# LAPORAN PRAKTIKUM INTERNET OF THINGS (IoT)

# Fakultas Vokasi, Universitas Brawijaya

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

**Author(s)** Muhammad Hafizh Al Furqon

*Fakultas Vokasi, Universitas Brawijaya*

*Email:* [*muhammad.hafizhalfurqon@gmail.com*](mailto:muhammad.hafizhalfurqon@gmail.com)

Perkembangan teknologi web semakin mendorong kebutuhan akan API (Application Programming Interface) yang efisien dan mudah diakses. Laravel, sebagai salah satu framework PHP yang populer, menyediakan fitur bawaan untuk pengembangan API yang cepat dan aman. Namun, dalam tahap pengujian dan pengembangan, keterbatasan akses ke server lokal dapat menjadi kendala. Ngrok hadir sebagai solusi dengan memungkinkan server lokal diekspos ke internet melalui tunneling. Laporan ini membahas langkah-langkah dalam pembuatan API menggunakan Laravel 12 serta integrasinya dengan Ngrok untuk pengujian akses secara eksternal. Proses dimulai dengan instalasi Laravel, konfigurasi database, pembuatan endpoint API menggunakan Laravel Sanctum untuk autentikasi, hingga pengujian menggunakan Postman dan akses publik melalui Ngrok. Hasil praktik ini menunjukkan bahwa kombinasi Laravel dan Ngrok dapat mempercepat proses pengembangan API serta memudahkan kolaborasi dalam tim.

**Keywords**—*API, Laravel 12, Ngrok, IoT, Tunneling*

**1. Introduction**

**1.1 Latar belakang**

Dalam era digital saat ini, API (Application Programming Interface) menjadi komponen penting dalam pengembangan perangkat lunak, termasuk dalam sistem berbasis IoT. Laravel, sebagai framework PHP modern, menyediakan berbagai fitur yang memudahkan pembuatan API yang efisien dan aman. Ngrok digunakan untuk membuat API Laravel dapat diakses secara publik tanpa perlu hosting langsung.

**1.2 Tujuan eksperimen**

1. Memahami dasar pembuatan API menggunakan Laravel 12.
2. Menerapkan migrasi database dengan Eloquent ORM.
3. Menggunakan Ngrok untuk mengakses API secara publik.
4. Menguji endpoint API dengan Postman atau alat lain.

**2. Methodology (Metodologi)**

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

* **Perangkat Lunak:**
  + Laravel 12
  + PHP 8+
  + Composer
  + MySQL & phpMyAdmin
  + Ngrok
  + Postman
* **Perangkat Keras:**
  + Laptop/PC dengan koneksi internet

**2.2 Implementation Steps (Langkah Implementasi)**

1. Persiapan Lingkungan

Install Laravel 12 menggunakan Composer:

Laravel new iot\_25

Masuk ke direktori proyek:

cd laravel\_new

Jalankan server lokal Laravel:

php artisan serve

1. Konfigurasi Database

Buat database baru di phpMyAdmin dengan nama iot\_25.

Ubah konfigurasi .env untuk koneksi database:  
 DB\_CONNECTION=mysql

DB\_HOST=127.0.0.1

DB\_PORT=3306

DB\_DATABASE=iot\_25

DB\_USERNAME=root

DB\_PASSWORD=

1. Pembuatan Model dan Migrasi Database

Buat model TransaksiSensor beserta migrasi:

php artisan make:model TransaksiSensor -m

Edit file migrasi

Schema::create('transaksi\_sensor', function (Blueprint $table) {

$table->id('id')->startingValue(1);

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

$table->integer('nilai1');

$table->integer('nilai2');

$table->timestamps();

});

Jalankan migrasi

php artisan migrate

1. Pembuatan Model Eloquent

Edit file app/Models/TransaksiSensor.php:

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class TransaksiSensor extends Model

{

use HasFactory;

protected $table = 'transaksi\_sensor';

protected $fillable = ['nama\_sensor', 'nilai1', 'nilai2'];

}

1. Pembuatan Controller API

Buat controller API:  
php artisan make:controller TransaksiSensorController --api

Tambahkan metode CRUD di app/Http/Controllers/TransaksiSensorController.php

1. Pembuatan Routing API

Edit file routes/api.php untuk menambahkan endpoint:

//posts

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

1. Menjalankan API dengan Ngrok

Jalankan Laravel

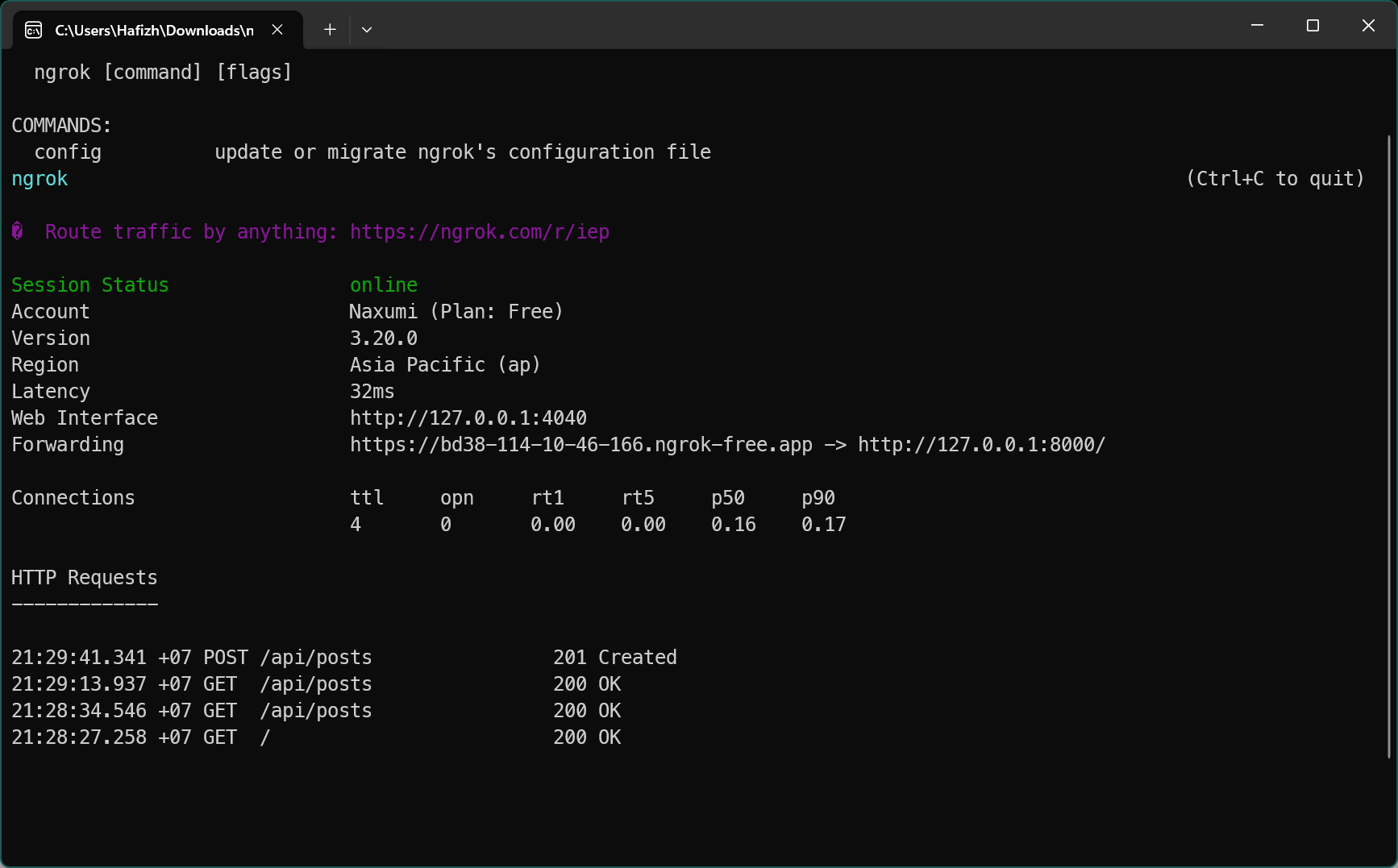
php artisan serve

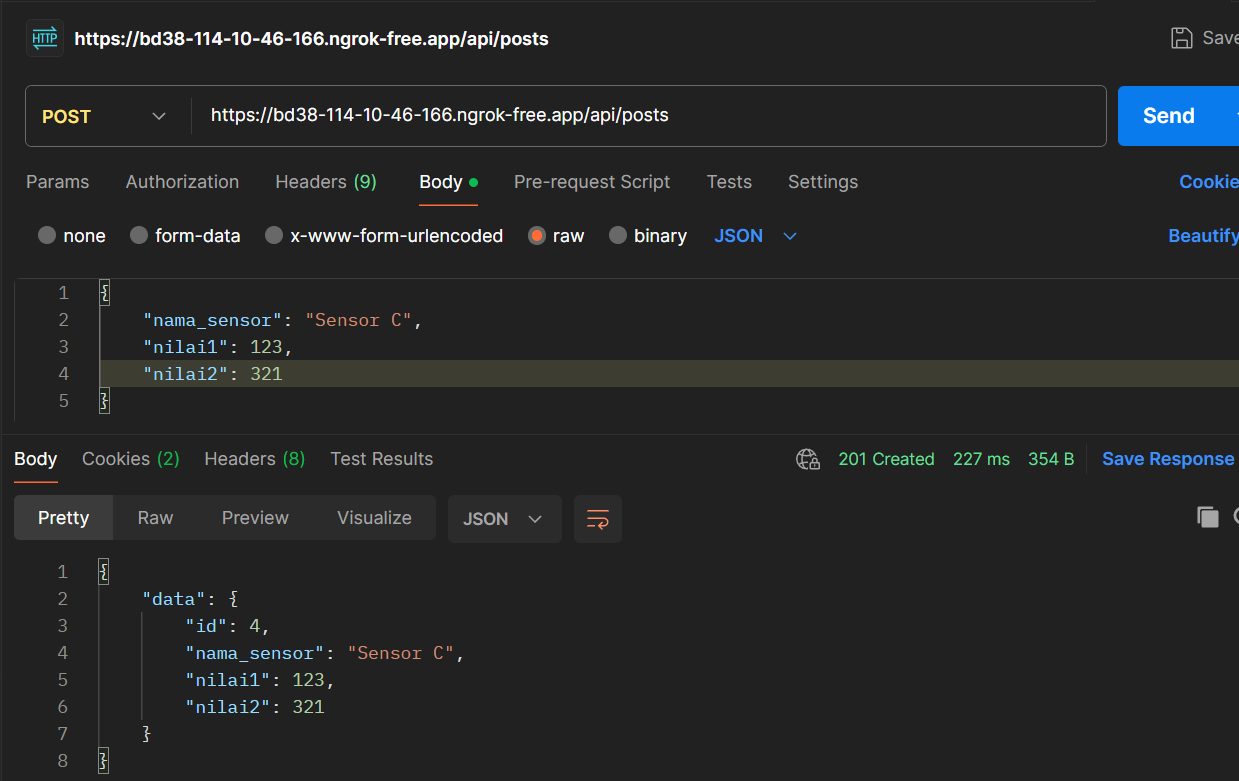
Jalankan Ngrok

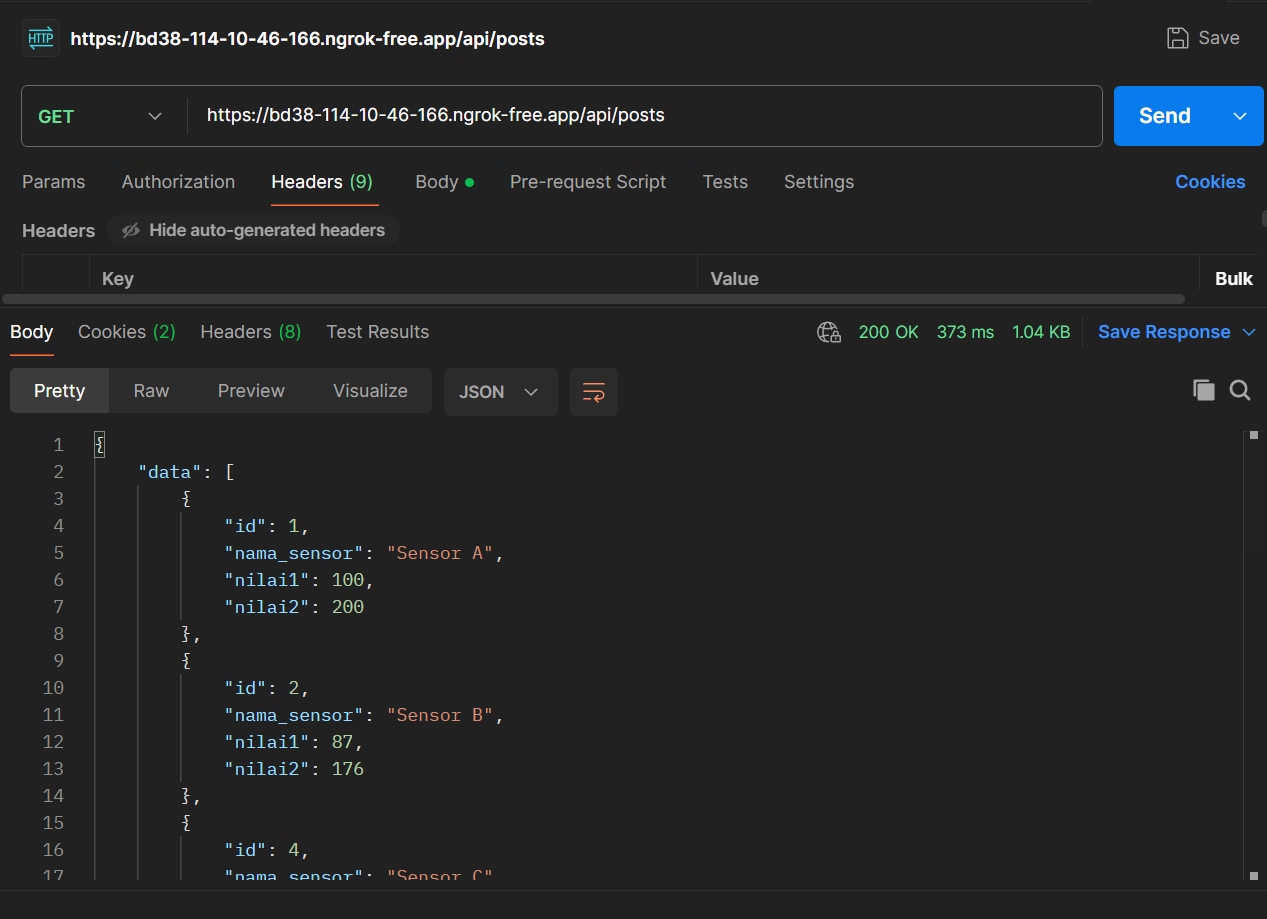
ngrok http <http://127.0.0.1:8000/>

Catat URL dari Ngrok untuk mengakses API secara publik.

1. Akses API dari URL Ngrok







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

* API Laravel berhasil dibuat dengan endpoint CRUD untuk TransaksiSensor.
* Ngrok memungkinkan akses API dari luar jaringan lokal.
* Pengujian menggunakan Postman menunjukkan API berjalan dengan baik.

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

**4.1 Kode Program (2025\_02\_25\_140802\_create\_transaksi\_sensors\_table.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('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');

}

};

**4.2 Kode Program(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 = [];

}

**4.3 Kode Program(TransaksiSensorResource.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,

];

}

}

**4.4 Kode Program(routes/api.php)**

<?php

use Illuminate\Auth\Middleware\Authenticate;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Route;

Route::get('/user', function (Request $request) {

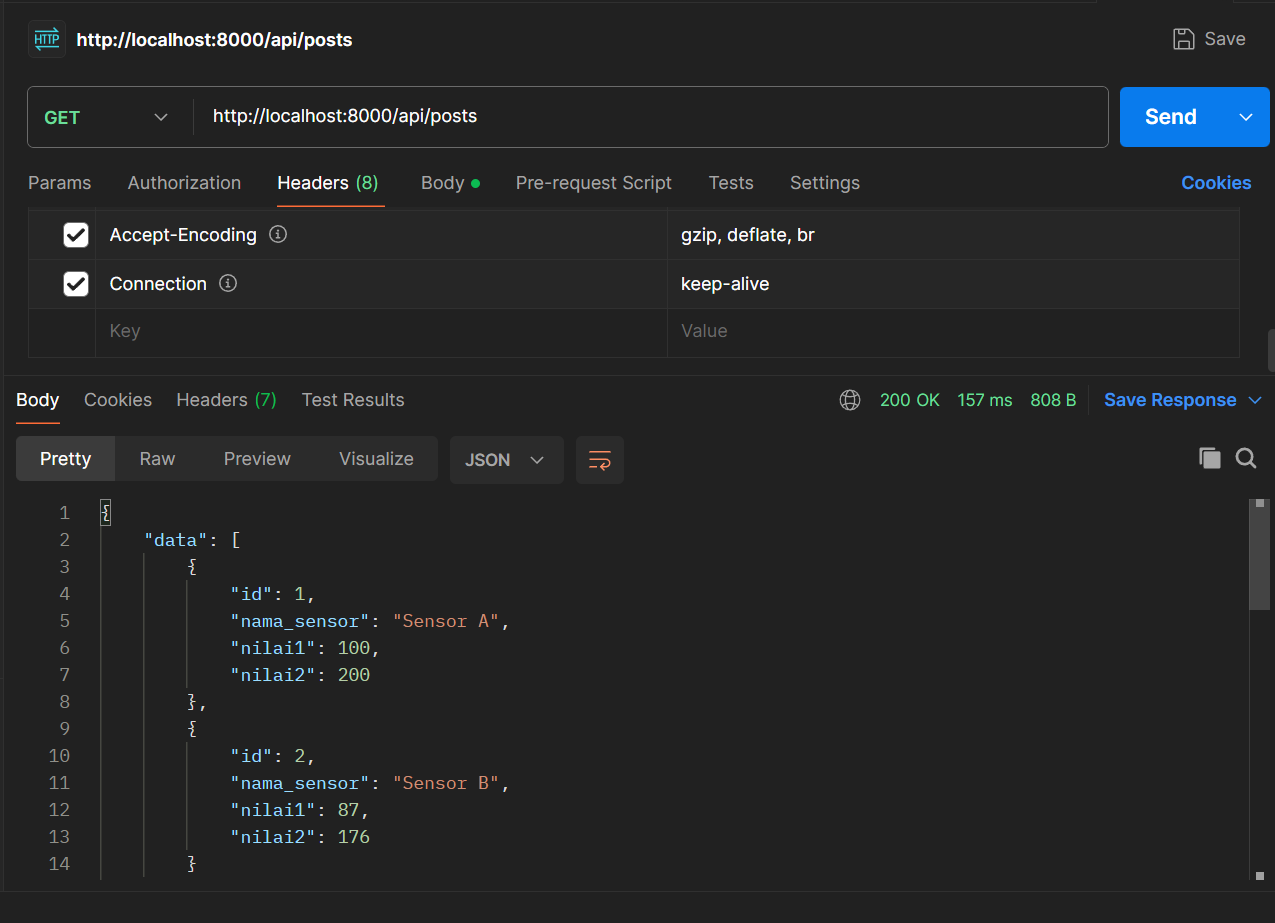
return $request->user();

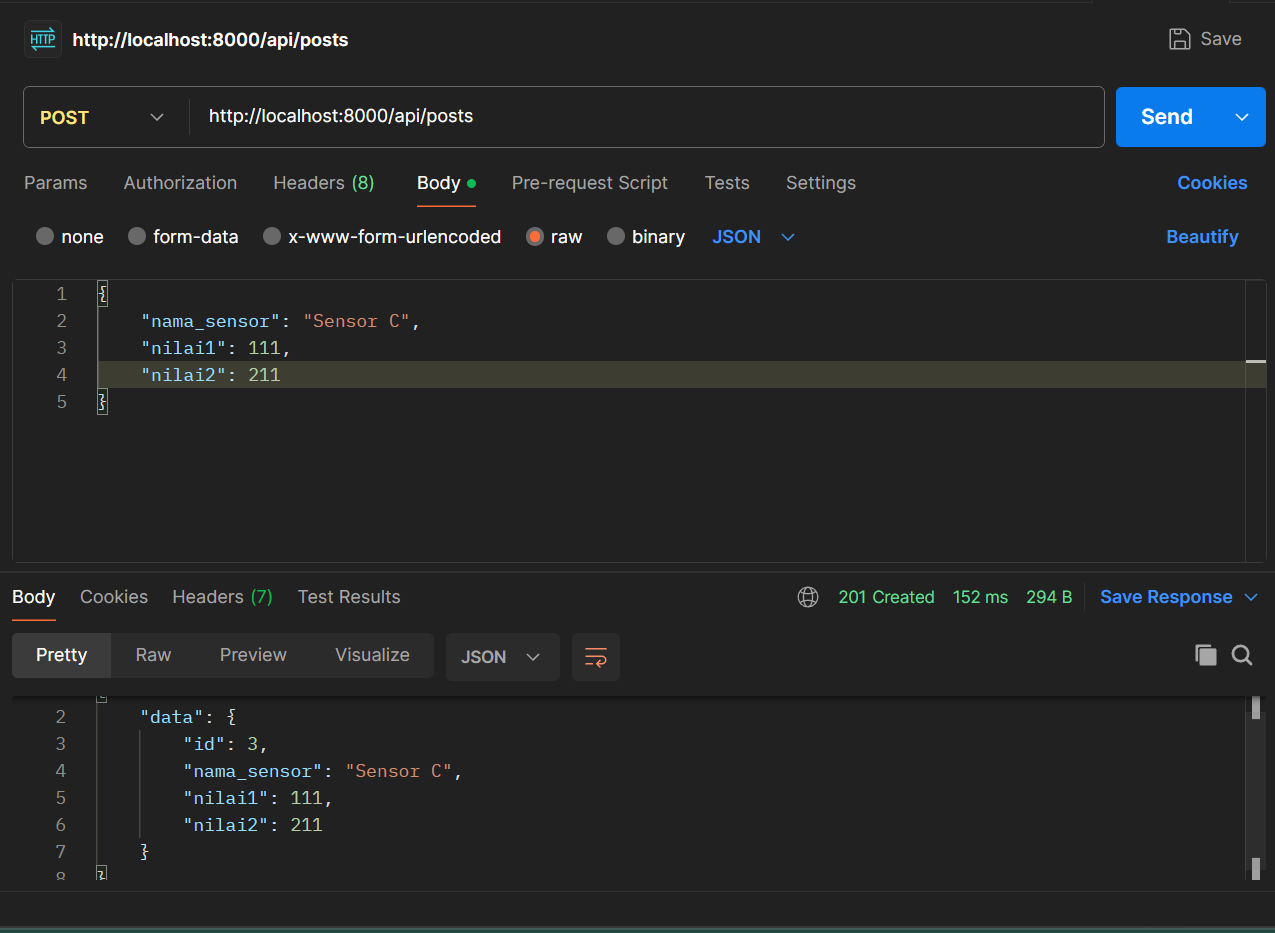
})->middleware(Authenticate::using('sanctum'));

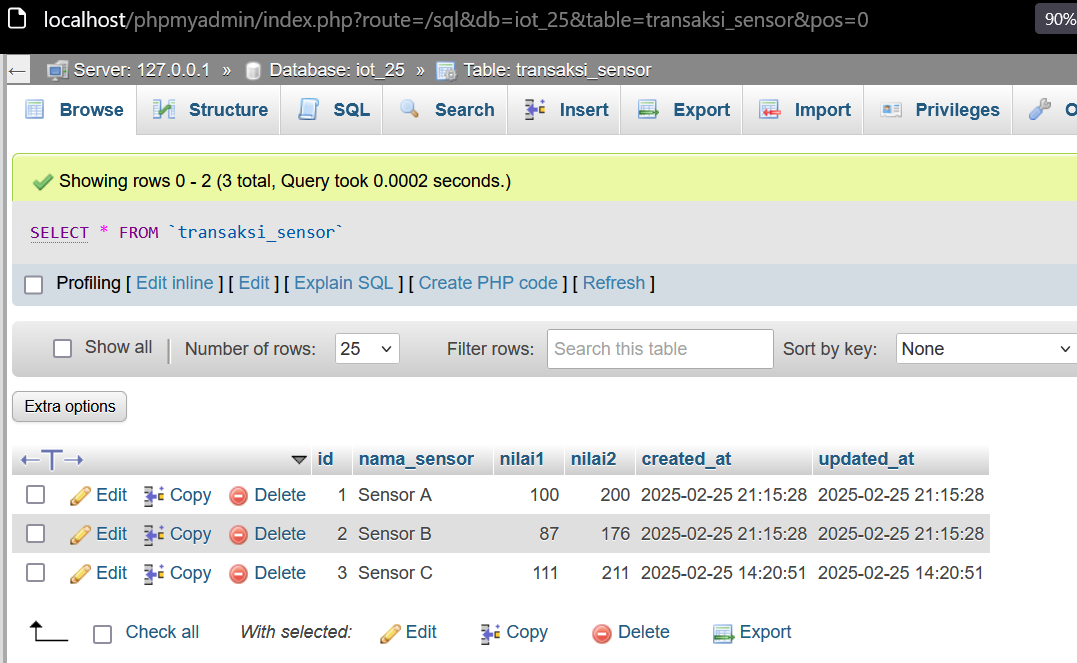
//posts

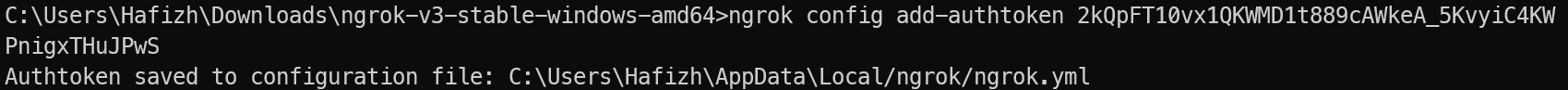
Route::apiResource('/posts', App\Http\Controllers\Api\TransaksiSensorController::class);

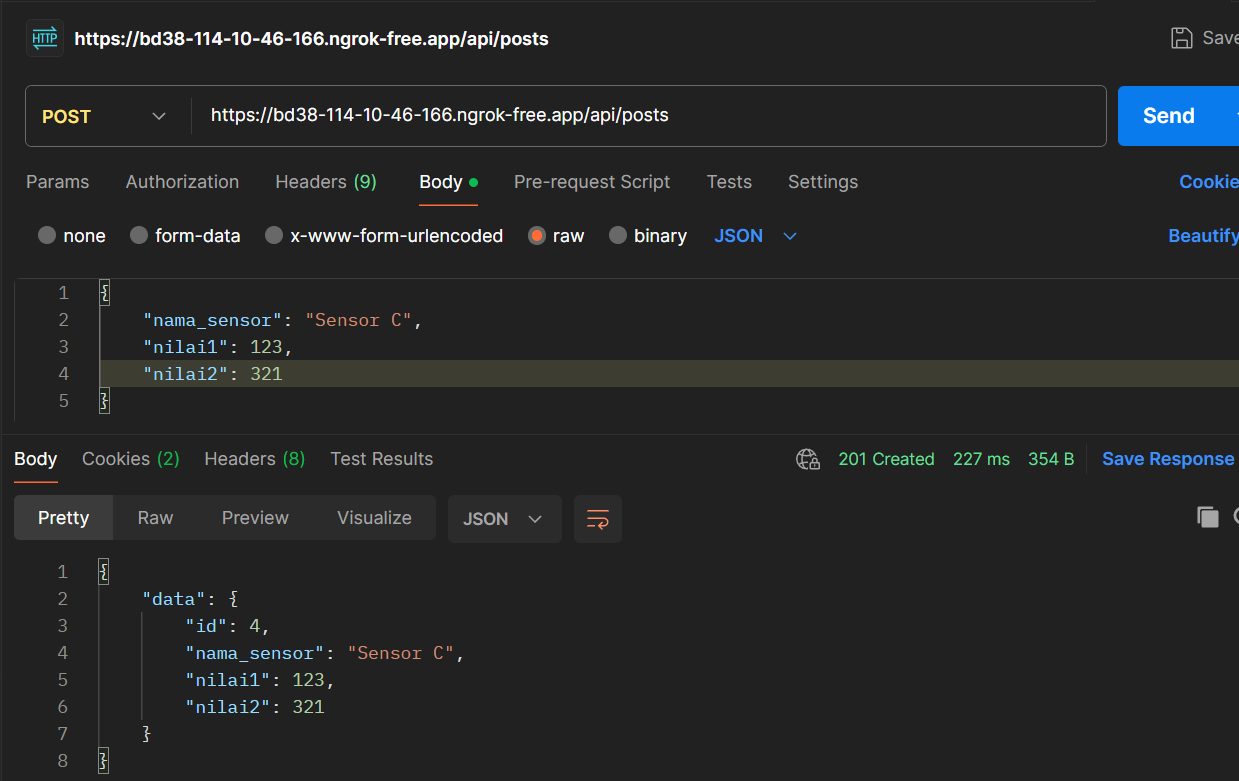
**4.3 Screenshot**

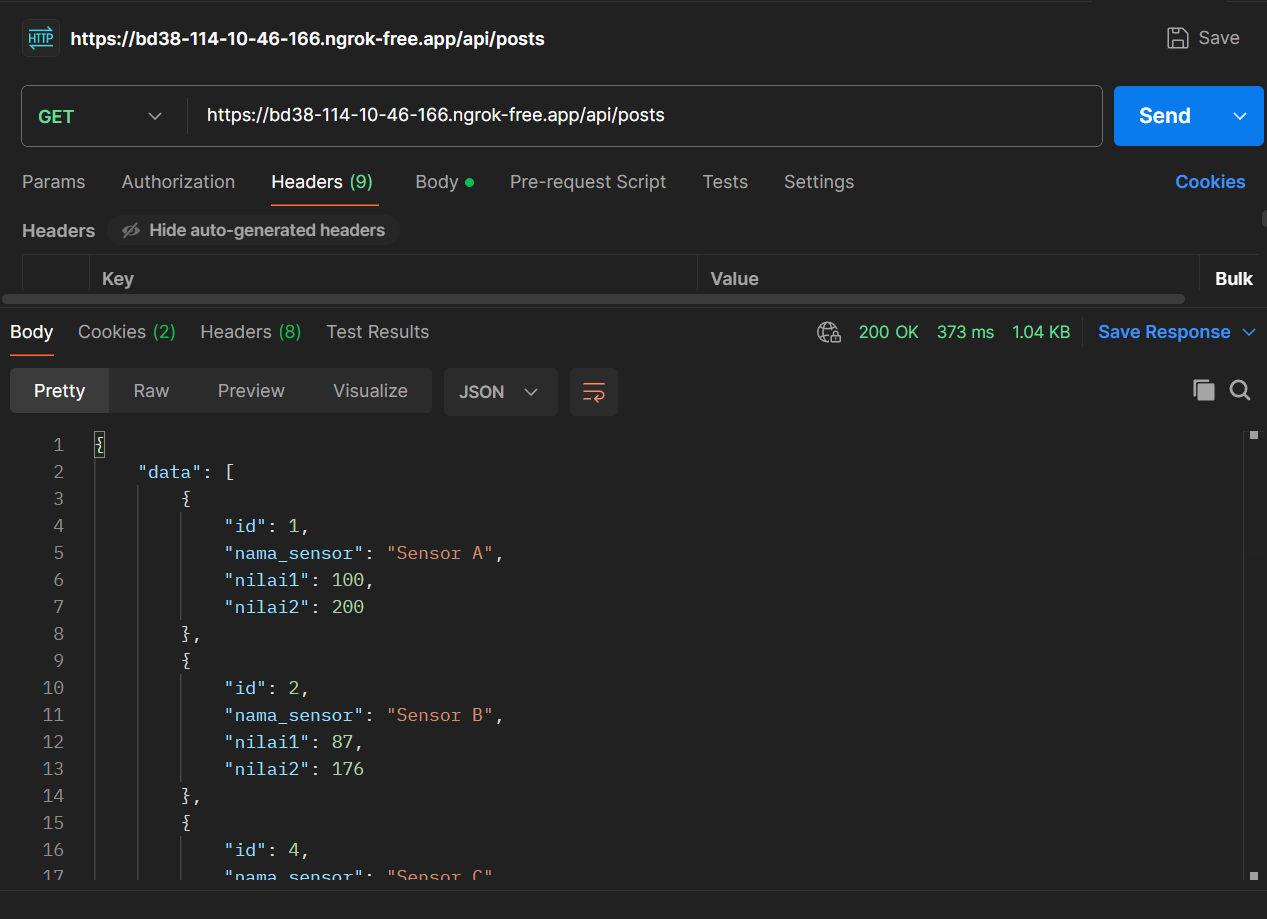
****

****

****

****

****

****