# LAPORAN PRAKTIKUM INTERNET OF THINGS (IoT)

# Fakultas Vokasi, Universitas Brawijaya



**Praktik Simulasi Relay, Button & LED**

*Akhmad Zaidan Hanin R*

*Fakultas Vokasi, Universitas Brawijaya   
Email : zaidanhanin12@gmail.com*

**Abstract** (Abstrak)

Praktikum ini bertujuan untuk memahami cara kerja relay, tombol (button), dan LED menggunakan mikrokontroler ESP32 dalam simulasi Wokwi. Relay digunakan sebagai saklar elektronik untuk mengendalikan perangkat, sedangkan tombol berfungsi sebagai input untuk mengontrol status LED dan relay. Simulasi dilakukan dengan menggunakan Visual Studio Code (VSCode) serta PlatformIO sebagai lingkungan pengembangan. Hasil dari simulasi menunjukkan bahwa relay dapat diaktifkan dan dinonaktifkan menggunakan tombol, serta LED dapat menyala saat relay aktif.

**1. Introduction**

* 1. **Latar belakang**

Relay merupakan komponen penting dalam sistem kendali elektronik yang memungkinkan pengendalian perangkat berdaya tinggi dengan sinyal berdaya rendah dari mikrokontroler. Pada simulasi ini, relay dikombinasikan dengan tombol dan LED untuk menunjukkan cara kerja dasar sistem kontrol berbasis ESP32.

* 1. **Tujuan eksperimen**

Memahami cara kerja relay dan integrasinya dengan ESP32.

Mempelajari penggunaan tombol sebagai input untuk mengendalikan relay.

Mengamati perubahan status LED sebagai indikator aktif atau tidaknya relay.

**2. Methodology (Metodologi)**

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

Laptop/Pc, wokwi, dan vscode

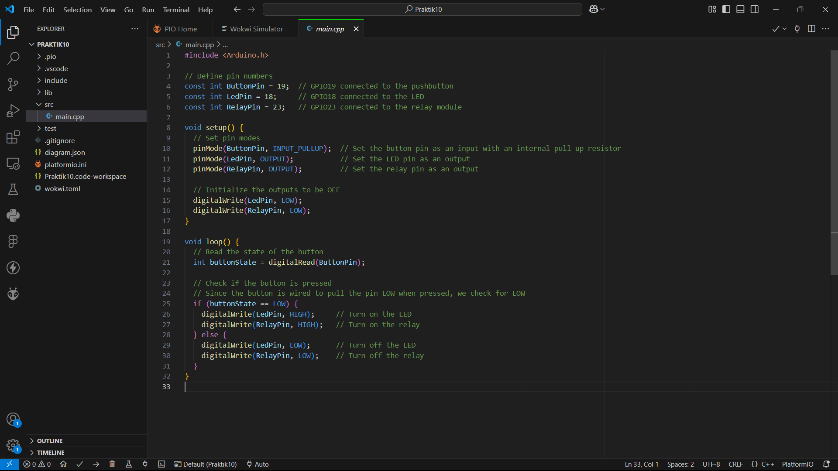
**2.2 Implementation Steps (Langkah Implementasi)**

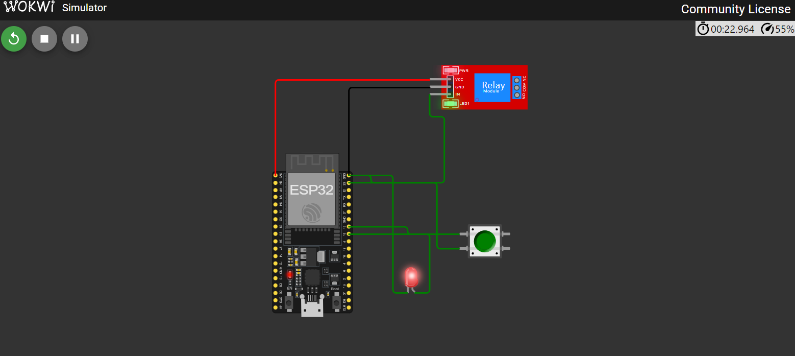
1. **Persiapan Lingkungan Pengembangan**
   * Membuat proyek baru untuk ESP32 di PlatformIO.
   * Menghubungkan ESP32 dengan modul relay, tombol, dan LED di Wokwi.
2. **Rangkaian Elektronik di Wokwi**
   * Relay Module
     + VCC → 3.3V ESP32
     + GND → GND ESP32
     + IN → GPIO23 ESP32
   * **Push Button**
     + Satu kaki ke GPIO19 ESP32
     + Satu kaki ke GND
   * **LED**
     + Katoda ke GND
     + Anoda ke GPIO18 ESP32 melalui resistor
3. **Penulisan Kode**
   * Menggunakan bahasa pemrograman C++ di dalam PlatformIO.
   * Menggunakan fungsi digitalRead() untuk membaca status tombol.
   * Menggunakan fungsi digitalWrite() untuk mengontrol LED dan relay.
   * Menggunakan pinMode() untuk mengatur mode pin ESP32.
4. **Simulasi dan Pengujian**
   * Menjalankan simulasi di Wokwi.
   * Menekan tombol untuk mengaktifkan dan menonaktifkan relay serta LED.
   * Mengamati perubahan status LED dan relay dalam simulasi.

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

**3.1 Experimental Results (Hasil Eksperimen)**

Hasil simulasi menunjukkan bahwa relay dapat dikontrol dengan baik menggunakan tombol. Saat tombol ditekan, relay aktif dan LED menyala, menandakan perangkat yang dikontrol oleh relay dalam keadaan ON. Berikut adalah tampilan simulasi:

**** Kode main.cpp



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

*https://wokwi.com/projects/424566078597603329*

# LAPORAN PRAKTIKUM INTERNET OF THINGS (IoT)

# Fakultas Vokasi, Universitas Brawijaya



**Praktik Simulasi Sensor Jarak  
(Ultrasonic)**

*Akhmad Zaidan Hanin R*

*Fakultas Vokasi, Universitas Brawijaya   
Email : zaidanhanin12@gmail.com*

**Abstract** (Abstrak)

Praktikum ini bertujuan untuk memahami cara kerja sensor ultrasonik HC-SR04 dalam mengukur jarak serta mengintegrasikannya dengan mikrokontroler ESP32. Sensor ultrasonik bekerja dengan prinsip pemantulan gelombang suara untuk menghitung jarak suatu objek. Program dikembangkan menggunakan PlatformIO di Visual Studio Code serta diuji melalui simulator Wokwi. Hasil pengujian menunjukkan bahwa sensor dapat mengukur jarak dengan akurasi yang cukup baik dan menampilkan hasilnya di Serial Monitor.

**1. Introduction**

* 1. **Latar belakang**

Sensor ultrasonik HC-SR04 sering digunakan dalam berbagai aplikasi seperti robotika, sistem parkir otomatis, dan perangkat IoT untuk mendeteksi objek serta mengukur jarak secara non-kontak. ESP32 sebagai mikrokontroler dengan fitur komunikasi nirkabel dapat memproses data dari sensor ini dengan cepat dan efisien.

* 1. **Tujuan eksperimen**

1. Memahami prinsip kerja sensor ultrasonik HC-SR04 dalam mengukur jarak.
2. Mengintegrasikan sensor HC-SR04 dengan ESP32.
3. Melakukan simulasi pengukuran jarak menggunakan Wokwi dan menampilkan hasil di Serial Monitor.

**2. Methodology (Metodologi)**

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

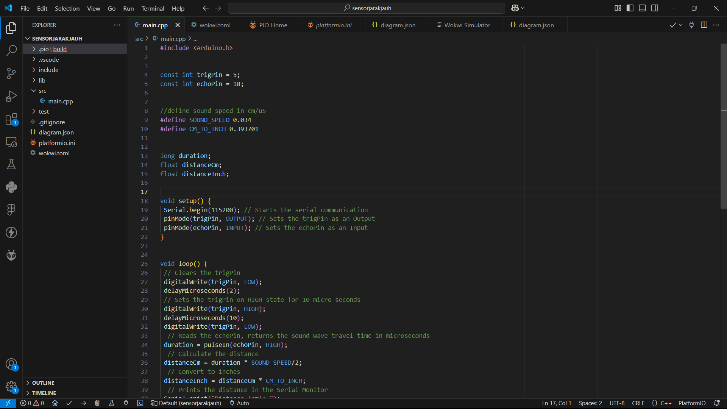
Laptop/Pc, wokwi, dan vscode

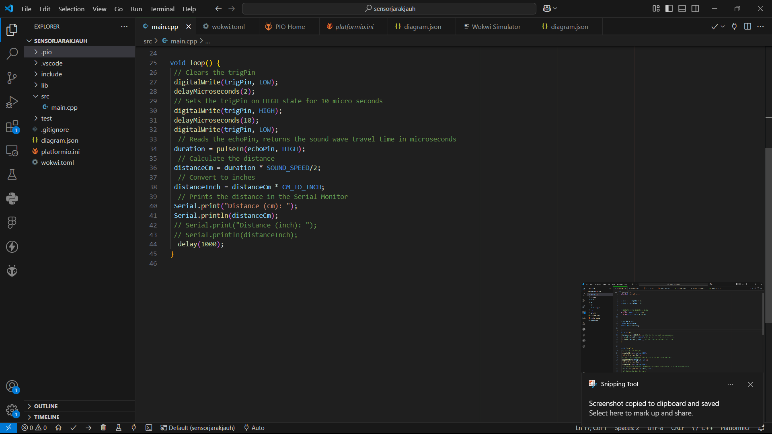
**2.2 Implementation Steps (Langkah Implementasi)**

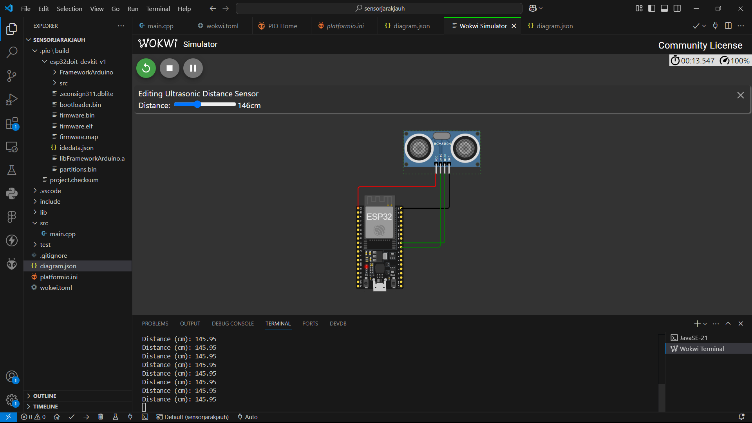
1. **Persiapan Lingkungan Pengembangan**
   * Membuat proyek baru untuk ESP32 di PlatformIO.
   * Menyusun rangkaian ESP32 dengan sensor ultrasonik di Wokwi.
2. **Konfigurasi Rangkaian Elektronik di Wokwi**
   * Sensor HC-SR04
     + VCC → 3.3V ESP32
     + GND → GND ESP32
     + Trig → GPIO 5 ESP32
     + Echo → GPIO 18 ESP32
3. **Penulisan Kode Program**
   * Menggunakan fungsi pulseIn() untuk membaca durasi pantulan gelombang ultrasonik.
   * Menghitung jarak menggunakan rumus:
   * Menampilkan hasil pengukuran pada Serial Monitor**.**
4. **Simulasi dan Pengujian**
   * Menjalankan simulasi di Wokwi.
   * Memantau perubahan jarak pada Serial Monitor.
   * Menyesuaikan nilai jarak dalam simulator untuk melihat perubahan output.

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

**3.1 Experimental Results (Hasil Eksperimen)**

 Kode main.cpp





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

*https://wokwi.com/projects/424566680229757953*

# LAPORAN PRAKTIKUM INTERNET OF THINGS (IoT)

# Fakultas Vokasi, Universitas Brawijaya



**Praktik Pembuatan API Menggunakan Laravel 11 dan Ngrok**

*Akhmad Zaidan Hanin R*

*Fakultas Vokasi, Universitas Brawijaya   
Email : zaidanhanin12@gmail.com*

**Abstract** (Abstrak)

Pengembangan API menjadi kebutuhan utama dalam pengembangan aplikasi berbasis web dan mobile. Laravel 11, sebagai framework PHP terbaru, menyediakan fitur yang memudahkan dalam pembuatan API. Praktikum ini bertujuan untuk mempelajari proses instalasi dan konfigurasi Laravel 11 menggunakan XAMPP, pembuatan API dengan fitur CRUD, serta penggunaan Ngrok untuk memungkinkan akses API secara publik. Hasil praktik menunjukkan bahwa Laravel 11 mempermudah pengembangan API dengan fitur routing dan migration yang terintegrasi. Selain itu, Ngrok memungkinkan pengujian API tanpa harus melakukan deployment ke server. Dengan demikian, kombinasi Laravel 11 dan Ngrok menjadi solusi efisien dalam pengembangan dan pengujian API secara lokal sebelum dipublikasikan secara luas.

**1. Introduction**

* 1. **Latar belakang**

Laravel adalah salah satu framework PHP yang populer dan digunakan untuk membangun aplikasi berbasis web, termasuk API. Laravel 11 hadir dengan berbagai fitur baru yang mempermudah pengembangan API. Untuk mengakses API secara publik selama pengembangan, kita dapat menggunakan Ngrok, yang memungkinkan tunneling ke localhost.

* 1. **Tujuan eksperimen**

1. Membangun API menggunakan Laravel 11 untuk mengelola data sensor.
2. Mengimplementasikan migrasi database, model, resource, dan controller sesuai standar Laravel.
3. Melakukan pengujian API menggunakan Postman untuk operasi CRUD (Create, Read, Update, Delete).
4. Meng-online-kan API menggunakan Ngrok agar dapat diakses secara publik melalui internet.

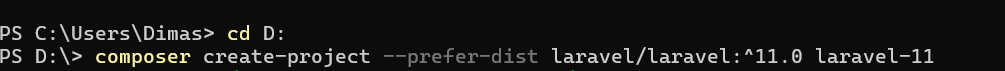
**2. Methodology (Metodologi)**

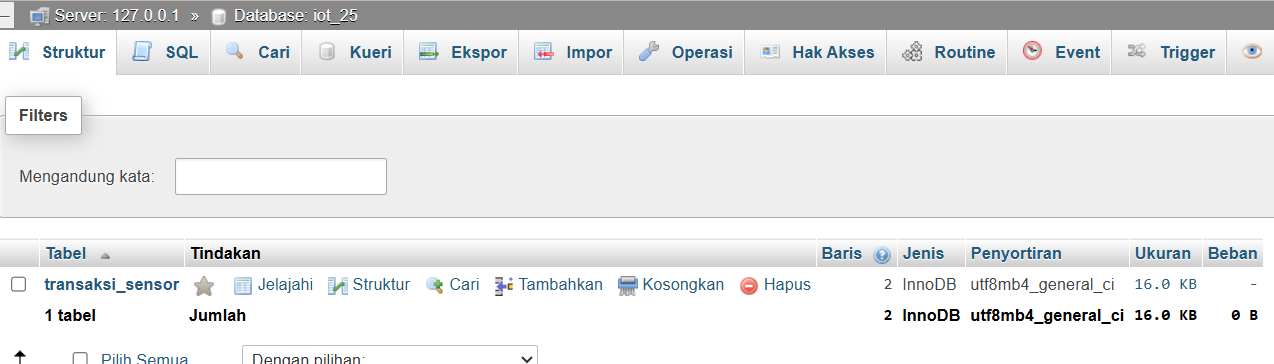
**2.1 Tools & Materials (Alat dan Bahan)**

Laptop/Pc, vscode, mysql, postman,dan ngrok

**2.2 Implementation Steps (Langkah Implementasi)**

1. Buka terminal buat memulai laravel 11

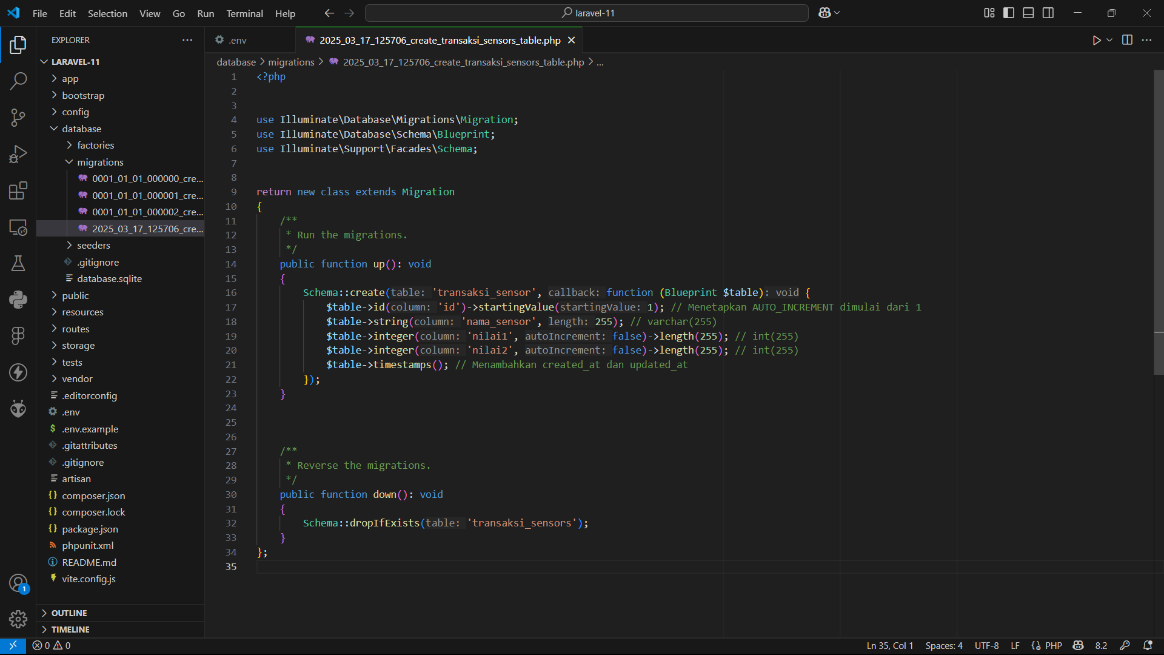


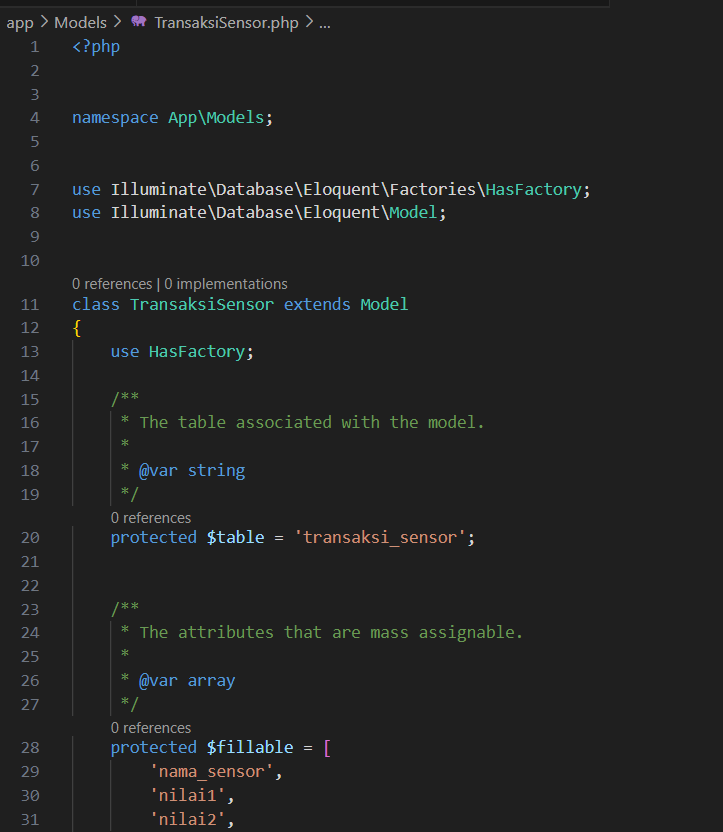
1. Buatlah database MySql
2. Buat file model TransaksiSensor.php dengan cara menjalankan perintah di terminal Vscode

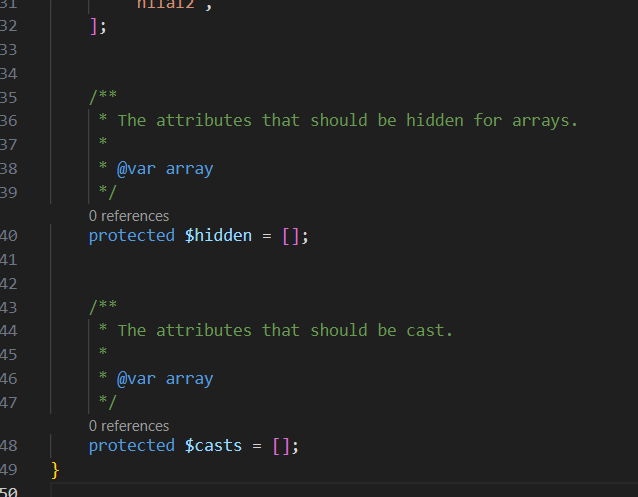
### php artisan make:model TransaksiSensor -m

### 

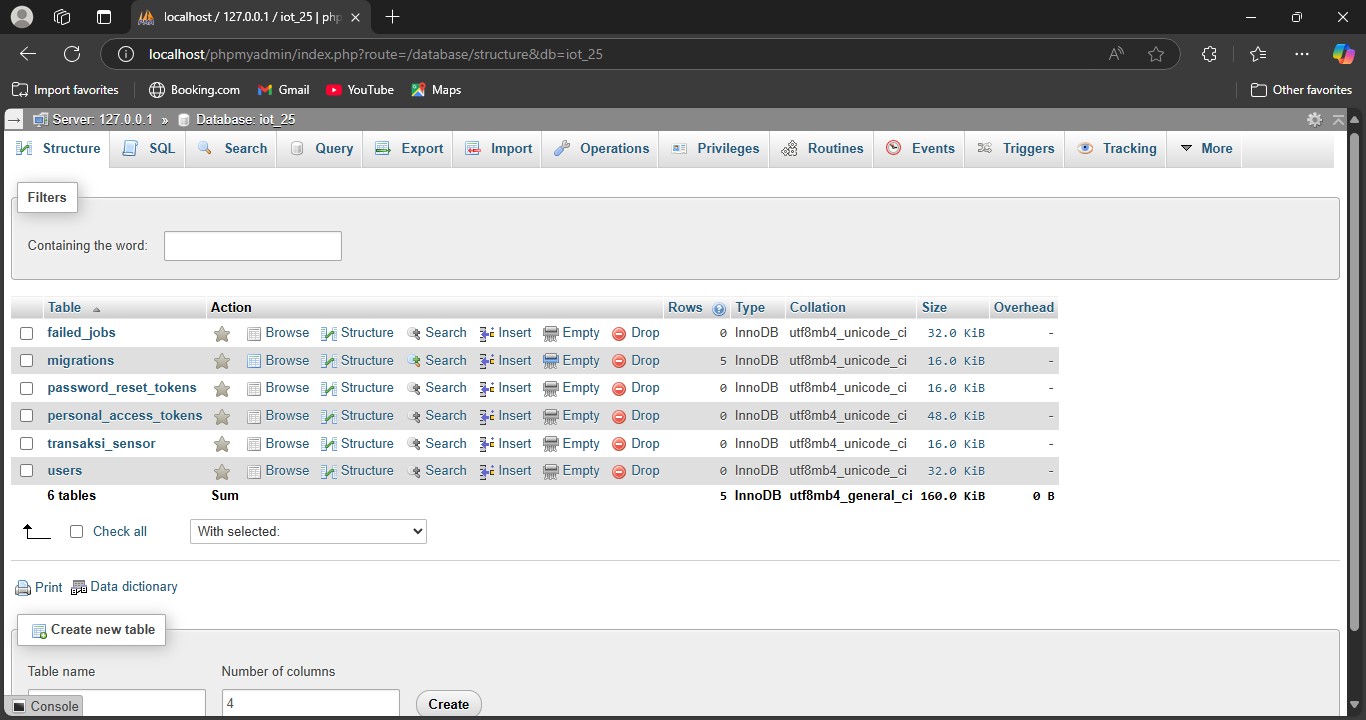
1. Kemudian melakukan pengeditan pada file **2025\_03\_10\_012536\_create\_transaksi\_sensors\_table.php**

****

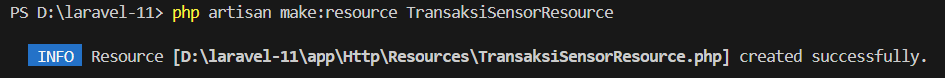
1. Setelah itu melakun pengeditan juga pada file **app/Models/TransaksiSensor.php**

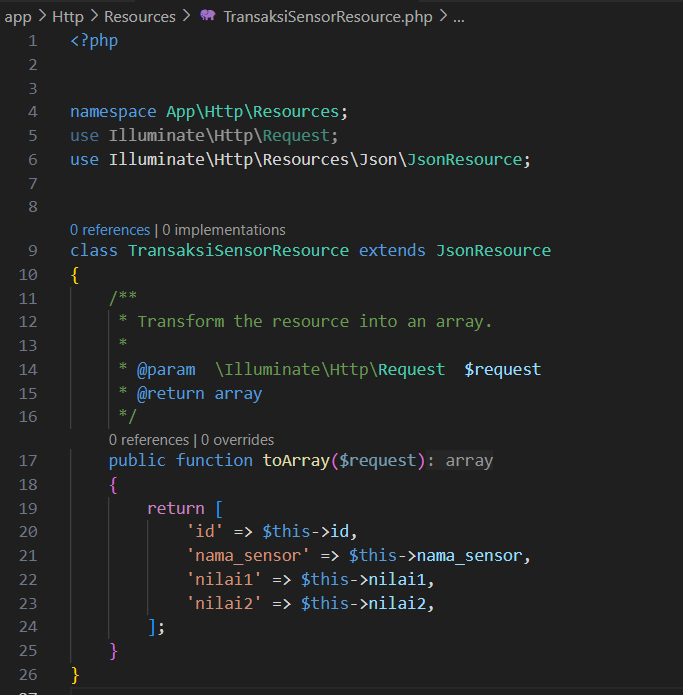


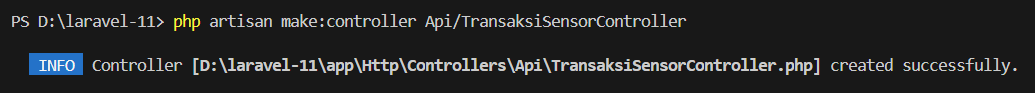
1. Kemudian lakukan perintah **php artisan migrate** diterminal dan maka pada database akan terinput terdapat table – table data



1. Kemudian buatlah resource dengan scema php artisan make:resource TransaksiSensorResource dan edit file



1. Buatlah API controller dengan menjalankan scema **php artisan make:controller Api/TransaksiSensorController** dan melakukan pengeditan pada file **app/Http/Controllers/Api/TransaksiSensorController.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

{

    /\*\*

     \* 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);

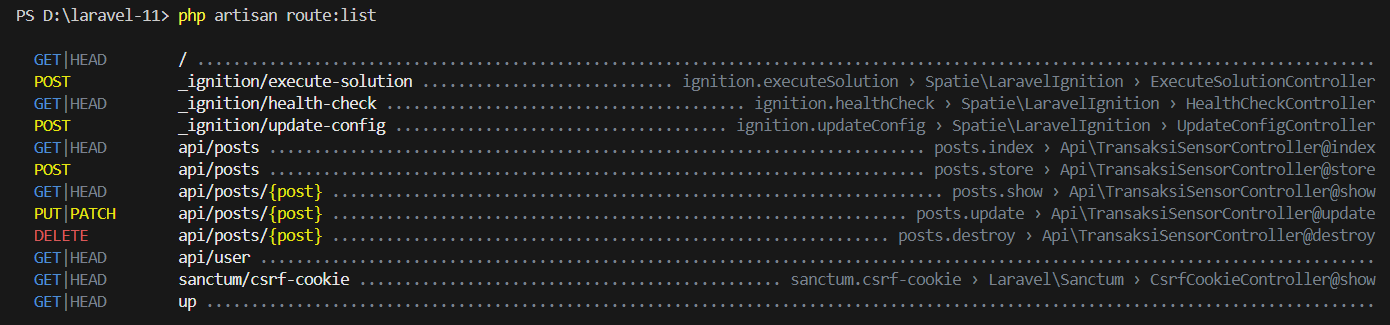
}

}

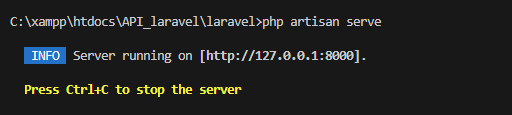
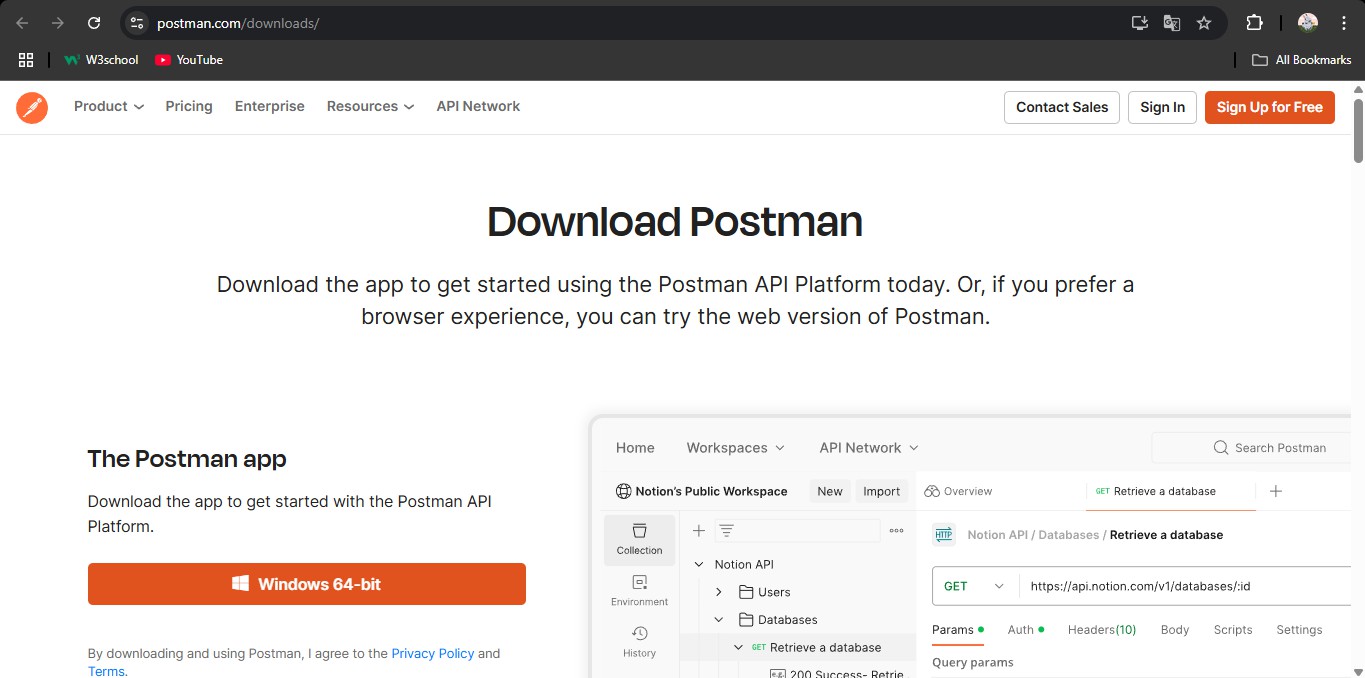
1. Buat route khusus API dengan menjalankan scema **php artisan make:api** dan melakukan pengeditan file



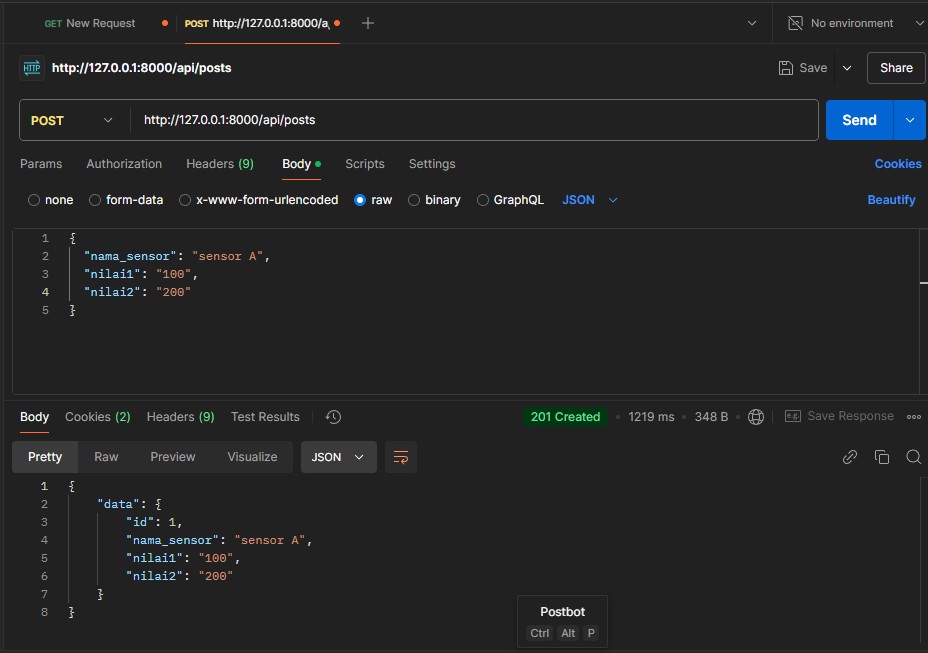
1. Kemudian pastikan routes telah terbentuk dengan menjalankan scema **php artisan route:list**

****

1. Download POSTMAN dan lakukan scema **php artisan serve** pada terminal



1. Kemudian lakukan pengambilan data melalui postman dengan melakukan scema berikut;

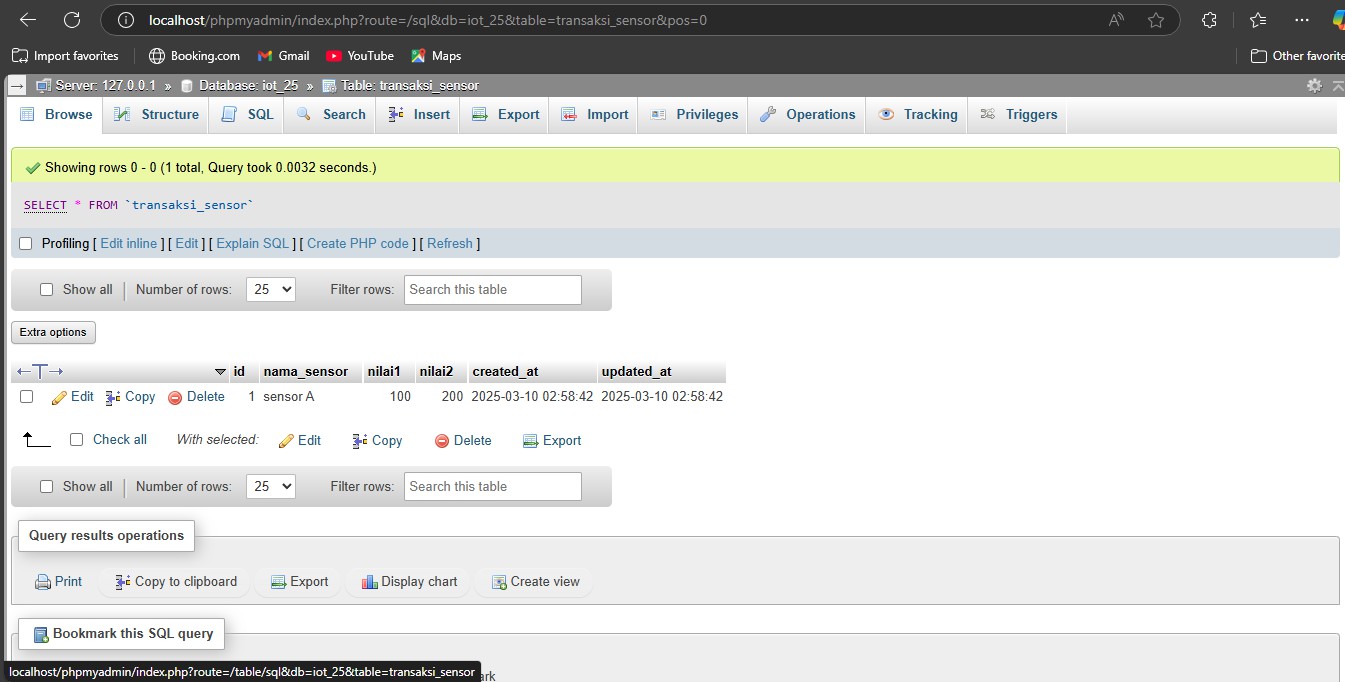
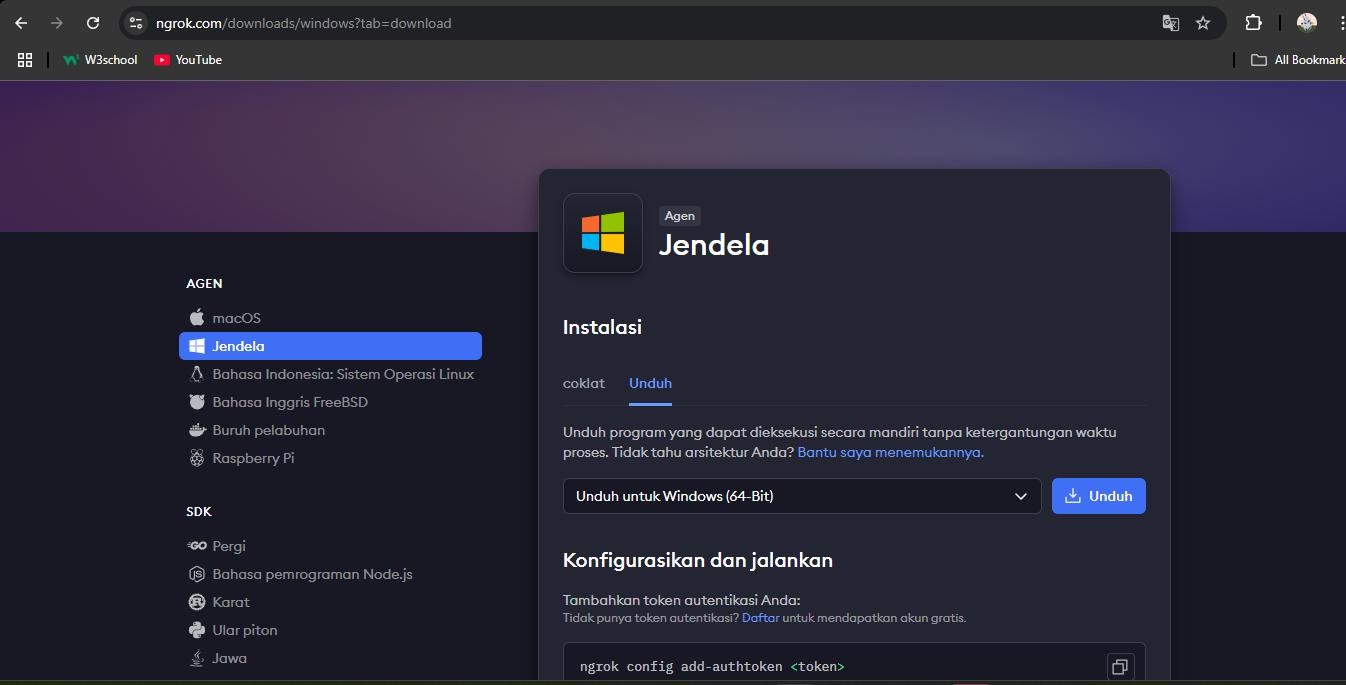


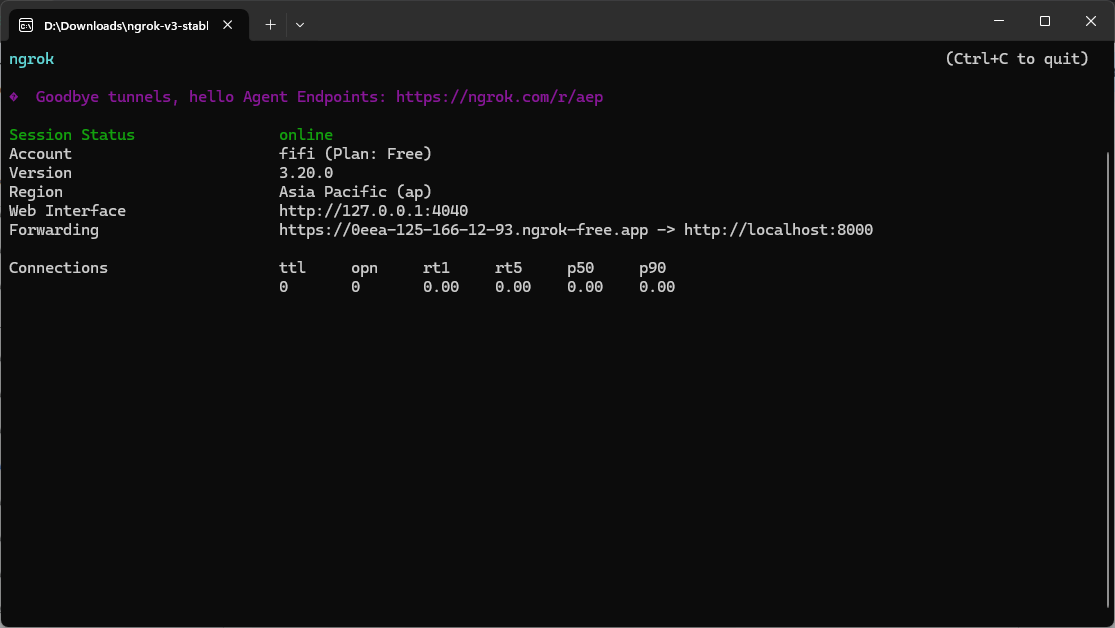
{

"nama\_sensor": "sensor A", "nilai1": "100",

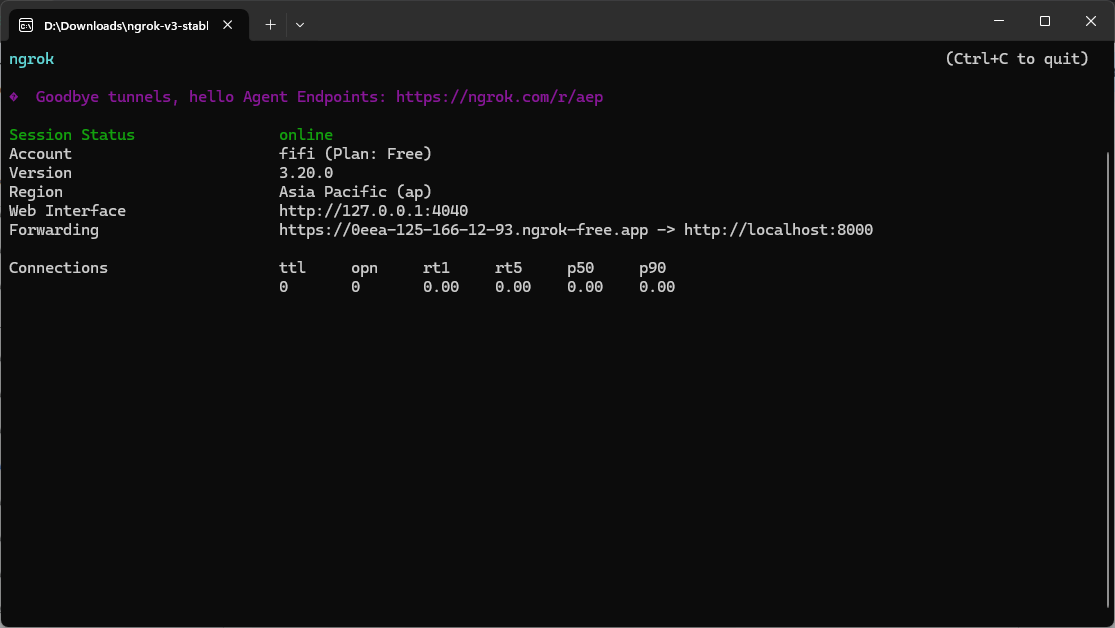
"nilai2": "200"

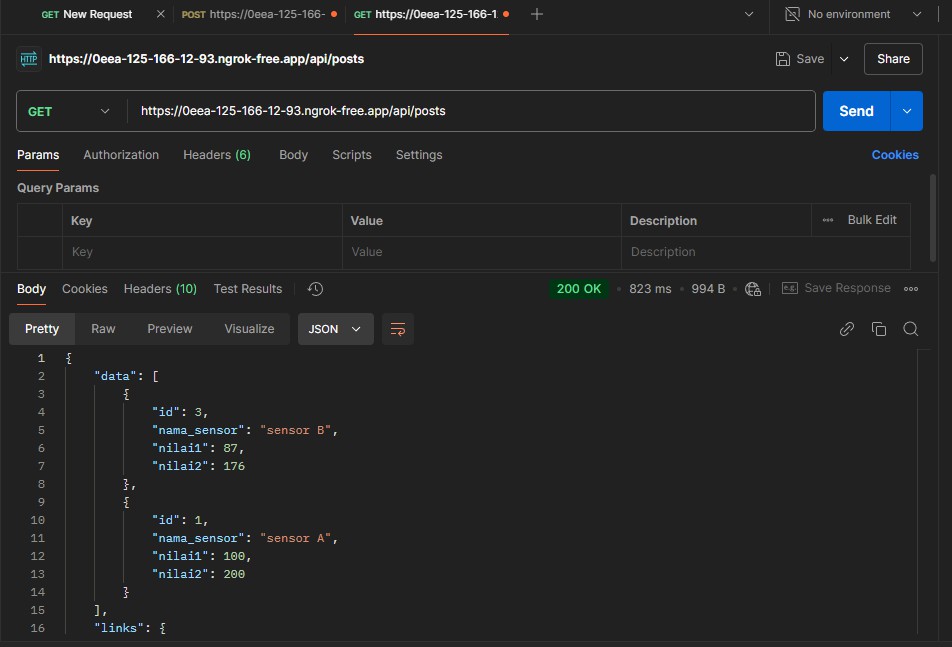
}

1. Setelah itu lakukan pengecekan database pada localhost
2. Langkah berikutnya adalah mengonline-kan API menggunakan service ngrok sehingga API dapat diakses melalui device iot atau simulasi wokwi iot
   1. Download aplikasi ngrok
   2. Kemudian lakukan konfigurasi pada file ngrok yang sudah di ekstrak
   3. Kemudian jalankan perintah berikut untuk mengonline kan laravel melalui port 8000 dengan scema berikut:

**ngrok http** [**http://localhost:8000**](http://localhost:8080/)

* 1. Dengan link yang diberikan oleh ngrok kita dapat melakukan percobaan munggunakan postman denan menanbahkan **/api/posts**



****