**Pratice Making API Using Laravel 11 and Ngrok**

*Damar Lintang Priyatama*

*Fakultas Vokasi, Universitas Brawijaya*

*Email:* [*damarpriya138@gmail.com*](mailto:damarpriya138@gmail.com)

**Abstract**

The Internet of Things (IoT) has revolutionized the way devices and systems communicate, requiring robust and scalable platforms for integration. One of the essential components in this integration is the Application Programming Interface (API), which facilitates communication between devices and web applications. This paper discusses the practice of creating APIs using **Laravel 11**, a modern PHP framework, alongside **Ngrok**, a tool that exposes local servers to the internet. The use of Laravel 11 offers an elegant and efficient way to develop secure, maintainable APIs, while Ngrok simplifies the testing and debugging process by providing a secure tunnel to locally hosted APIs. The practice involves setting up a Laravel 11 project, creating RESTful API endpoints, and using Ngrok to make these endpoints accessible over the internet for IoT device interactions. By combining Laravel's robust API functionalities with Ngrok's tunnel service, developers can streamline their workflow and create reliable, secure, and scalable APIs for IoT applications. This paper will demonstrate the key steps involved in API creation and the potential benefits of using these tools in real-world IoT projects.

*Keywords—* *IoT, RESTful API, Local Tunneling, Scalable Systems, Secure API Development.*

**1. Introduction**

**1.1 Background**

This project aims to demonstrate the process of creating a RESTful API using Laravel 11 to manage IoT sensor data. The API will enable users to interact with sensor data, including creating, retrieving, updating, and deleting records. The development of IoT systems often requires a reliable and secure API that allows for data retrieval, updates, and interactions. Laravel 11, with its extensive set of features, provides an excellent platform for building such APIs. The process involves setting up a database to store sensor data, creating models and migrations for database management, and developing API controllers to handle requests. Furthermore, using Ngrok allows the locally running Laravel application to be exposed to the internet, making it accessible for devices, such as IoT sensors or simulators like Wokwi, to interact with the API.

**1.2 Purpose Experience**  
The purpose of this project is to practice and showcase the integration of **Laravel 11** with **Ngrok** to create a scalable and secure API that can interact with IoT devices. The API will be responsible for managing sensor data, which includes storing sensor readings (e.g., nilai1 and nilai2), as well as other sensor attributes. This project will also demonstrate how to use **Postman** to test API endpoints locally and externally after exposing the application using Ngrok.

**2. Methodology**

**2.1 Tools & Materials**

-Laptop Asus

-Vscode

-laravel 11

-Postman

-Xampp

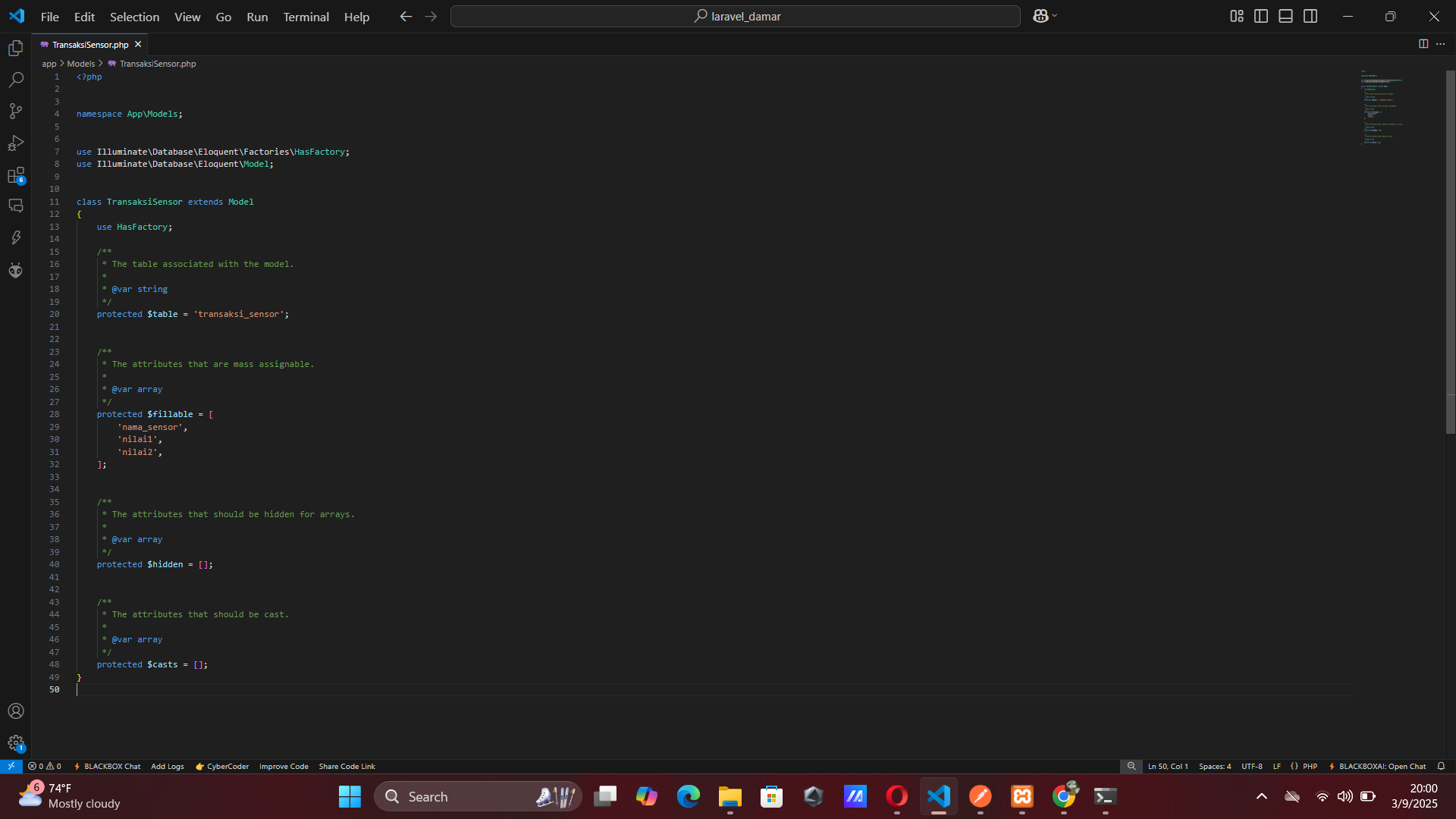
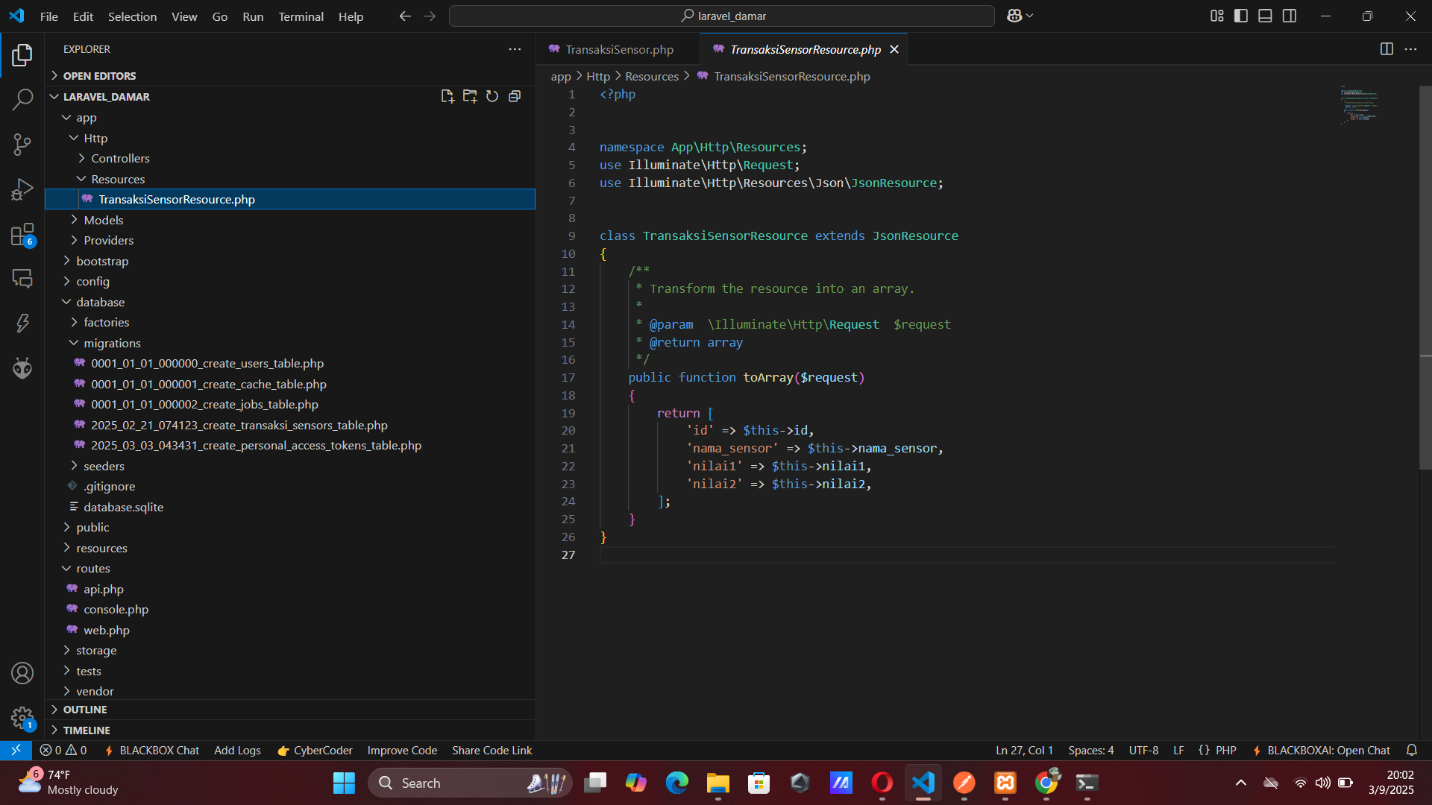
-Ngrok

-Composer

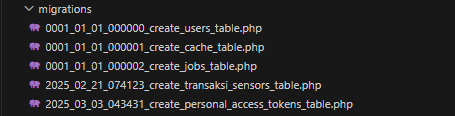
**2.2 Implementation Steps**

-Create a MySQL database iot\_25 in phpMyAdmin

-Run the command:

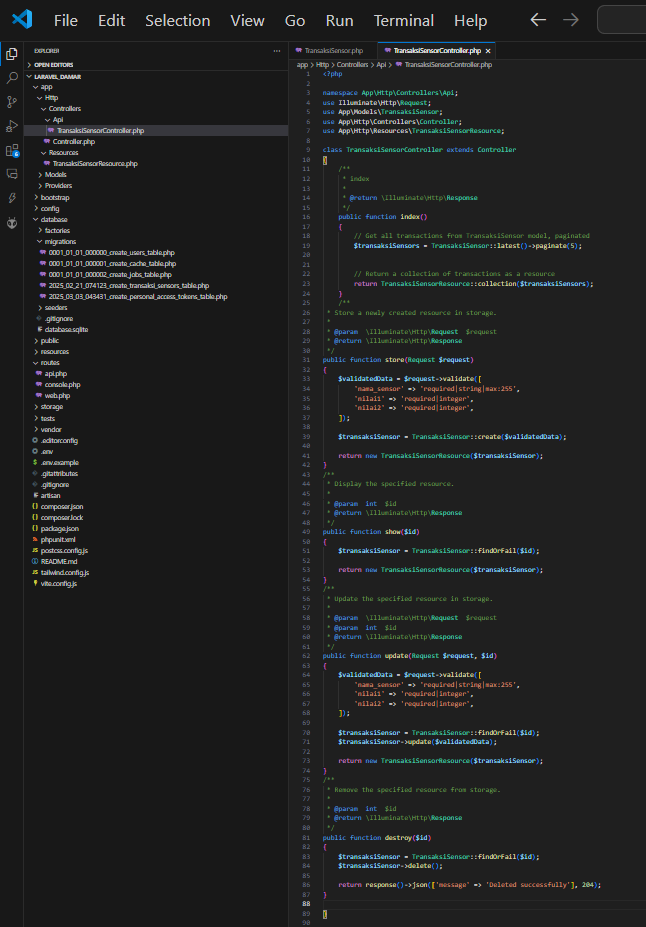
php artisan make:model TransaksiSensor and php artisan make:resource TransaksiSensorResource

-Modify the migration file to create the transaksi\_sensor table, then run:

php artisan migrate

-Create API Controller by run the command:

php artisan make:controller Api/TransaksiSensorController



-Define CRUD methods (index, store, show, update, destroy) in the controller to handle API requests.

-Test API with Postman by start the Laravel server:

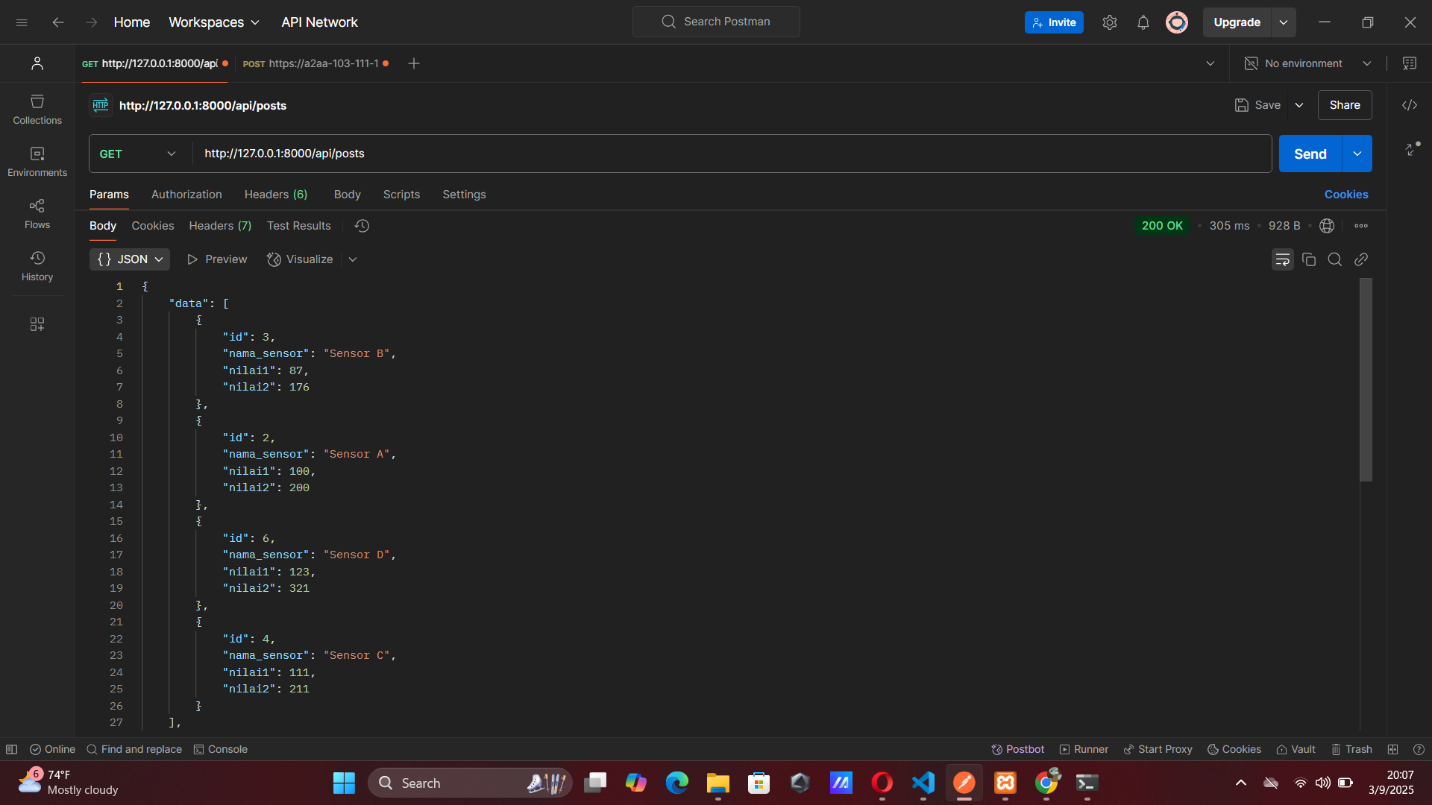
php artisan serve

-After that open Postman and test the API:

GET request to retrieve data: http://127.0.0.1:8000/api/posts

POST request to insert data into the database (set headers and body accordingly).

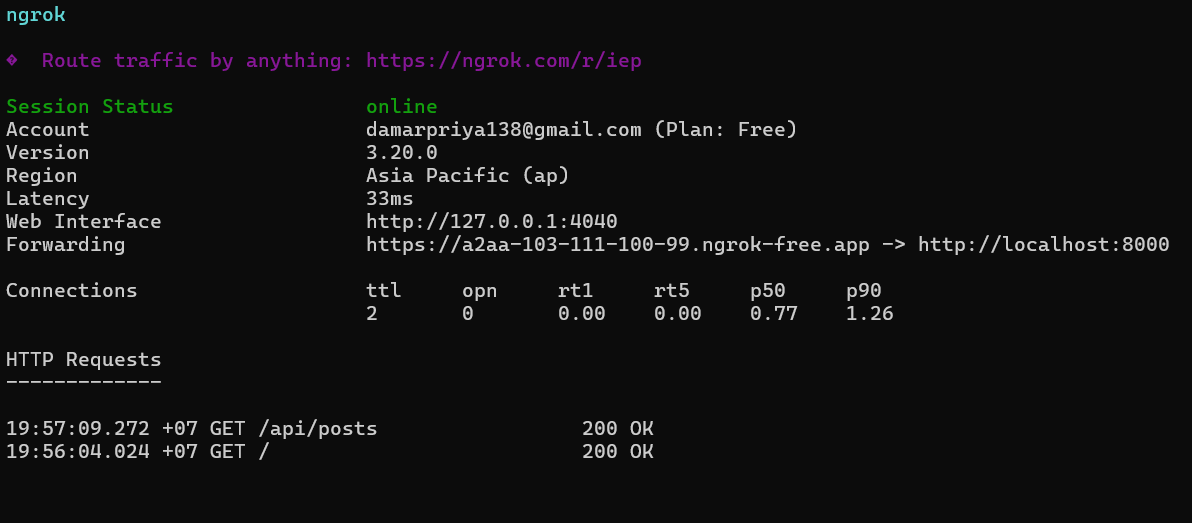
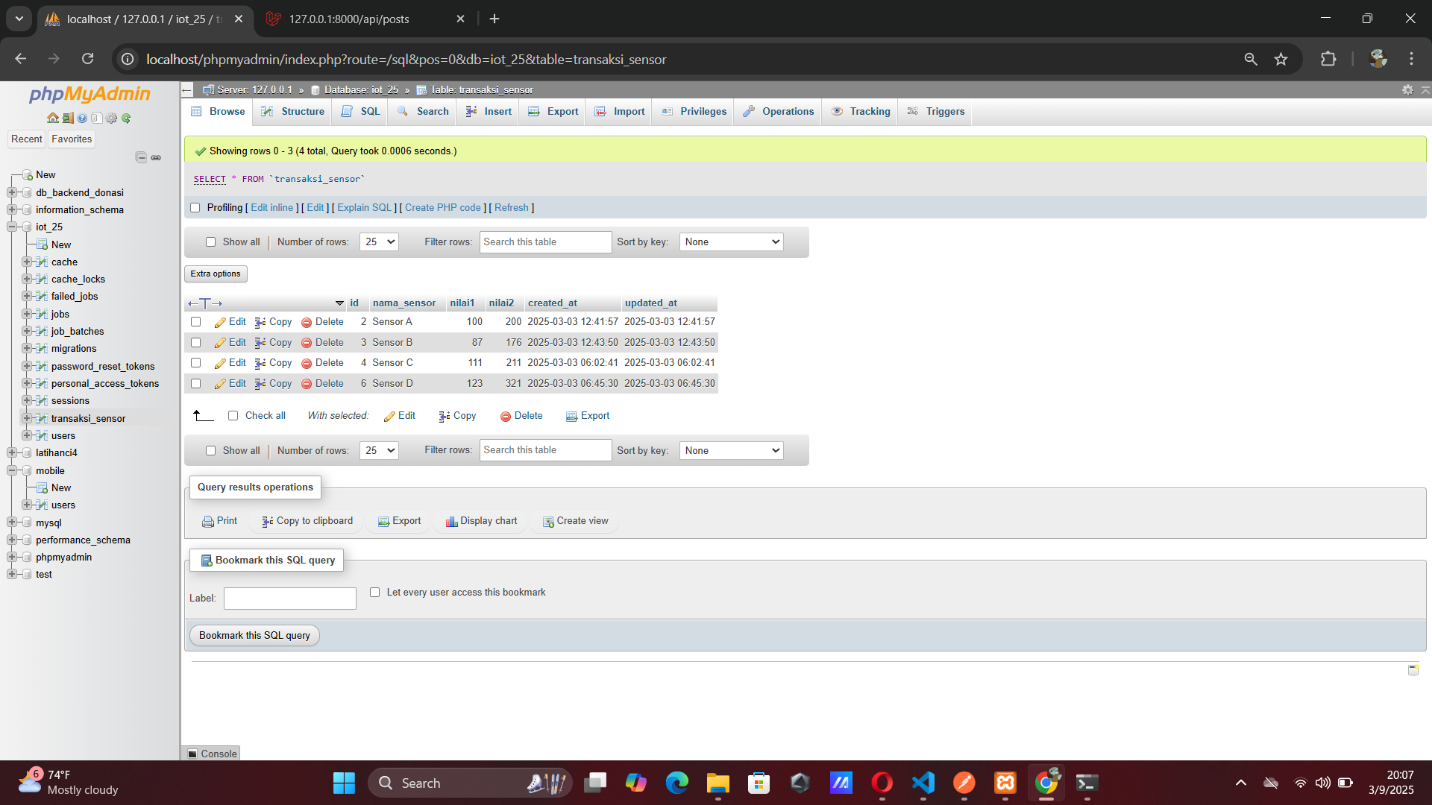
-Expose API using Ngrok and run the following command to expose the Laravel server to the internet:

ngrok http http://localhost:8000

-Test to Use exposed the Ngrok in Postman to perform GET and POST requests.

**3. Results and Discussion**

**3.1 Experimental Results**

* **ngrok http** [**http://localhost:8000**](http://localhost:8080) 
* Result data API by Postman