**Laporan Minggu 10 (Chapter 15 Web Dashboard IoT) – Laravel + IoT Practicum**

**Building an IoT Dashboard Interface**

*Gunawan Danyarto*

*University of Brawijaya Vocation Faculty*

*Email : [gunawandany11@gmail.com](mailto:gunawandany11@g,ail.com)*

**Abstract**

This practicum involves creating a web-based dashboard interface to visualize sensor data in the form of graphs using Laravel. The data is collected from IoT sensors connected to an API and displayed using Laravel’s MVC architecture. The activity also includes exporting data into Excel format using the **maatwebsite/excel** package. The main goal of this experiment is to build a dynamic and real-time data visualization system using modern web technologies.

**1. Introduction**

**1.1 Background**

After successfully collecting sensor data via APIs and storing it in a database, the next step is to display that data in a user-friendly interface. Data visualization is essential in IoT systems to allow users to quickly analyze the environment or monitored system. Laravel, as a modern PHP framework, provides strong support for this need with its MVC architecture and a wide variety of packages, such as maatwebsite/excel.

**1.2 Objective of the Experiment**

· Visualize IoT sensor data as charts using Laravel.

· Create a controller and view to render the graphs.

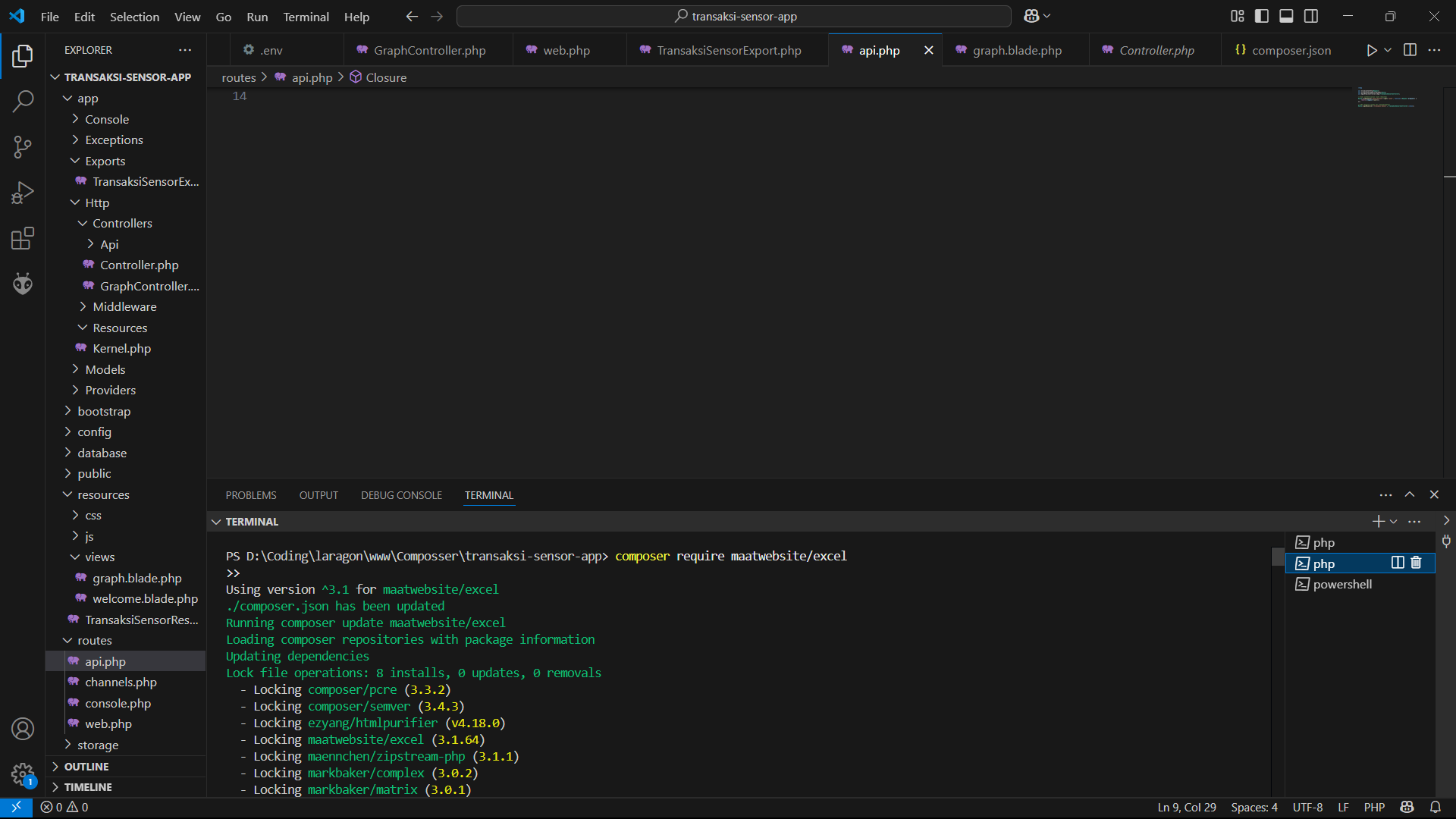
· Provide an export feature to download sensor data in Excel format.

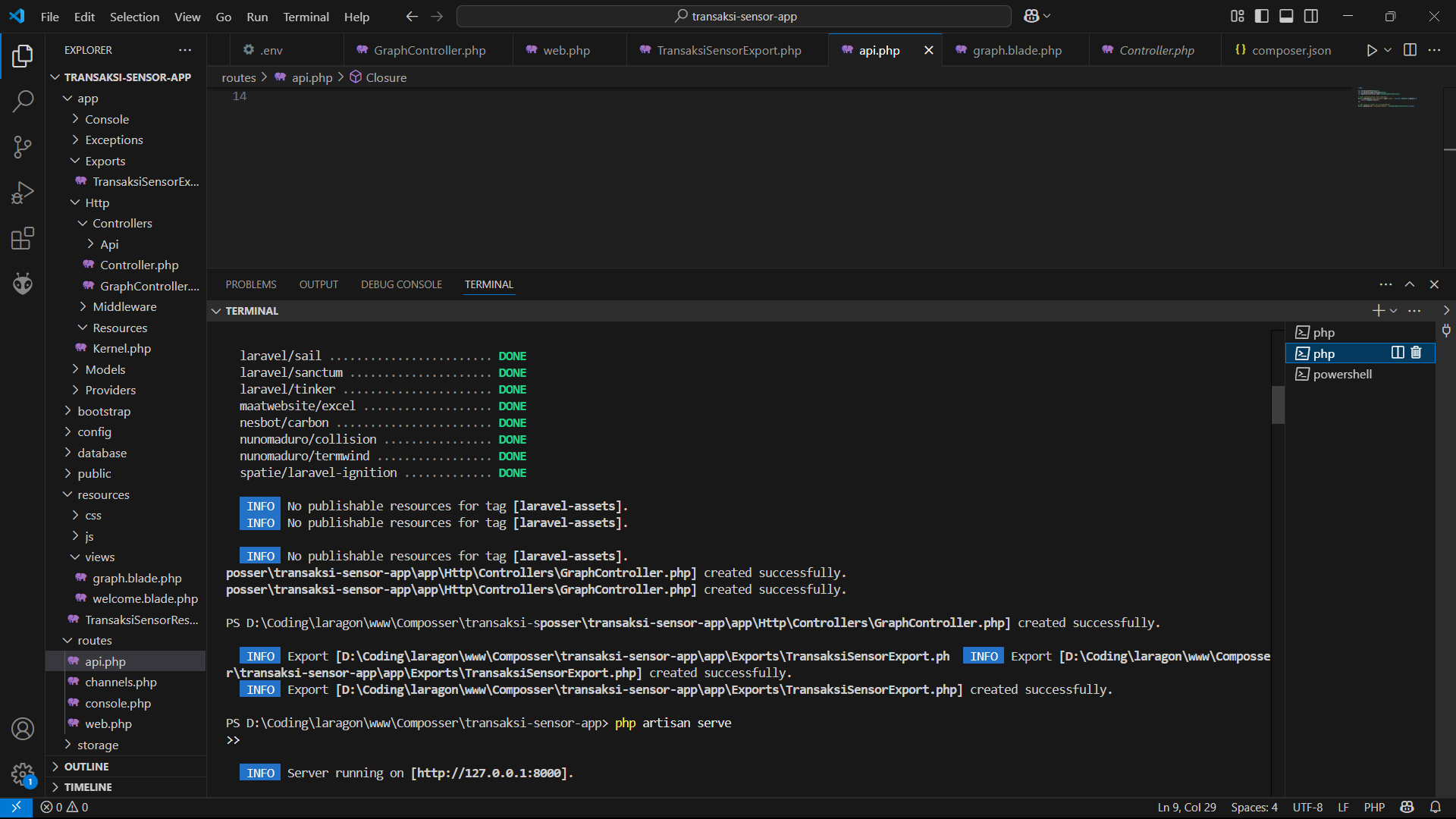
· Design a dashboard interface to support data-driven decision-making in IoT environments.

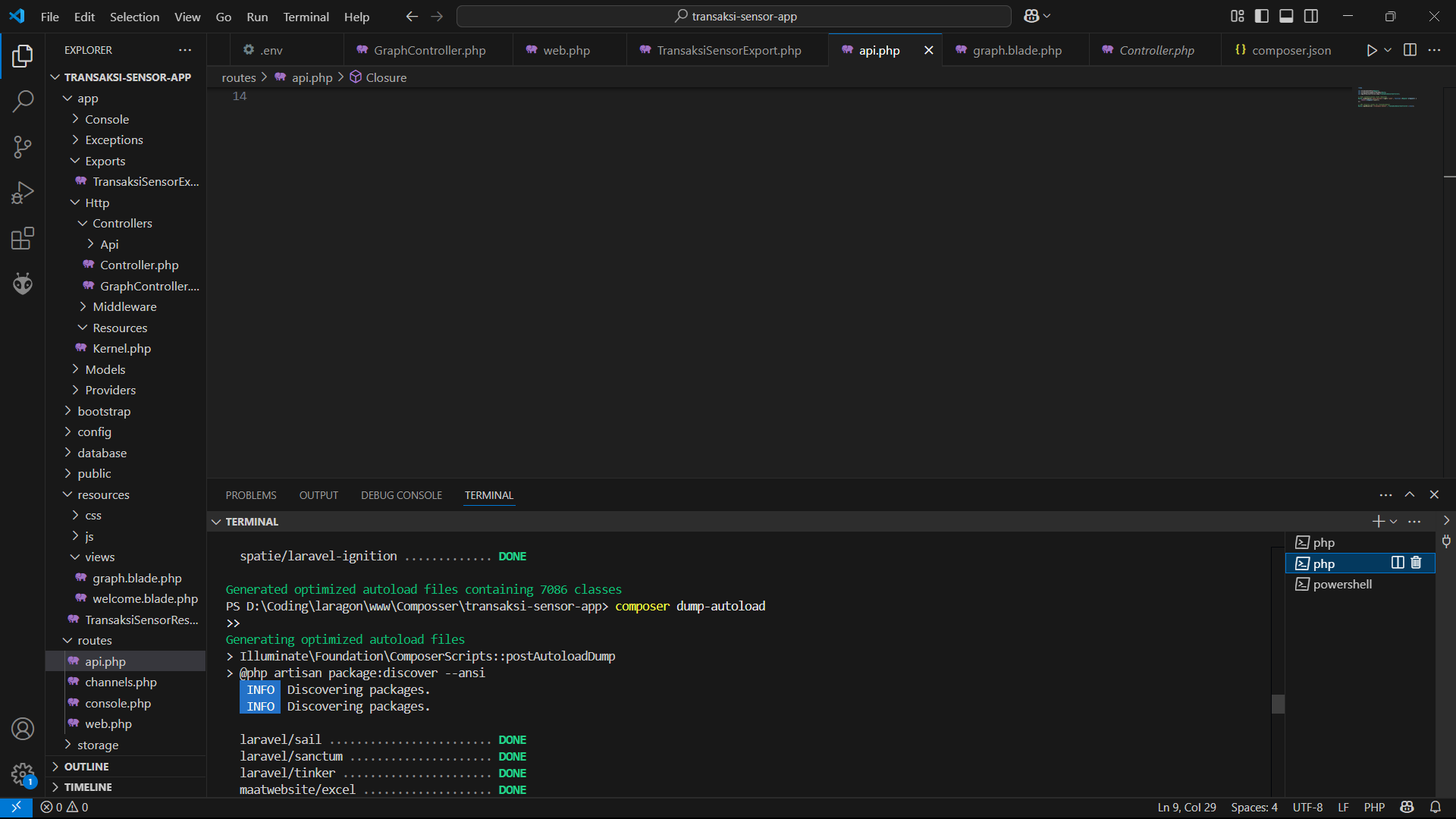
**2. Methodology**

### 2.1 Tools & Materials

|  |  |
| --- | --- |
| No | Item |
| 1 | Laravel Framework |
| 2 | Composer |
| 3 | VS Code + Terminal |
| 4 | IoT Sesnor Data (DB) |
| 5 | Maatwebsite / excel Packages |
| 6 | MySQL Db and Laragon |
| 7 | Web Browser |







**2.2 Step-by-Step Implementation**

#### ****Step 1 – Open Laravel Project from Week 12****

Launch the Laravel project created during Practicum 12 using Visual Studio Code.

Open the terminal.

#### ****Step 2 – Install Excel Export Package****

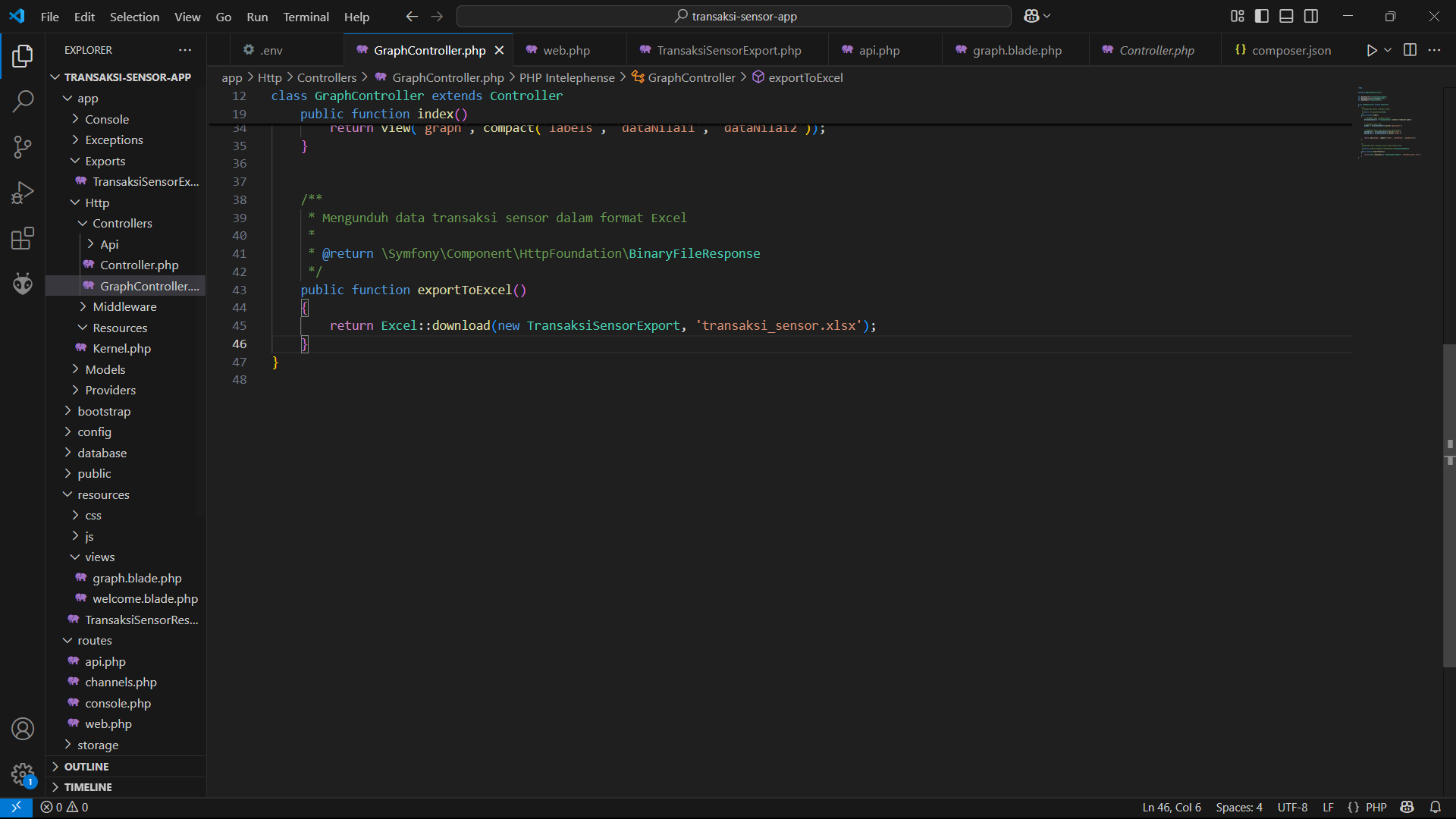
composer require maatwebsite/excel

#### ****Step 3 – Create Controller****

php artisan make:controller GraphController

#### ****Step 4 – GraphController Code****

Add the following code to GraphController.php:

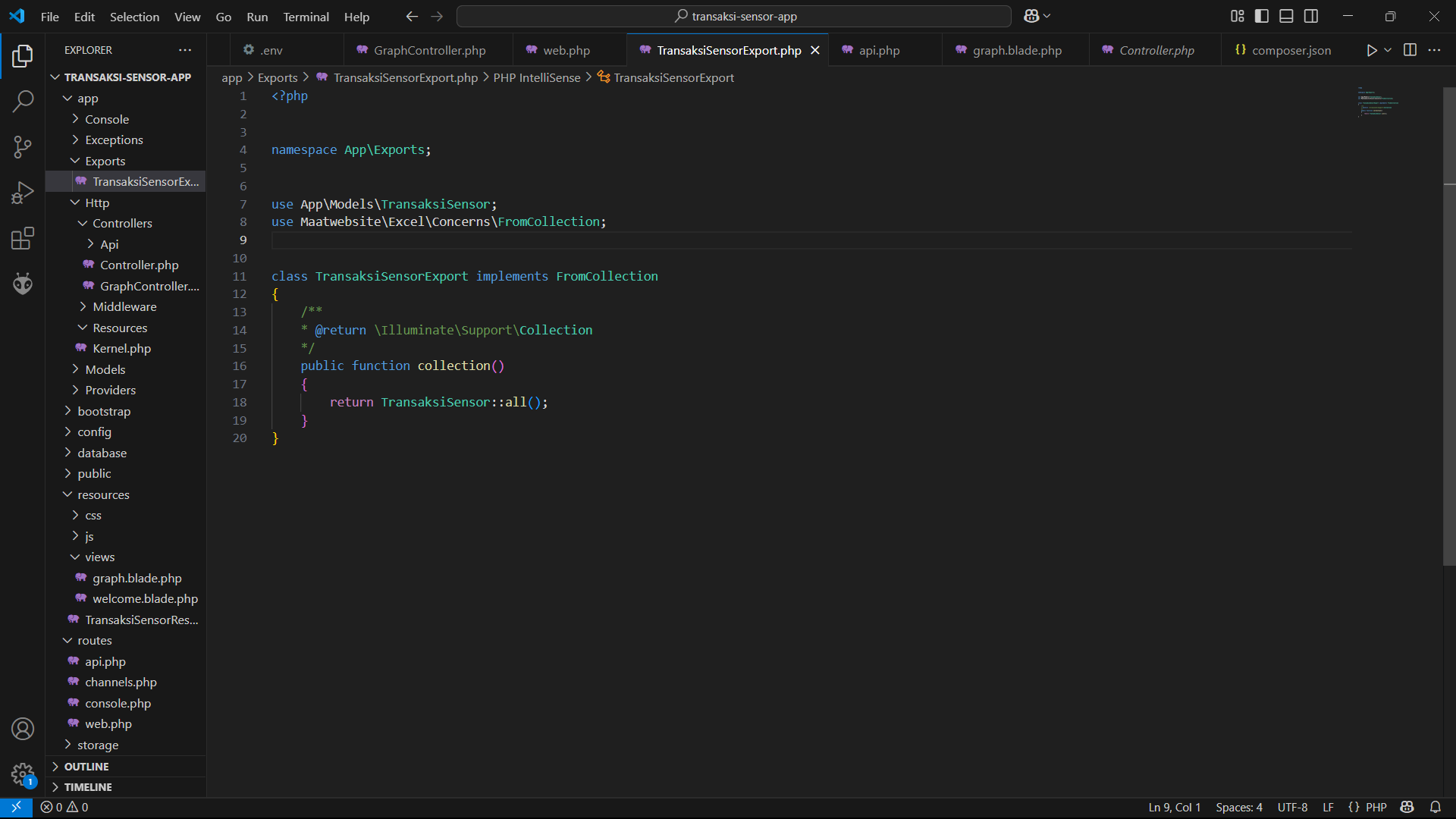


#### ****Step 5 – Create Export Class****

php artisan make:export TransaksiSensorExport --model=TransaksiSensor

#### ****Step 6 – Add Export Logic****

Inside TransaksiSensorExport.php, add :



#### ****Step 7 – Run the Laravel Server****

php artisan serve

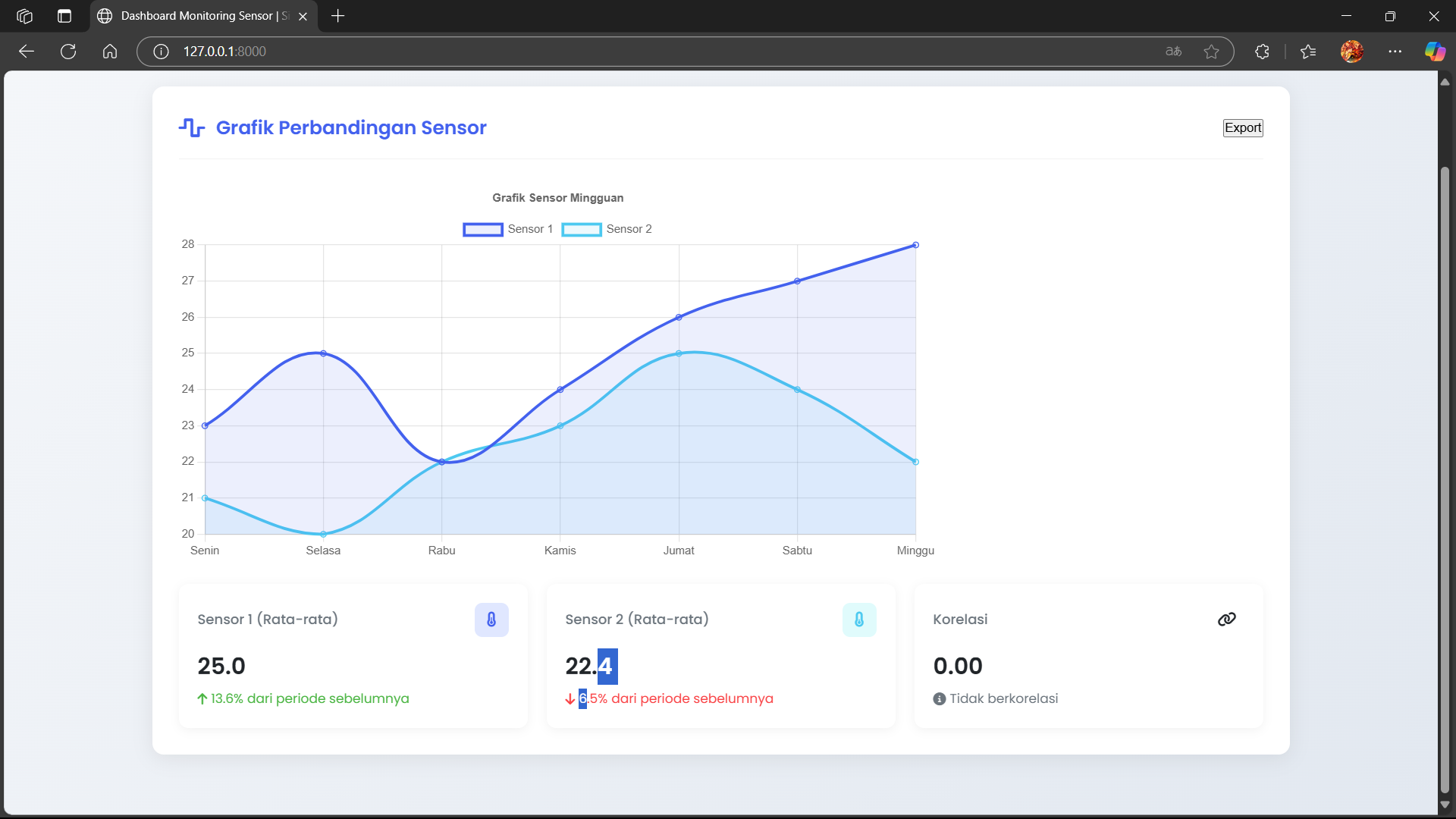
Access the dashboard at http://127.0.0.1:8000/graph

You will see a chart showing data from the IoT database (iot\_25).

### **Results :**

### **Web.php : Screenshot (1105) **3.1 Dashboard Chart Interface****

Displayed a real-time line or bar chart showing the last 10 IoT sensor readings.****

****

Labels are dynamically loaded from the database (nama\_sensor).

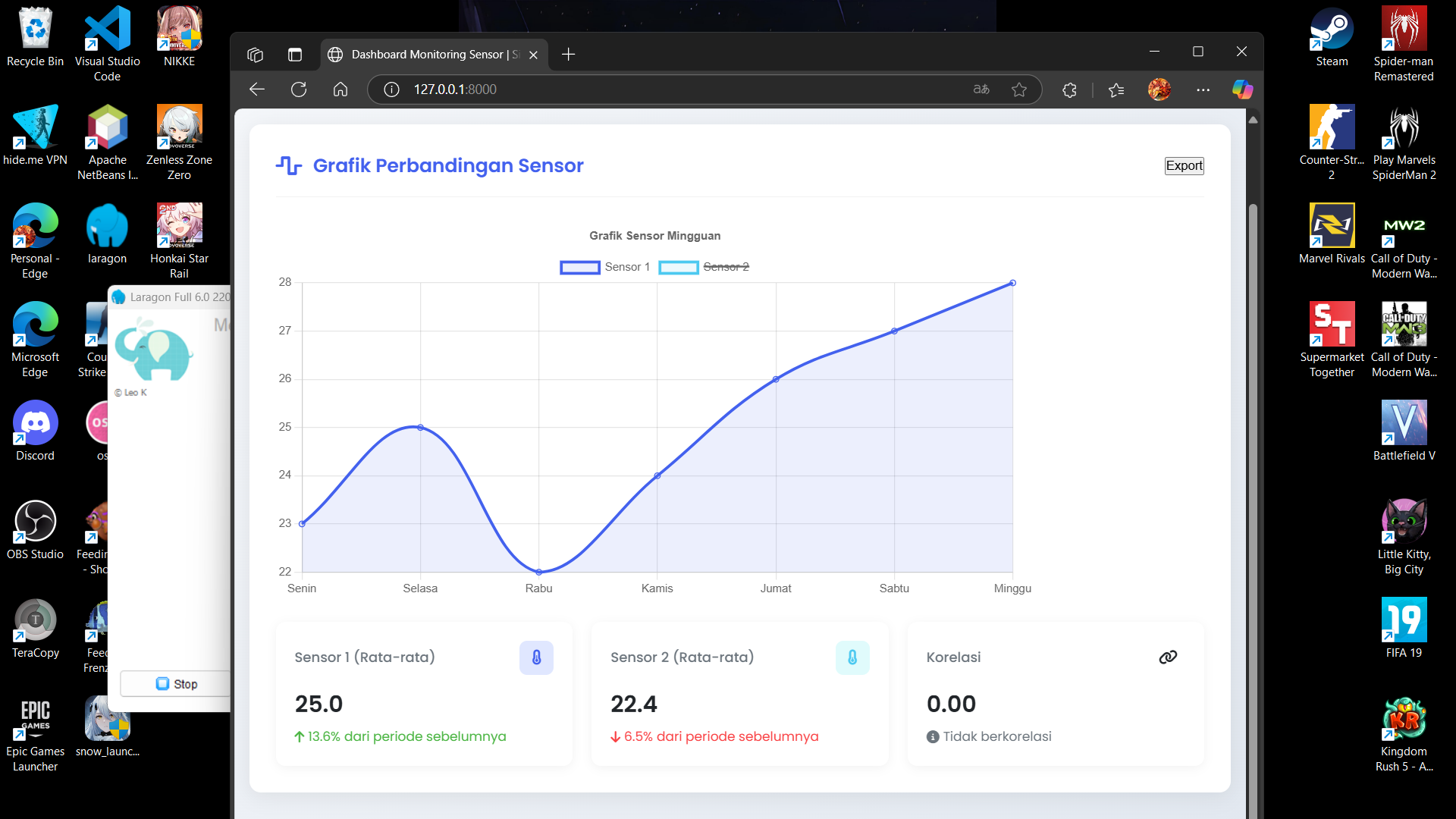
****

Chart data includes nilai1 and nilai2.

### ****3.2 Excel Export****

A button was added to export the displayed data as an .xlsx file.

Downloaded Excel file is named transaksi\_sensor.xlsx and contains all sensor data.

**5. Conclusion**

This chapter demonstrates the transition from simulation to real hardware using ESP32. With successful LED control, Wi-Fi scanning, and API data transmission from a DHT22 sensor, the hardware is now capable of functioning as an IoT node. PlatformIO, combined with proper wiring and code structuring, enables developers to confidently build and test embedded applications in real-world conditions.