### **Practical API Development Using Laravel 11 and Ngrok**

*Candra Wahyu Perdana*

Vocational Faculty, Brawijaya University

Email: candraega28@gmail.com

**Abstract**

This experiment focuses on the development of a RESTful API using Laravel 11, integrated with Ngrok for remote access. The API is designed to manage sensor transaction data stored in a MySQL database named **iot\_25**. The implementation includes creating a model, database migration, resource transformation, and API controller for performing CRUD operations. Testing is conducted using **Postman**, ensuring the API correctly retrieves, inserts, updates, and deletes data. The results confirm that Laravel 11 effectively handles API requests, providing structured JSON responses.

*Keywords—Laravel 11, API Development, Ngrok, Postman, RESTful API, CRUD, PHP, MySQL*

**1. Introduction**

**1.1 Background**

The rise of Internet of Things (IoT) applications has increased the need for efficient API development to manage sensor data. Laravel, a powerful PHP framework, simplifies API development through built-in functionalities like Eloquent ORM, migrations, and resource handling. By integrating **Ngrok**, we can expose the local development server to the internet for testing and debugging.

This report details the step-by-step process of building an API for **sensor transactions**, covering database setup, model creation, routing, and testing using **Postman**.

**1.2 Objectives**

* Develop a RESTful API using Laravel 11 to manage sensor transaction data.
* Implement database migrations and API controllers for CRUD operations.
* Test the API endpoints using Postman and expose the local server with Ngrok.

**2. Methodology**

**2.1 Tools & Materials**

**Software:**

* Laravel 11
* Ngrok
* Postman
* PHPMyAdmin / MySQL
* Composer

**Development Environment:**

* Laravel Local Server (php artisan serve)
* API Testing (Postman)
* Remote Access (Ngrok)

#### **2.2 Implementation Steps**

### **1. Setting Up the Database**

* + Create a new database named **iot\_25** in **phpMyAdmin**.

**2. Creating the Laravel Model**

* + Run the following command to generate a model for handling **sensor transactions**:

php artisan make:model TransaksiSensor

### **3. Modifying the Migration File**

* + RModify the file **2025\_02\_21\_074123\_create\_transaksi\_sensors\_table.php** to define the database schema:

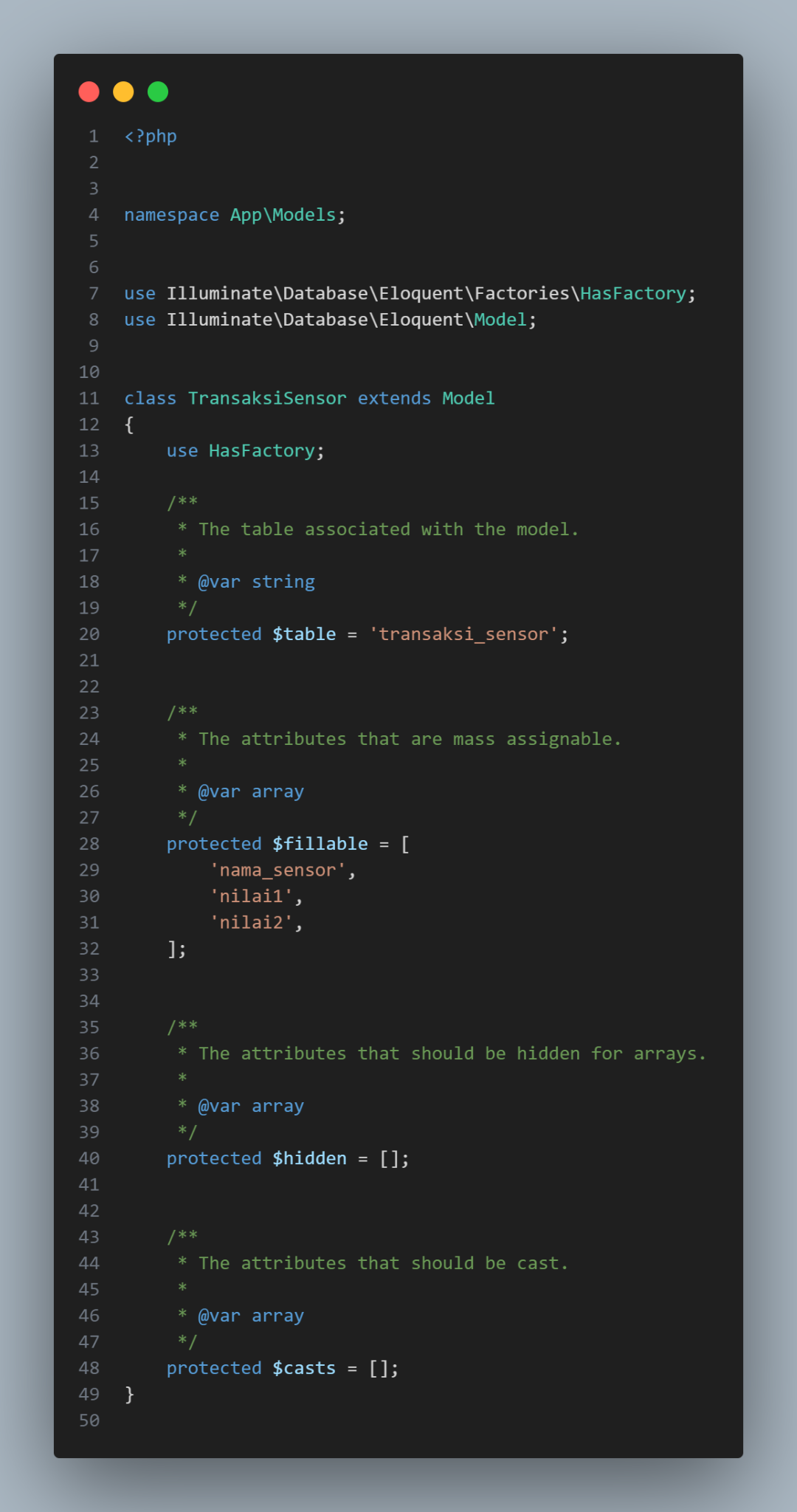


Run the migration command:

php artisan migrate

### **4. Creating the Model**

* + Modify **app/Models/TransaksiSensor.php**:



### **5. Creating API Resources**

Run:

php artisan make:resource TransaksiSensorResource

Modify **app/Http/Resources/TransaksiSensorResource.php**:



**6. Creating API Controller**

* + Run php artisan make:controller Api/TransaksiSensorController
  + Modify **app/Http/Controllers/Api/TransaksiSensorController.php**:



**7. Defining API Routes**

* + Modify **routes/api.php**:



Run : php artisan route:list

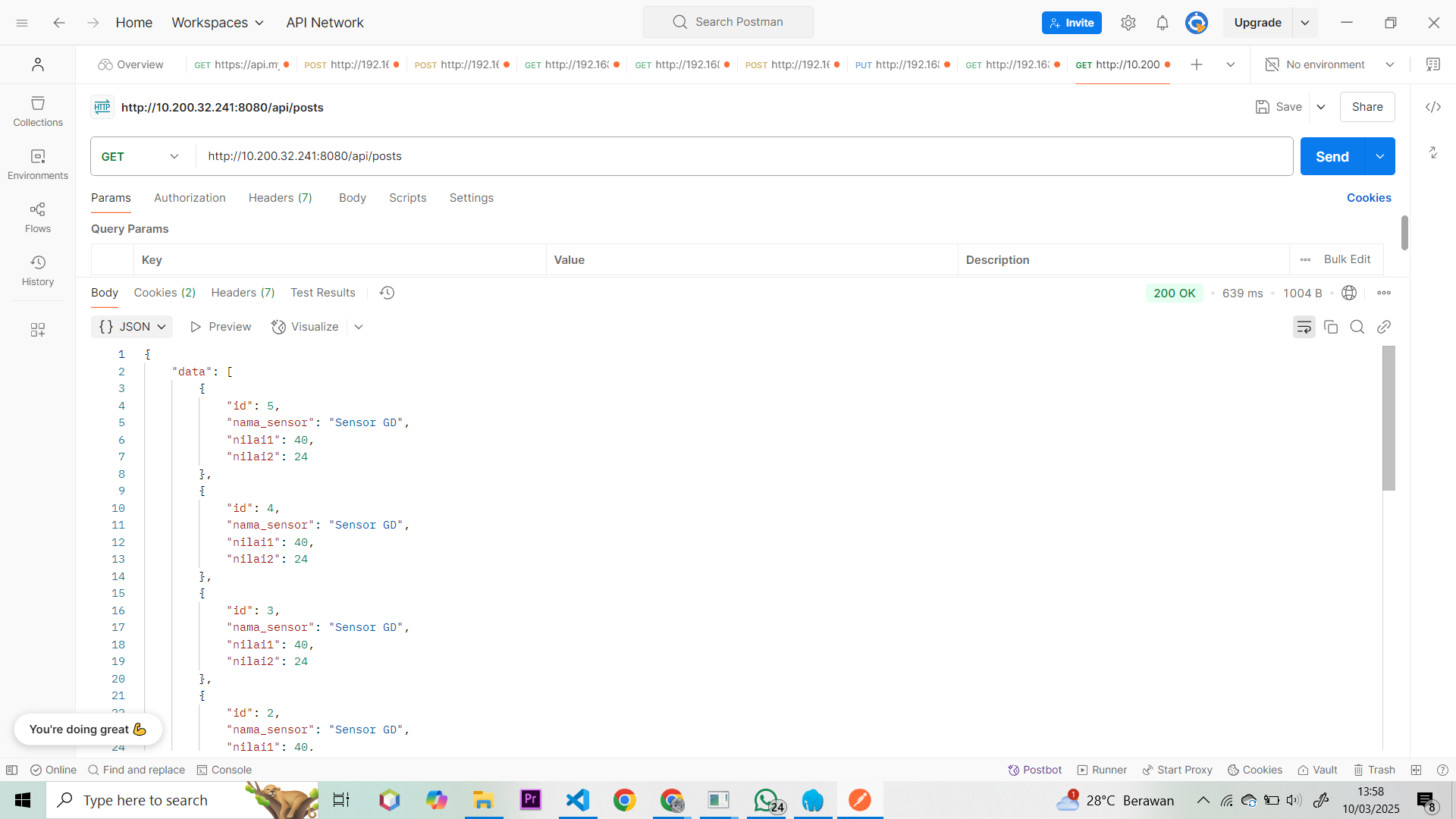
**8. Testing API with Postman**

1. Start Laravel Server : php artisan serve

2. Open Postman and test **GET** request:

* **URL:** http://127.0.0.1:8000/api/posts
* **Method:** GET
* **Response Format:** JSON

Expected Output :



**9. Exposing API via Ngrok**

Run : ngrok https 8000

Copy the generated **public URL** and use it in Postman for remote API access.

### **Conclusion**

This report describes the process of building and testing a Laravel 11 API for sensor transactions. The API successfully performs CRUD operations and returns JSON responses. By integrating **Ngrok**, we can expose the API for remote testing, making it useful for IoT applications and real-time data collection.