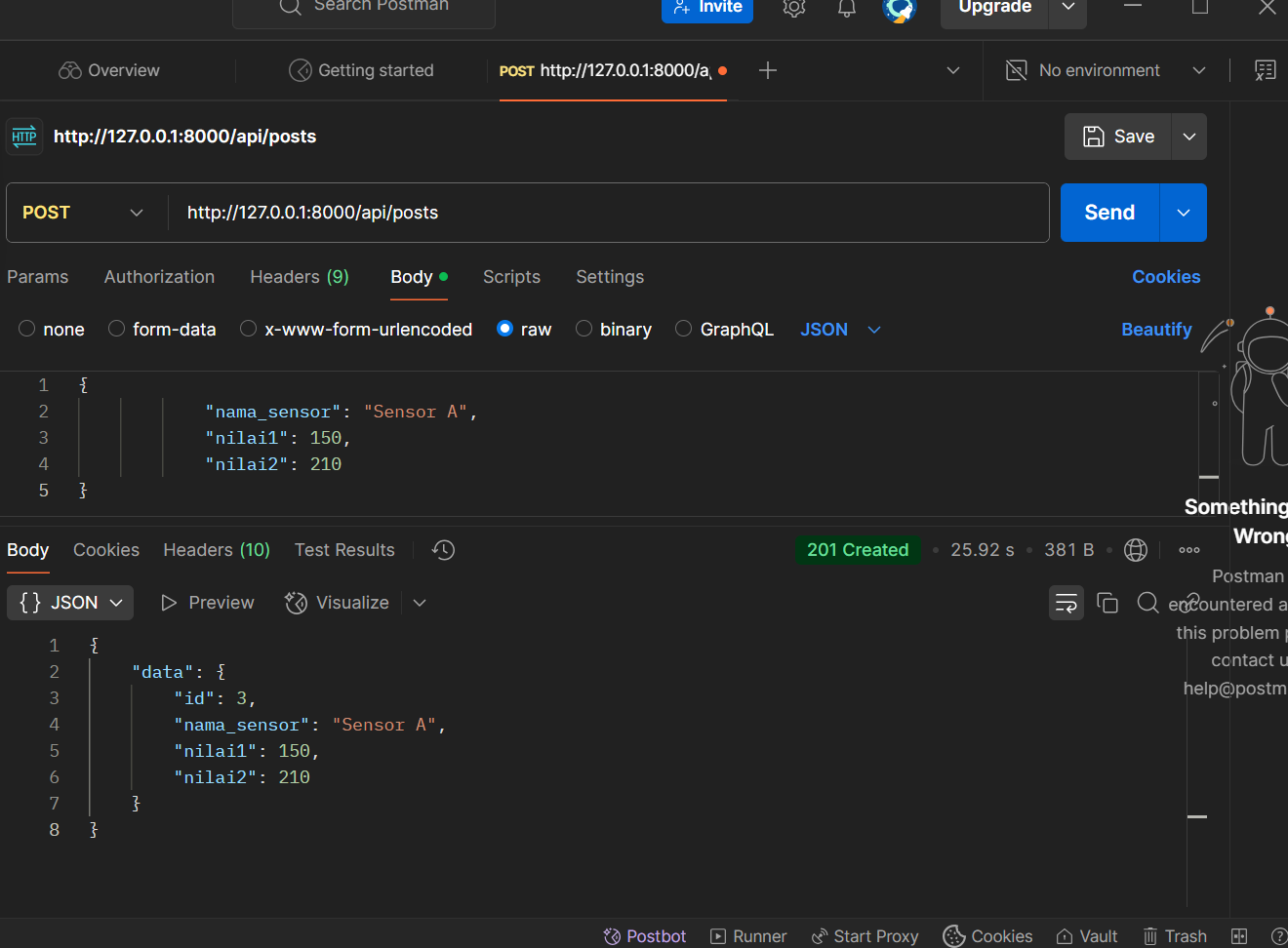


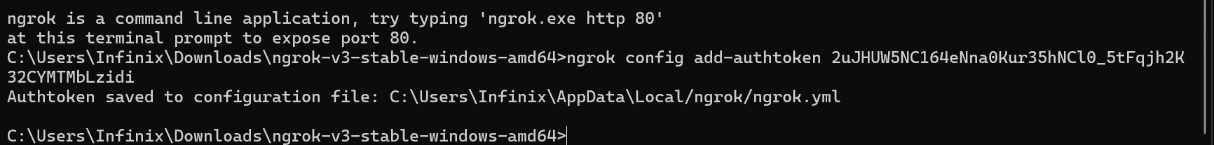
**Gambar 2.2.8** Gambar hasil simulasi.

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

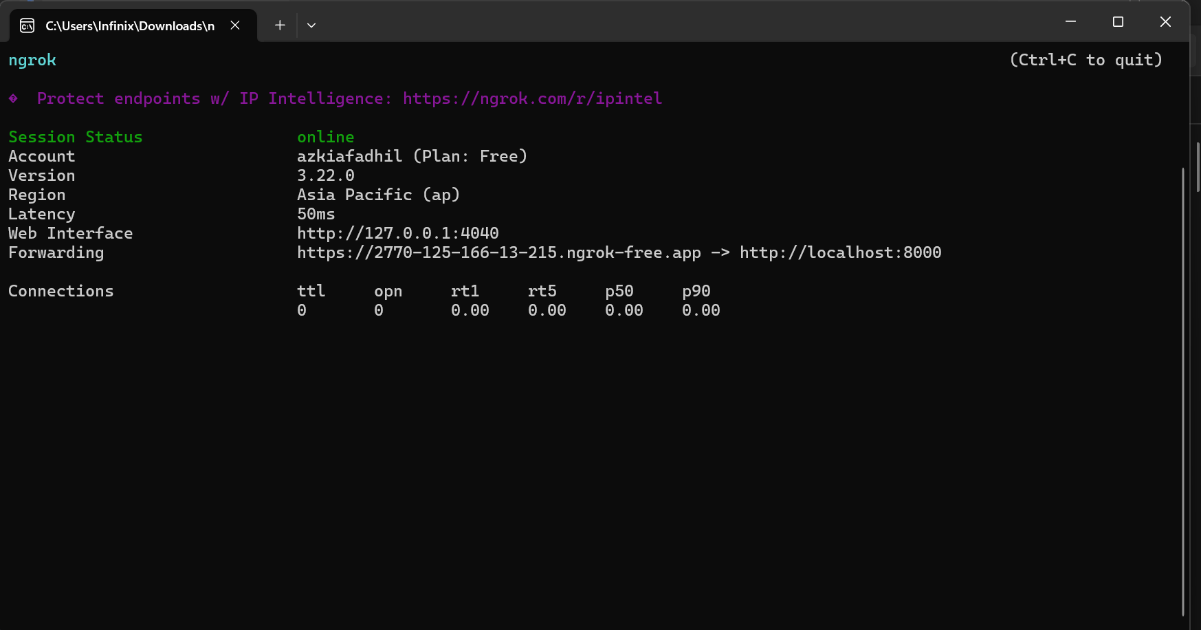
Pilih method GET untuk mengambil data dari database , kemudian klik tombol SEND

1. Input Data menggunakan postman.
2. 
3. 
4. Mengaplikasikan Ngrok setelah di Install, masukkan perintah sesuai yang ada dibawah ini:



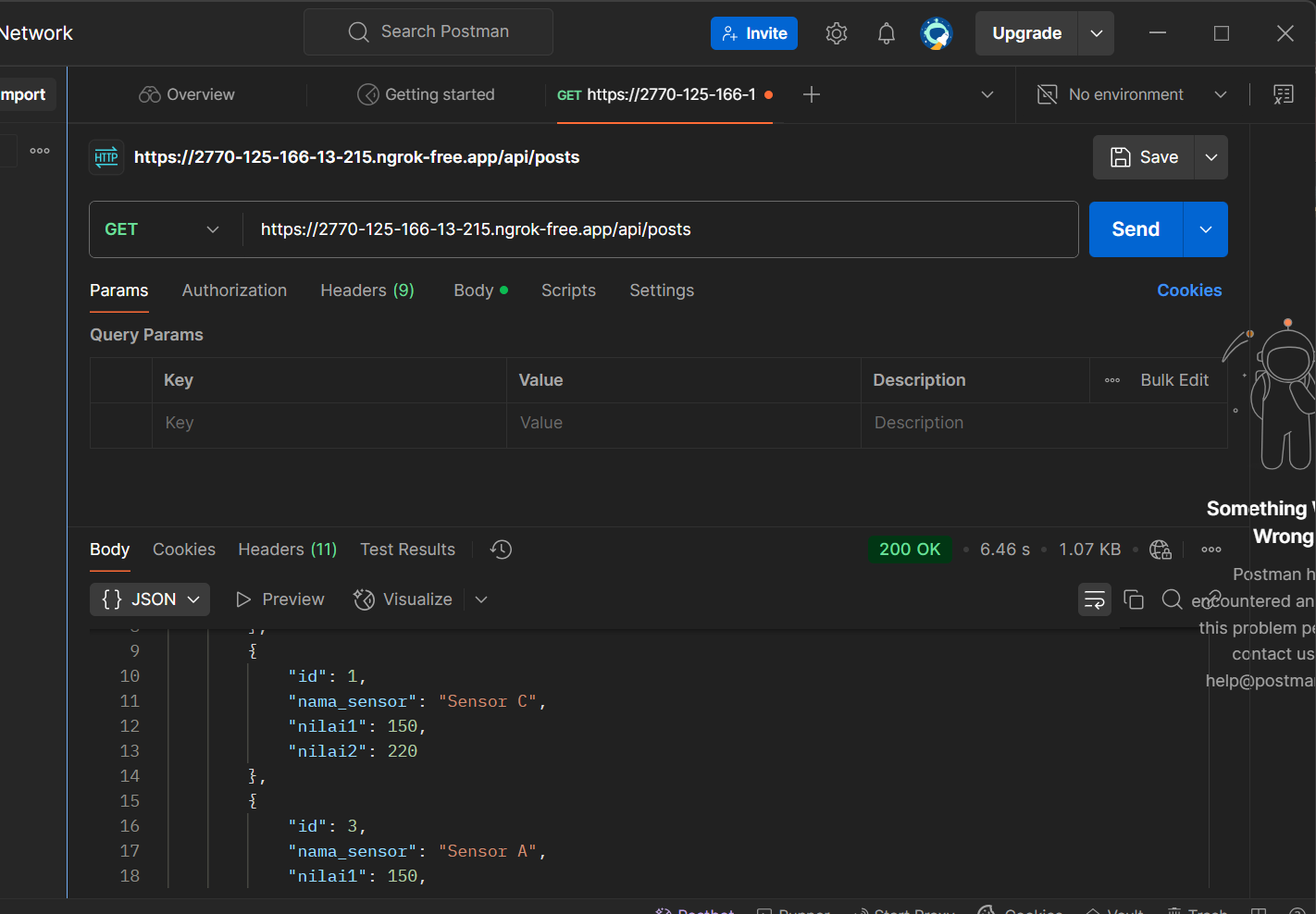
1. Kemudian jalankan perintah berikut untuk mengonline kan laravel melalui port 8000

ngrok http http://localhost:8000

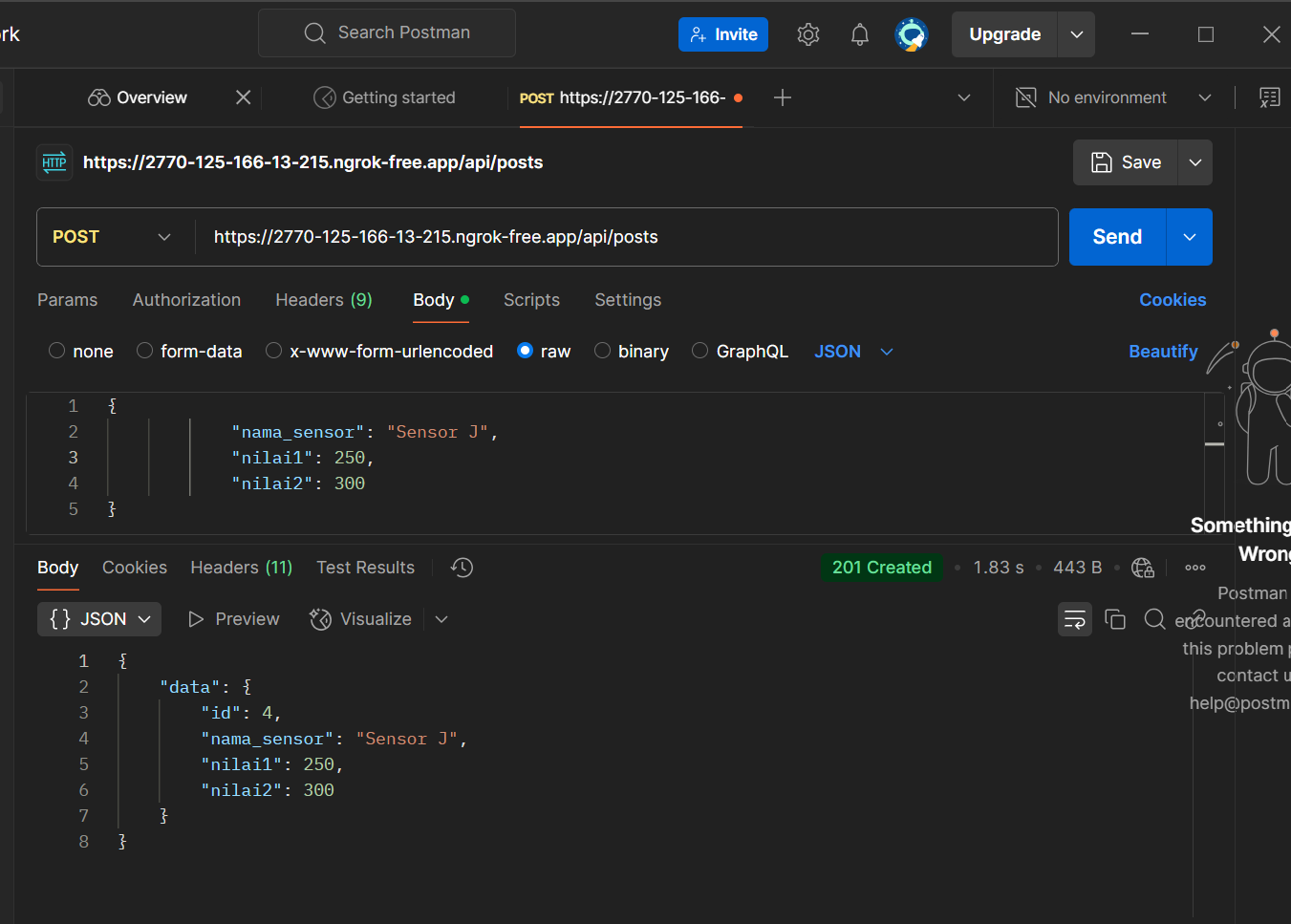


1. Lakukan percobaan menggunakan postman menggunakan URL yang diberikan oleh ngrok.

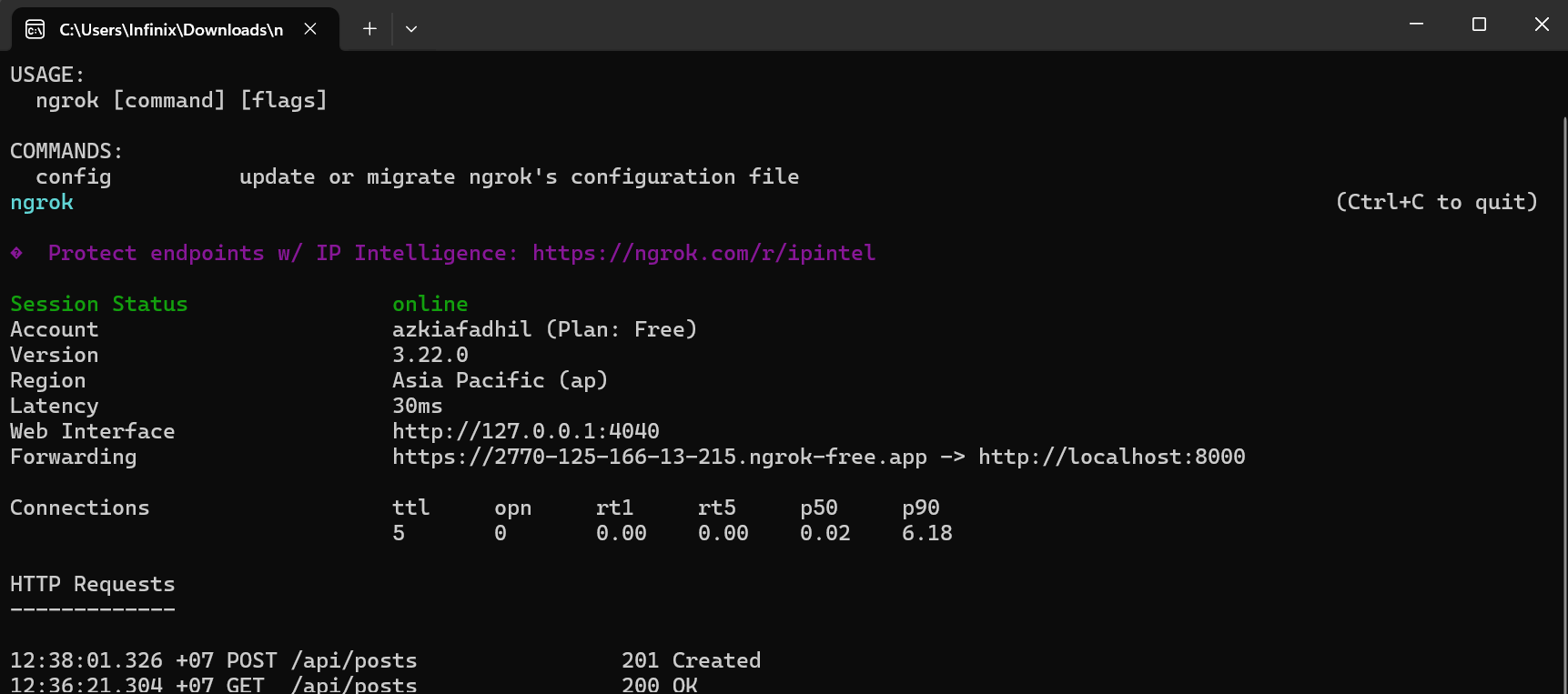




1. Selanjutnya adalah input data menggunakan Post.



1. **Hasil dan Pembahasan**
   1. **Hasil Eksperimen**

****