**Membuat Tampilan InterfaceWeb Dashboard IoT**

*Pelangi Anggel*

Fakultas Vokasi , Universitas Brawijaya

Email: pelangianggel283@gmail.com

**Abstract**

This practice aims to understand the process of creating a web-based dashboard interface for IoT applications. The dashboard serves as a visual interface to display data from IoT devices, improving the user’s ability to monitor and interact with real-time information. The implementation involves using basic web development tools such as HTML, CSS, and JavaScript. The results demonstrate how IoT data can be effectively visualized through interactive charts and responsive design. This practice highlights the importance of user interfaces in modern IoT systems to ensure data readability and system usability.

*Keywords — Web Dashboard, IoT, Interface, HTML, Data Visualization*

**1. Introduction**

**1.1 Background**

A Web Dashboard Interface is an essential component in IoT systems, providing users with an interactive platform to monitor real-time data from sensors and devices. This interface is typically accessed through a browser and developed using web technologies such as HTML, CSS, and JavaScript. By creating a custom dashboard, developers can present complex data in a clear and user-friendly format. This practice is critical in fields such as smart homes, environmental monitoring, and industrial IoT.

**1.2 Objectives**

This experiment aims to introduce students to the basic process of designing and developing a simple web dashboard interface. The focus is on using standard web development tools to display sensor data and simulate an IoT monitoring environment.

**2. Methodology**

* 1. **Tools & Materials**
* **VS Code**
* **Internet Access**
* **Database php**
* **Browser**

**2.2 Implementation Steps**

**1. Create a Model and Export Class**

* Create a model to access sensor data from the database.
* Create an export class to enable data export in Excel format.

**2. Add Routes in the web.php File**

* Add a route to display the dashboard.
* Add a route to handle data export to Excel.

**3. Create a Controller**

* Create a controller to retrieve sensor data from the database.
* Pass the data to the view for visualization.
* Add a function to handle Excel export.

**4. Create the View (Dashboard Page)**

* Create a Blade file (graph.blade.php) to design the dashboard.
* Display the sensor data using a chart library (e.g., Chart.js).
* Include a button or link to export data to Excel.

**5. Run the Laravel Application**

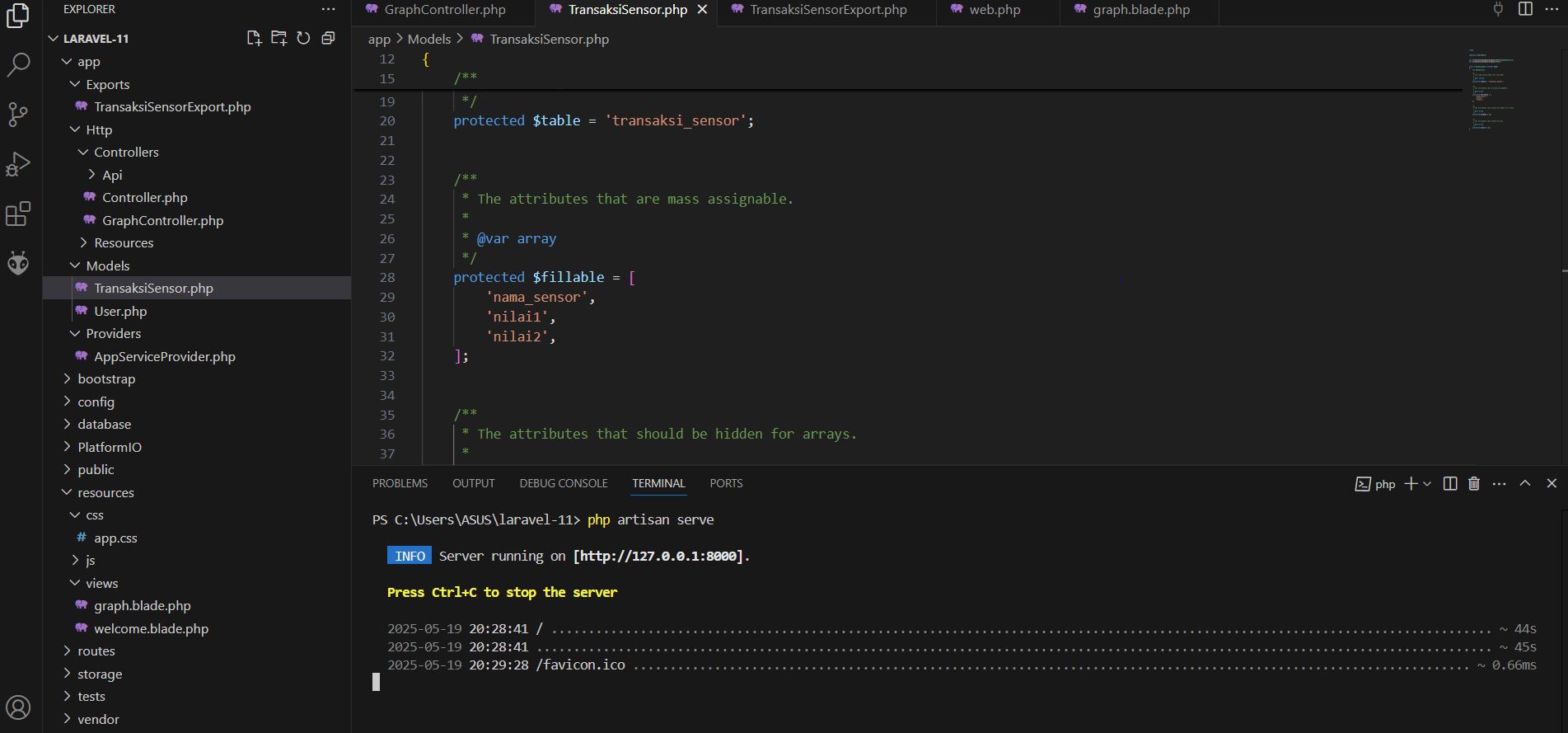
* Start the Laravel server using the php artisan serve command.
* Open the application in a browser to view the dashboard and export feature.

**3. Results and Discussion**

**3.1 Experimental Results**

|  |  |  |
| --- | --- | --- |
| **Platform** | **Completed Task** | **Key Outcome** |
| Laravel App | Displaying sensor data in chart | Graph displays real-time data from the iot\_25 database |
| Chart.js | Visualize data | Interactive line chart showing sensor value trends |

Artisan serve result:



Grafik sensor result:

