**Creating a Web Dashboard Interface for IoT Systems**

*Damar Lintang Priyatama*

*Fakultas Vokasi, Universitas Brawijaya*

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

**Abstract**

The integration of Internet of Things (IoT) with modern web technologies enables real-time monitoring and data visualization to support informed decision-making. This report presents the implementation of a web-based dashboard interface using Laravel and Chart.js, specifically designed to monitor and compare IoT sensor data. It includes data exporting features using Excel and a visually engaging front-end layout for user-friendly analytics. The approach emphasizes modular Laravel structure, controller-based logic, and integration of third-party libraries for enhanced user experience and data utility.

*Keywords—* *IoT, Laravel, Chart.js, Web Dashboard, Data Visualization.*

**1. Introduction**

**1.1 Background**

The rapid advancement of IoT technologies allows devices to collect and exchange data autonomously. However, the challenge lies in how this data is presented and utilized. Dashboards are crucial in visualizing sensor data trends, patterns, and correlations that can drive operational efficiencies or scientific insights.

In a Laravel-based application environment, developers can leverage MVC architecture to structure the interface cleanly, handle sensor data efficiently, and export it when needed. This project builds on a previous Laravel setup (from Practice 12) and demonstrates how to create an interactive and dynamic dashboard

**1.2 Purpose Experience**  
The purpose of this project is to develop an interactive and informative web dashboard that enables users to monitor and compare data from IoT sensors in real time. This includes the ability to visualize trends between multiple sensor inputs and to export the gathered data for further offline analysis. Throughout the development process, the experience gained includes applying the Laravel MVC framework to structure backend logic and routing, utilizing Chart.js to render responsive line graphs, and integrating Excel export functionality using the Maatwebsite package.

Additionally, the process improved understanding of integrating data models, controlling views through Laravel controllers, and designing a clean and responsive user interface using HTML, CSS, and JavaScript. The project not only strengthens back-end and front-end development skills but also provides valuable insight into building scalable, data-centric web applications tailored to IoT environments.

**2. Methodology**

**2.1 Tools & Materials**

-Laptop Asus

-Vscode for coding

-laravel 11

-Xampp

-Composer

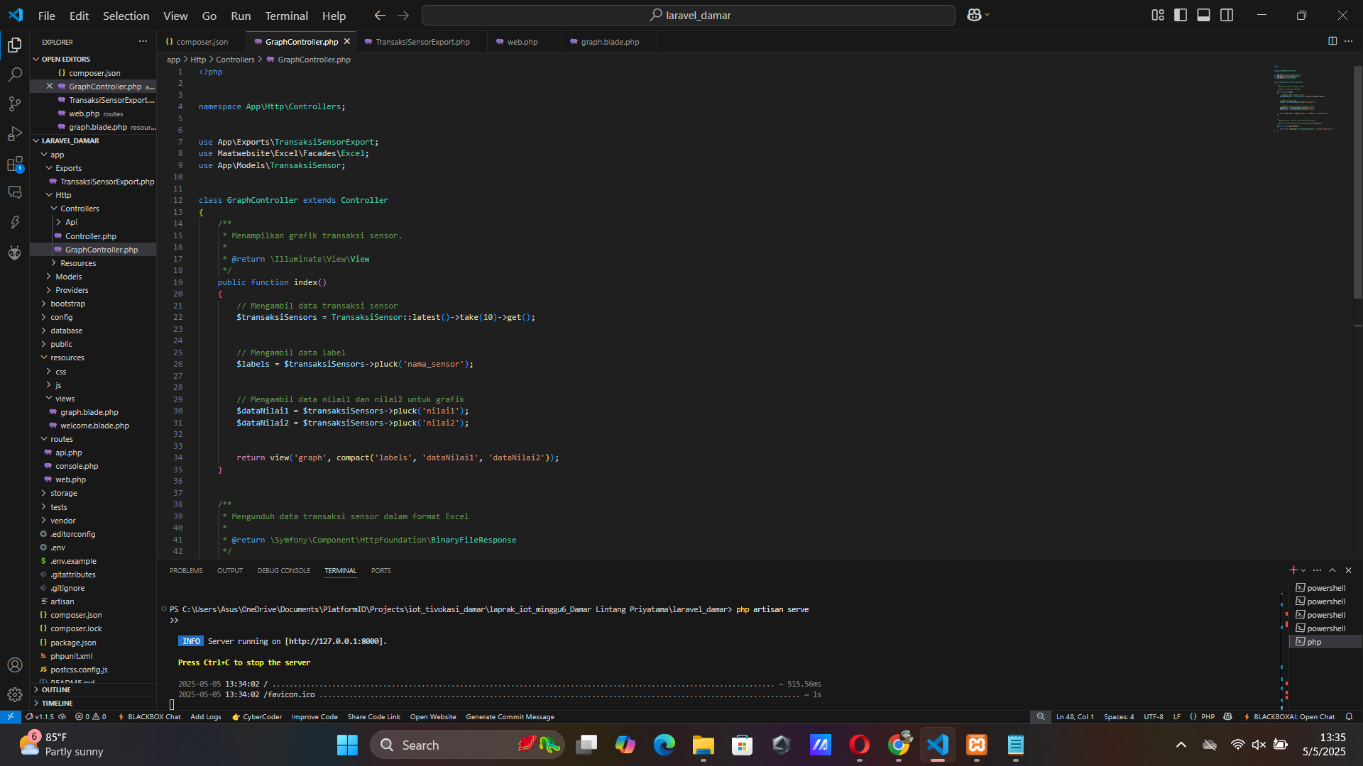
-Chart.js

-MySQL

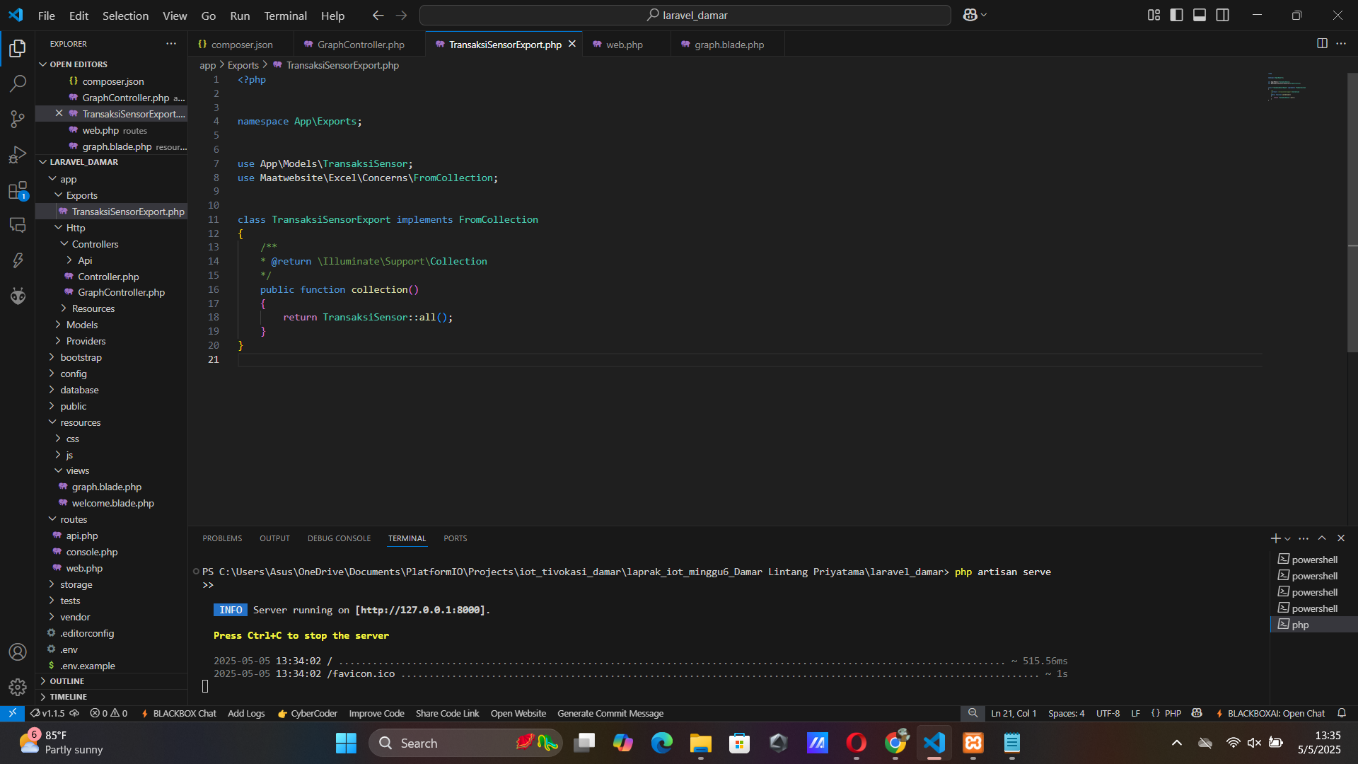
**2.2 Implementation Steps**

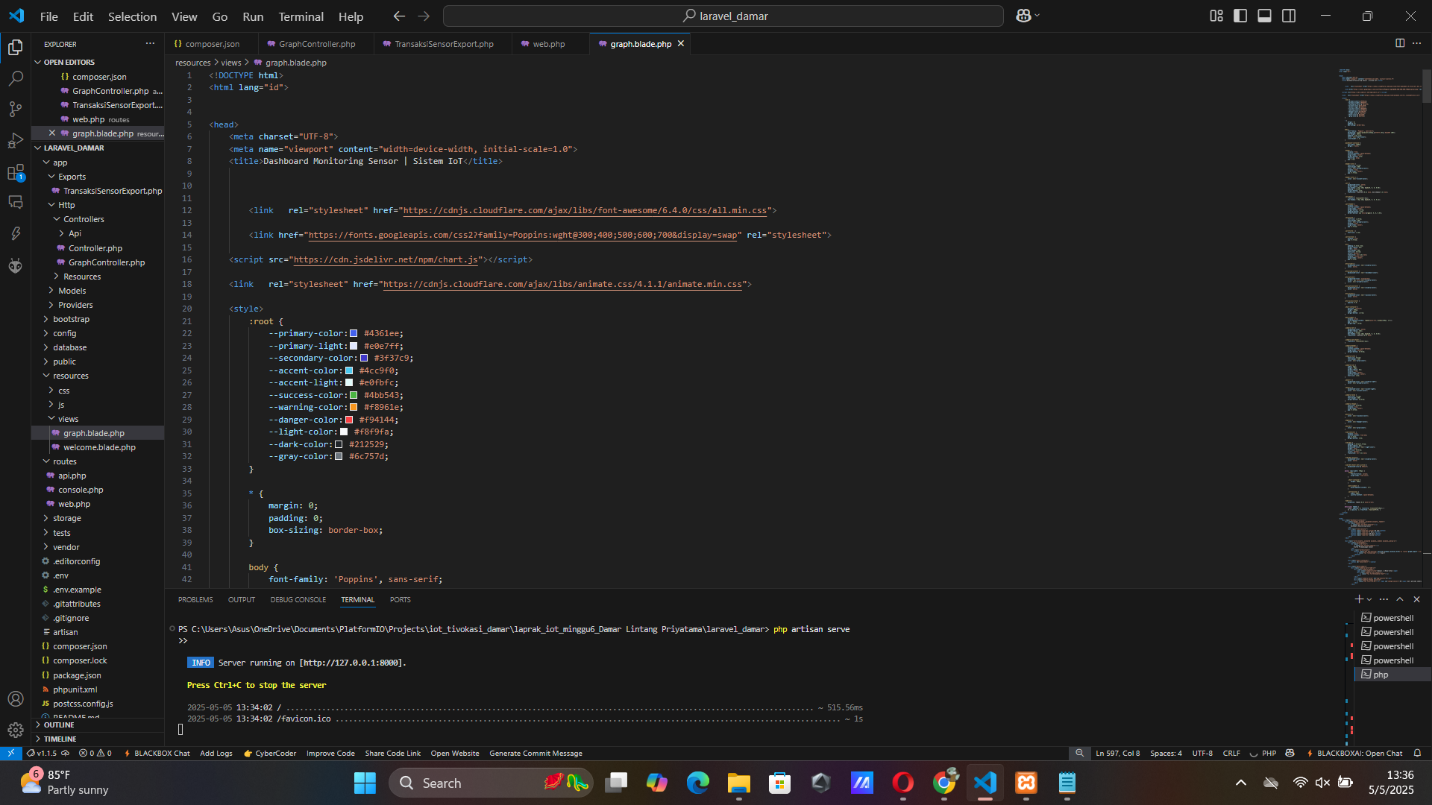
-Open the laravel folder that was created in Practice 12 on VSCode

-After that open the terminal and run the following code:

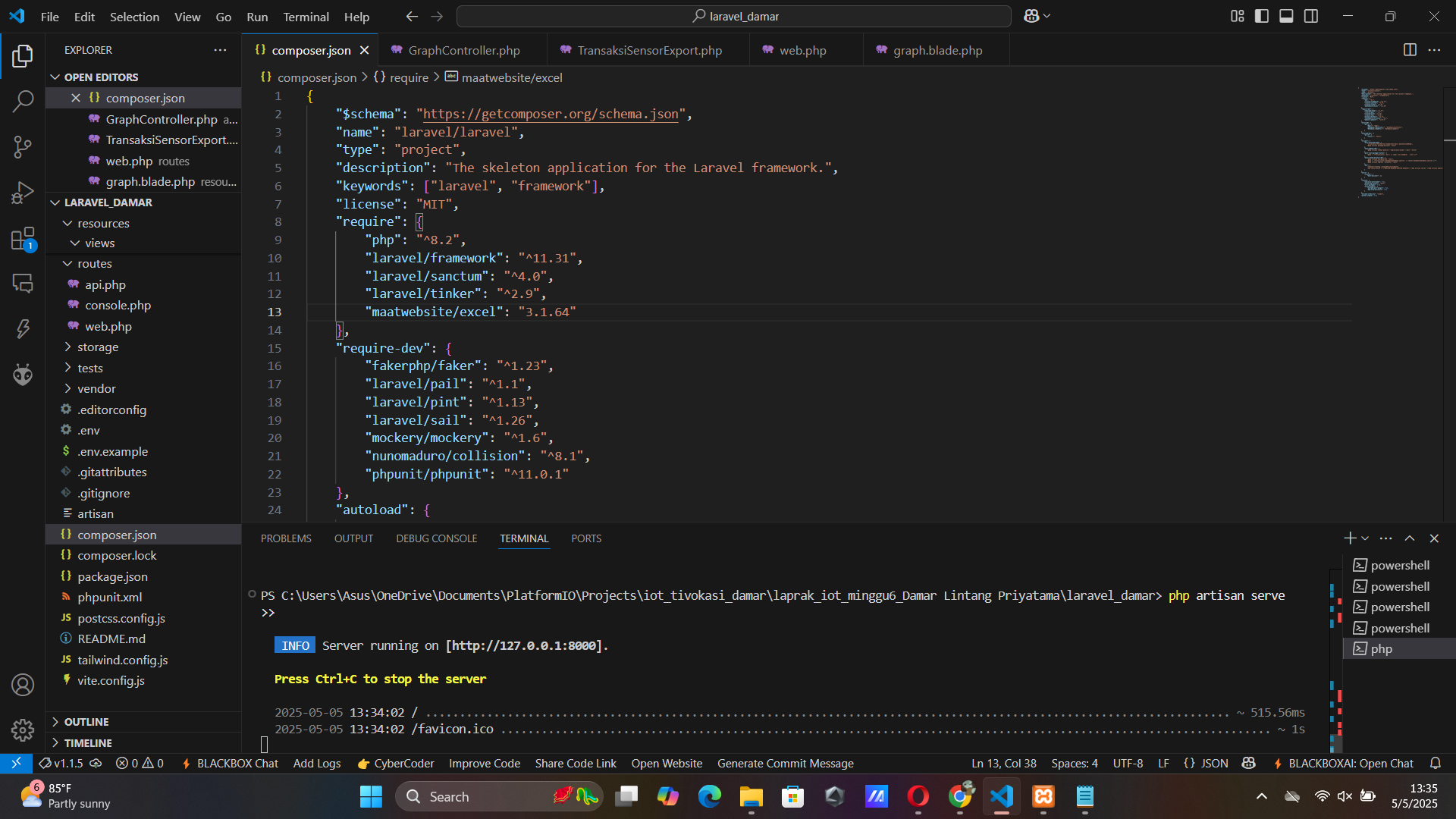
1. composer require maatwebsite/excel
2. php artisan make:controller GraphController

-After that, run this command on the terminal:

php artisan make:export TransaksiSensorExport --model=TransaksiSensor

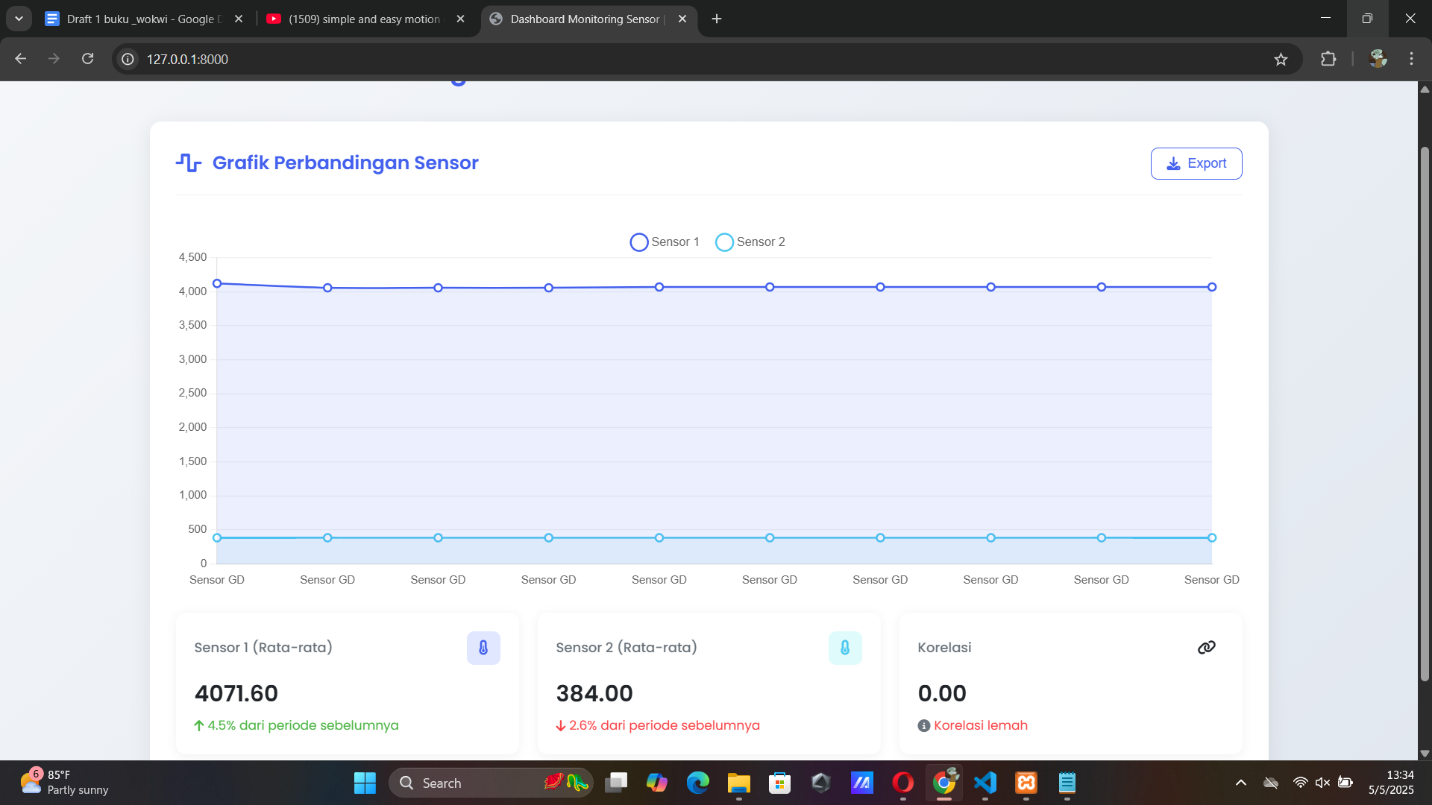
-After that, create a graph.blade.php file in the resources/views folder and add the code

-Once everything is done, run the program with the following command

php artisan serve



**3. Results and Discussion**

**3.1 Experimental Results**