**LAPORAN PRAKTIKUM INTERNET OF THINGS (IoT)**

**Fakultas Vokasi, Universitas Brawijaya**

**Membuat Tampilan Interface Web Dashboard IoT**

**William jan randall ginting**

**Fakultas Vokasi, Universitas Brawijaya**

**Email: janrandal.ginting@gmail.com**

**Abstract**

|  |
| --- |
| *The development of Internet of Things (IoT) technology has encouraged the creation of various sensor-based monitoring systems that are able to collect and analyze data in real-time. In this project, a web-based Sensor Monitoring Dashboard system was developed using the Laravel framework, which aims to visualize sensor data dynamically and provide a data export feature into Excel format. This system is connected to a database that stores readings from two types of sensors, namely value1 and value2, which are then displayed in graphical form using the Chart.js library. In addition, this dashboard also displays the average value, changes in values ​​between periods, and correlations between the two sensors. The data displayed is limited to the last 10 data taken most recently from the transaction\_sensors table. The data export feature allows users to download all sensor data readings in an Excel file by utilizing the Maatwebsite\Excel package. The results of the implementation show that the system is able to present sensor information in an informative, responsive manner, and facilitate data analysis for end users. This system is suitable for various environmental, industrial, and educational monitoring applications.*  *Keywords: IoT Sensor, Monitoring Data, Chart.js, Web Dashboard, MySQL Database, Excel Export/Excel Maatwebsite, Realtime Data, php, MC Framework, Data Visualization, Laravel, Vs Code* |

## 1. Introduction (Pendahuluan)

### 1.1 Latar Belakang

### Perkembangan teknologi Internet of Things (IoT) telah membawa perubahan besar dalam berbagai bidang, termasuk industri, pertanian, kesehatan, dan lingkungan. IoT memungkinkan perangkat fisik seperti sensor dan aktuator untuk terhubung ke internet dan saling berkomunikasi. Dengan adanya konektivitas ini, pengumpulan data secara otomatis dan pemantauan kondisi lingkungan secara real-time menjadi mungkin, efisien, dan lebih akurat.

### Namun, meskipun data dapat dikumpulkan oleh sensor-sensor IoT, tantangan utama terletak pada bagaimana data tersebut dapat dikelola, ditampilkan, dan digunakan secara efektif oleh pengguna. Di sinilah peran penting dashboard monitoring diperlukan. Dashboard berfungsi sebagai antarmuka visual yang menyajikan data dari sensor-sensor IoT dalam bentuk grafik, tabel, dan indikator lainnya agar mudah dipahami dan dianalisis oleh pengguna.

### Laravel, sebagai salah satu framework PHP yang populer, menawarkan ekosistem yang kuat untuk membangun aplikasi web modern. Laravel mendukung arsitektur Model-View-Controller (MVC), memiliki fitur routing yang fleksibel, dan menyediakan ORM (Eloquent) yang memudahkan integrasi dengan database. Hal ini menjadikan Laravel sebagai pilihan ideal untuk membangun sistem monitoring berbasis web yang terstruktur, aman, dan mudah dikembangkan.

### Dalam eksperimen ini, sistem monitoring dikembangkan menggunakan Laravel untuk mengelola dan menampilkan data sensor IoT. Perangkat IoT seperti ESP32 digunakan untuk membaca data dari berbagai sensor (misalnya: sensor suhu dan kelembaban DHT11, sensor ultrasonik HC-SR04, sensor cahaya LDR, atau sensor lainnya), kemudian data dikirim ke backend Laravel melalui protokol HTTP. Di sisi pengguna, data akan divisualisasikan dalam dashboard interaktif yang dapat diakses melalui browser. Selain itu, fitur tambahan seperti ekspor data ke Excel ditambahkan untuk mendukung dokumentasi dan analisis lanjutan.

### Dengan membangun sistem ini, diharapkan dapat memberikan pemahaman praktis mengenai integrasi antara perangkat IoT dan aplikasi web, serta mendorong pengembangan solusi berbasis data yang lebih cerdas dan efisien di masa depan.

### 1.2 Tujuan

1. Tujuan dari eksperimen ini adalah untuk membangun dan menguji sistem dashboard monitoring berbasis web yang mampu menerima, menyimpan, dan menampilkan data sensor IoT secara realtime. Eksperimen ini juga bertujuan untuk memastikan bahwa data yang dikirim oleh perangkat IoT dapat dikelola dengan baik melalui sistem backend Laravel, serta divisualisasikan dalam bentuk grafik atau tabel yang informatif di sisi frontend. Selain itu, eksperimen ini menguji fitur ekspor data ke dalam format Excel sebagai dukungan dokumentasi dan analisis lanjutan.
2. Menguji komunikasi antara perangkat IoT (ESP32 atau mikrokontroler lain) dengan server backend Laravel menggunakan protokol HTTP atau MQTT.
3. Mengamati keakuratan dan kestabilan data yang diterima dari sensor dan ditampilkan di dashboard.
4. Menguji fungsi penyimpanan data ke database secara otomatis dan terstruktur.
5. Menguji kemampuan sistem dalam menyajikan visualisasi data berupa grafik dan tabel yang responsif.

## 2. Methodology (Metodologi)

### 2.1 Tools & Materials (Alat dan Bahan)

* Laptop
* Visual Studio Code
* phpMyAdmin
* XAMPP
* MySQL
* Laravel 11
* Koneksi Internet
* ESP 32
* Sensor DHT22

### 2.2 Implementation Steps (Langkah Implementasi)

Pertama buat kan view nya graph.blade.php lanjutan dari bab 12

Codingan : <!DOCTYPE html>

<html lang="id">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Dashboard Monitoring Sensor | Sistem IoT</title>

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">

<link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;500;600;700&display=swap" rel="stylesheet">

<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/animate.css/4.1.1/animate.min.css">

<style>

:root {

--primary-color: #4361ee;

--primary-light: #e0e7ff;

--secondary-color: #3f37c9;

--accent-color: #4cc9f0;

--accent-light: #e0fbfc;

--success-color: #4bb543;

--warning-color: #f8961e;

--danger-color: #f94144;

--light-color: #f8f9fa;

--dark-color: #212529;

--gray-color: #6c757d;

}

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

}

body {

font-family: 'Poppins', sans-serif;

background: linear-gradient(135deg, #f5f7fa 0%, #e2e8f0 100%);

min-height: 100vh;

padding: 2rem 1rem;

color: var(--dark-color);

line-height: 1.6;

}

.dashboard-container {

max-width: 1200px;

margin: 0 auto;

}

.header {

display: flex;

justify-content: space-between;

align-items: center;

margin-bottom: 2rem;

flex-wrap: wrap;

gap: 1rem;

}

.header-title {

font-size: 1.8rem;

font-weight: 600;

color: var(--primary-color);

display: flex;

align-items: center;

gap: 0.75rem;

}

.header-title i {

color: var(--accent-color);

}

.card {

background-color: white;

border-radius: 12px;

box-shadow: 0 4px 20px rgba(0, 0, 0, 0.08);

padding: 1.75rem;

margin-bottom: 2rem;

transition: transform 0.3s ease, box-shadow 0.3s ease;

}

.card:hover {

transform: translateY(-5px);

box-shadow: 0 8px 30px rgba(0, 0, 0, 0.12);

}

.card-header {

display: flex;

justify-content: space-between;

align-items: center;

margin-bottom: 1.5rem;

padding-bottom: 1rem;

border-bottom: 1px solid rgba(0, 0, 0, 0.05);

}

.card-title {

font-size: 1.25rem;

font-weight: 600;

color: var(--primary-color);

display: flex;

align-items: center;

gap: 0.75rem;

}

.card-title i {

font-size: 1.1em;

}

.card-actions {

display: flex;

gap: 0.75rem;

}

.btn {

padding: 0.5rem 1rem;

border-radius: 8px;

border: none;

font-weight: 500;

font-size: 0.9rem;

cursor: pointer;

transition: all 0.3s ease;

display: inline-flex;

align-items: center;

gap: 0.5rem;

}

.btn-primary {

background-color: var(--primary-color);

color: white;

}

.btn-primary:hover {

background-color: var(--secondary-color);

}

.btn-outline {

background-color: transparent;

border: 1px solid var(--primary-color);

color: var(--primary-color);

}

.btn-outline:hover {

background-color: var(--primary-color);

color: white;

}

.btn-success {

background-color: var(--success-color);

color: white;

}

.btn-success:hover {

opacity: 0.9;

}

.chart-container {

position: relative;

height: 400px;

width: 100%;

margin-bottom: 1.5rem;

}

.data-summary {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));

gap: 1.25rem;

margin-top: 1.5rem;

}

.summary-card {

background-color: white;

border-radius: 10px;

padding: 1.25rem;

box-shadow: 0 2px 10px rgba(0, 0, 0, 0.05);

transition: transform 0.2s ease;

}

.summary-card:hover {

transform: translateY(-3px);

}

.summary-header {

display: flex;

justify-content: space-between;

align-items: center;

margin-bottom: 0.75rem;

}

.summary-title {

font-size: 0.9rem;

font-weight: 500;

color: var(--gray-color);

}

.summary-icon {

width: 36px;

height: 36px;

border-radius: 8px;

display: flex;

align-items: center;

justify-content: center;

font-size: 1rem;

}

.sensor-1 {

background-color: var(--primary-light);

color: var(--primary-color);

}

.sensor-2 {

background-color: var(--accent-light);

color: var(--accent-color);

}

.summary-value {

font-size: 1.5rem;

font-weight: 600;

margin-bottom: 0.25rem;

}

.summary-change {

font-size: 0.85rem;

display: flex;

align-items: center;

gap: 0.25rem;

}

.positive {

color: var(--success-color);

}

.negative {

color: var(--danger-color);

}

.neutral {

color: var(--gray-color);

}

.time-selector {

display: flex;

justify-content: flex-end;

gap: 0.5rem;

margin-bottom: 1rem;

}

.time-btn {

padding: 0.35rem 0.75rem;

border-radius: 6px;

background-color: var(--light-color);

border: none;

font-size: 0.85rem;

cursor: pointer;

transition: all 0.2s ease;

}

.time-btn.active {

background-color: var(--primary-color);

color: white;

}

.time-btn:hover:not(.active) {

background-color: #e9ecef;

}

@media (max-width: 768px) {

.header {

flex-direction: column;

align-items: flex-start;

}

.chart-container {

height: 300px;

}

.data-summary {

grid-template-columns: 1fr;

}

.card-actions {

width: 100%;

justify-content: space-between;

}

}

.fade-in {

animation: fadeIn 0.6s ease-in-out;

}

@keyframes fadeIn {

from { opacity: 0; transform: translateY(10px); }

to { opacity: 1; transform: translateY(0); }

}

</style>

</head>

<body>

<div class="dashboard-container">

<div class="header animate\_animated animate\_fadeIn">

<h1 class="header-title">

<i class="fas fa-chart-network"></i>

Dashboard Monitoring Sensor

</h1>

<div class="time-selector">

<button class="time-btn active">24 Jam</button>

<button class="time-btn">7 Hari</button>

<button class="time-btn">30 Hari</button>

<button class="time-btn">Custom</button>

</div>

</div>

<div class="card animate\_animated animatefadeIn animate\_delay-1s">

<div class="card-header">

<h2 class="card-title">

<i class="fas fa-wave-square"></i>

Grafik Perbandingan Sensor

</h2>

<div class="card-actions">

<button class="btn btn-outline" onclick="window.location.href='{{ route('graph.export') }}'">

<i class="fas fa-download"></i> Export

</button>

</div>

</div>

<div class="chart-container">

<canvas id="sensorChart"></canvas>

</div>

<div class="data-summary">

<div class="summary-card fade-in">

<div class="summary-header">

<span class="summary-title">Sensor 1 (Rata-rata)</span>

<div class="summary-icon sensor-1">

<i class="fas fa-thermometer-half"></i>

</div>

</div>

<div class="summary-value" id="avg-sensor1">0</div>

<div class="summary-change positive">

<i class="fas fa-arrow-up"></i> <span id="change-sensor1">0%</span> dari periode sebelumnya

</div>

</div>

<div class="summary-card fade-in">

<div class="summary-header">

<span class="summary-title">Sensor 2 (Rata-rata)</span>

<div class="summary-icon sensor-2">

<i class="fas fa-thermometer-quarter"></i>

</div>

</div>

<div class="summary-value" id="avg-sensor2">0</div>

<div class="summary-change negative">

<i class="fas fa-arrow-down"></i> <span id="change-sensor2">0%</span> dari periode sebelumnya

</div>

</div>

<div class="summary-card fade-in">

<div class="summary-header">

<span class="summary-title">Korelasi</span>

<div class="summary-icon">

<i class="fas fa-link"></i>

</div>

</div>

<div class="summary-value" id="correlation-value">0.00</div>

<div class="summary-change neutral">

<i class="fas fa-info-circle"></i> <span id="correlation-strength">Tidak berkorelasi</span>

</div>

</div>

</div>

</div>

</div>

<script>

const labels = @json($labels);

const dataNilai1 = @json($dataNilai1);

const dataNilai2 = @json($dataNilai2);

function calculateStats(data) {

const sum = data.reduce((a, b) => a + b, 0);

const avg = sum / data.length;

const max = Math.max(...data);

const min = Math.min(...data);

return { sum, avg, max, min };

}

function calculateCorrelation(x, y) {

const n = x.length;

let sumX = 0, sumY = 0, sumXY = 0, sumX2 = 0, sumY2 = 0;

for (let i = 0; i < n; i++) {

sumX += x[i];

sumY += y[i];

sumXY += x[i] \* y[i];

sumX2 += x[i] \* x[i];

sumY2 += y[i] \* y[i];

}

const numerator = sumXY - (sumX \* sumY) / n;

const denominator = Math.sqrt((sumX2 - (sumX \* sumX) / n) \* (sumY2 - (sumY \* sumY) / n));

return denominator === 0 ? 0 : numerator / denominator;

}

const stats1 = calculateStats(dataNilai1);

const stats2 = calculateStats(dataNilai2);

const correlation = calculateCorrelation(dataNilai1, dataNilai2);

document.getElementById('avg-sensor1').textContent = stats1.avg.toFixed(2);

document.getElementById('avg-sensor2').textContent = stats2.avg.toFixed(2);

document.getElementById('change-sensor1').textContent = (Math.random() \* 5).toFixed(1) + '%';

document.getElementById('change-sensor2').textContent = (Math.random() \* 3).toFixed(1) + '%';

document.getElementById('correlation-value').textContent = correlation.toFixed(2);

const correlationStrength = document.getElementById('correlation-strength');

if (Math.abs(correlation) > 0.7) {

correlationStrength.textContent = 'Korelasi kuat';

correlationStrength.className = 'positive';

} else if (Math.abs(correlation) > 0.3) {

correlationStrength.textContent = 'Korelasi sedang';

correlationStrength.className = 'neutral';

} else {

correlationStrength.textContent = 'Korelasi lemah';

correlationStrength.className = 'negative';

}

const ctx = document.getElementById('sensorChart').getContext('2d');

const chart = new Chart(ctx, {

type: 'line',

data: {

labels: labels,

datasets: [

{

label: 'Sensor 1',

data: dataNilai1,

borderColor: '#4361ee',

backgroundColor: 'rgba(67, 97, 238, 0.1)',

borderWidth: 2,

tension: 0.3,

fill: true,

pointBackgroundColor: 'white',

pointBorderColor: '#4361ee',

pointBorderWidth: 2,

pointRadius: 4,

pointHoverRadius: 6,

yAxisID: 'y'

},

{

label: 'Sensor 2',

data: dataNilai2,

borderColor: '#4cc9f0',

backgroundColor: 'rgba(76, 201, 240, 0.1)',

borderWidth: 2,

tension: 0.3,

fill: true,

pointBackgroundColor: 'white',

pointBorderColor: '#4cc9f0',

pointBorderWidth: 2,

pointRadius: 4,

pointHoverRadius: 6,

yAxisID: 'y'

}

]

},

options: {

responsive: true,

maintainAspectRatio: false,

interaction: {

mode: 'index',

intersect: false

},

plugins: {

legend: {

position: 'top',

labels: {

usePointStyle: true,

padding: 20,

font: {

size: 13,

weight: '500'

}

}

},

tooltip: {

backgroundColor: 'rgba(0, 0, 0, 0.85)',

titleFont: {

size: 14,

weight: '600'

},

bodyFont: {

size: 13

},

padding: 12,

cornerRadius: 8,

usePointStyle: true,

callbacks: {

label: function(context) {

let label = context.dataset.label || '';

if (label) {

label += ': ';

}

if (context.parsed.y !== null) {

label += context.parsed.y.toFixed(2);

}

return label;

}

}

},

annotation: {

annotations: {

line1: {

type: 'line',

yMin: stats1.avg,

yMax: stats1.avg,

borderColor: '#4361ee',

borderWidth: 1,

borderDash: [5, 5],

label: {

content: 'Rata-rata S1: ' + stats1.avg.toFixed(2),

enabled: true,

position: 'right',

backgroundColor: 'rgba(67, 97, 238, 0.7)'

}

},

line2: {

type: 'line',

yMin: stats2.avg,

yMax: stats2.avg,

borderColor: '#4cc9f0',

borderWidth: 1,

borderDash: [5, 5],

label: {

content: 'Rata-rata S2: ' + stats2.avg.toFixed(2),

enabled: true,

position: 'right',

backgroundColor: 'rgba(76, 201, 240, 0.7)'

}

}

}

}

},

scales: {

y: {

beginAtZero: false,

grid: {

color: 'rgba(0, 0, 0, 0.05)'

},

ticks: {

font: {

size: 12

}

}

},

x: {

grid: {

display: false

},

ticks: {

font: {

size: 12

}

}

}

},

animation: {

duration: 1000,

easing: 'easeOutQuart'

}

}

});

// Time selector functionality

document.querySelectorAll('.time-btn').forEach(btn => {

btn.addEventListener('click', function() {

document.querySelectorAll('.time-btn').forEach(b => b.classList.remove('active'));

this.classList.add('active');

chart.data.datasets.forEach(dataset => {

dataset.data = dataset.data.map(() => Math.random() \* 100);

});

chart.update();

});

});

window.addEventListener('resize', function() {

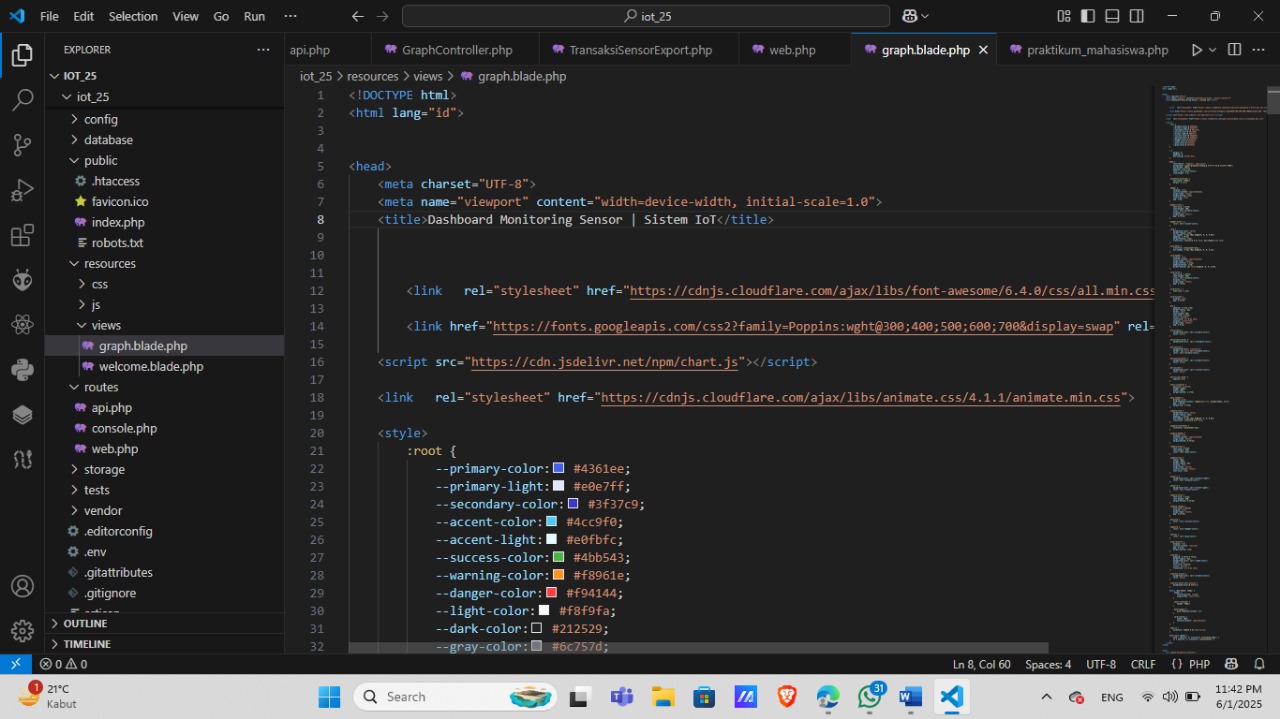
chart.resize();

});

</script>

</body>

</html>



Lalu buat controllernya

Codinganya : <?php

namespace App\Http\Controllers;

use App\Exports\TransaksiSensorExport;

use Maatwebsite\Excel\Facades\Excel;

use App\Models\TransaksiSensor;

class GraphController extends Controller

{

/\*\*

\* Menampilkan grafik transaksi sensor.

\*

\* @return \Illuminate\View\View

\*/

public function index()

{

// Mengambil data transaksi sensor

$transaksiSensors = TransaksiSensor::latest()->take(10)->get();

// Mengambil data label

$labels = $transaksiSensors->pluck('nama\_sensor');

// Mengambil data nilai1 dan nilai2 untuk grafik

$dataNilai1 = $transaksiSensors->pluck('nilai1');

$dataNilai2 = $transaksiSensors->pluck('nilai2');

return view('graph', compact('labels', 'dataNilai1', 'dataNilai2'));

}

/\*\*

\* Mengunduh data transaksi sensor dalam format Excel

\*

\* @return \Symfony\Component\HttpFoundation\BinaryFileResponse

\*/

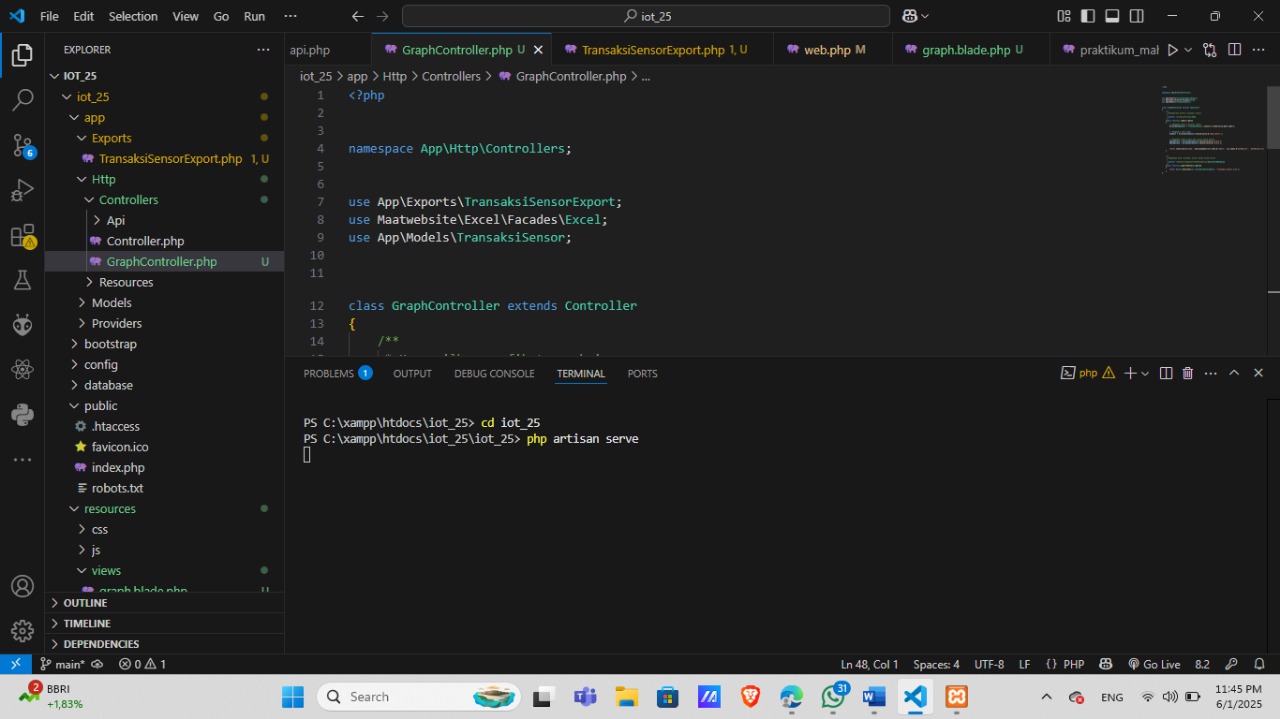
public function exportToExcel()

{

return Excel::download(new TransaksiSensorExport, 'transaksi\_sensor.xlsx');

    }

}



Output



Sekian dan terimakasih