

LAPORAN PRAKTIKUM INTERNET OF THINGS (IoT)

Fakultas Vokasi, Universitas Brawijaya

Membuat Tampilan Interface Web Dashboard IoT

Intan Tania

Fakultas Vokasi, Universitas Brawijaya

Email: taniaintan050@gmail.com

ABSTRAK

This practical project focuses on building a web dashboard interface to visualize real-time data collected from Internet of Things (IoT) devices. The Laravel framework is utilized to manage the backend operations and serve REST APIs that facilitate communication between IoT hardware and the web system. The user interface is constructed using HTML and CSS to create an informative and responsive layout. The results show that Laravel simplifies the integration process by offering a clean project structure and API support, enabling smooth data flow from sensors to the dashboard. This practicum enhances the understanding of IoT-to-web data integration and provides experience in developing interactive monitoring applications.

Keywords-- IoT, Web Dashboard, Laravel, Real-time Data, REST API, Sensor Monitoring, Data Visualization, Backend Integration, Frontend Interface.

1. Pendahuluan

1.1 Latar Belakang

Perkembangan teknologi Internet of Things (IoT) membawa kemampuan baru dalam menghubungkan berbagai perangkat elektronik agar bisa saling bertukar informasi melalui jaringan internet. Salah satu kebutuhan utama dalam penerapan IoT adalah kemampuan untuk mengawasi data sensor secara langsung. Oleh karena itu, dibutuhkan sebuah media visual yang dapat menampilkan informasi tersebut secara real-time dengan tampilan yang mudah dipahami pengguna.

Web dashboard menjadi pilihan yang tepat untuk menyajikan data dari sensor karena tampilannya yang fleksibel dan dapat menampilkan informasi dalam bentuk grafik, tabel, maupun indikator lainnya. Pada praktikum ini, web dashboard dibangun dengan framework Laravel yang terkenal memiliki struktur kerja yang rapi, mudah dikembangkan, dan mendukung pembuatan REST API untuk komunikasi data antara perangkat dan sistem.

Dengan membuat antarmuka web untuk dashboard IoT, peserta praktikum tidak hanya mempelajari pembuatan aplikasi web, tetapi juga memahami proses integrasi data sensor dari perangkat fisik ke dalam sistem berbasis web.

1.2 Tujuan

Adapun tujuan dari pelaksanaan praktikum ini antara lain:

1. Membuat antarmuka web untuk menampilkan informasi dari sensor secara langsung dan interaktif.
2. Menggunakan Laravel sebagai framework backend dalam mengelola data dan membangun REST API.
3. Menyambungkan perangkat IoT dengan sistem web agar data sensor dapat direkam dan divisualisasikan.
4. Memberikan pemahaman tentang mekanisme pertukaran data antara perangkat fisik dengan aplikasi berbasis web.
5. Melatih

2. Metodologi

2.1 Alat dan Bahan

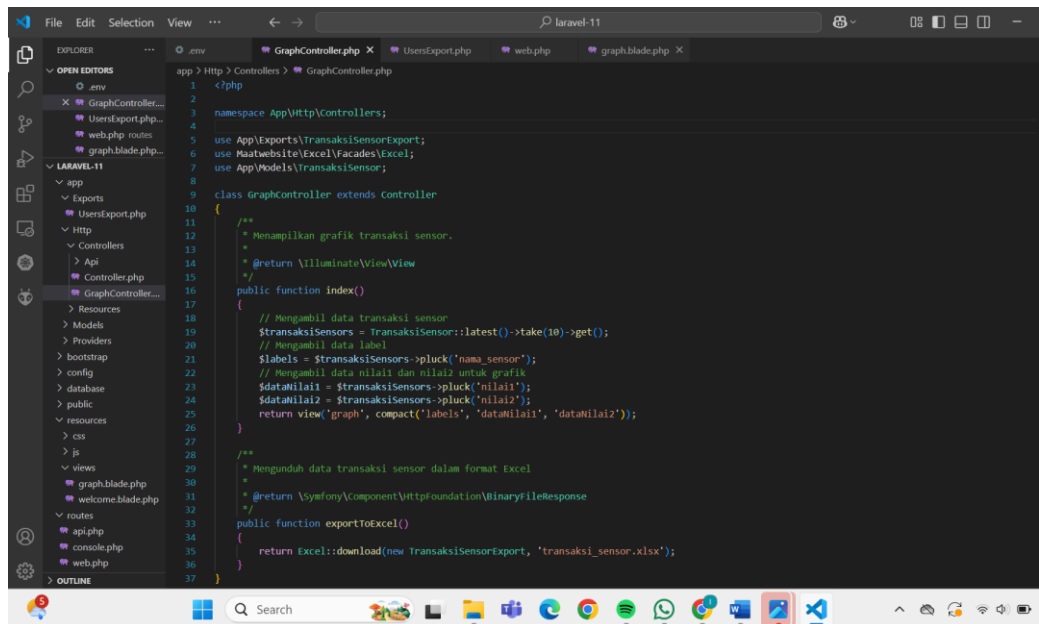
- Framework Laravel
- XAMPP
- Visual Studio Code
- Web Browser (Chrome)

2.2 Langkah Implementasi

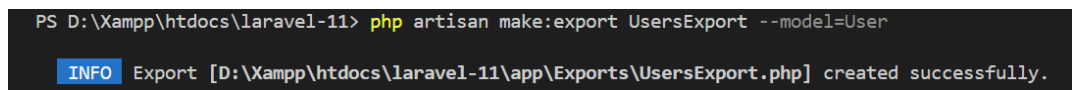
1. Buka folder proyek Laravel sebelumnya di Praktikum 12 yang bernama **Laravel-11** melalui Visual Studio Code.
2. Jalankan perintah composer require maatwebsite/excel dan php artisan make:controller GraphController untuk membuat controller grafik.

```
PS D:\Xampp\htdocs\laravel-11> php artisan make:controller GraphController  
  
INFO Controller [D:\Xampp\htdocs\laravel-11\app\Http\Controllers\GraphController.php] created successfully.
```

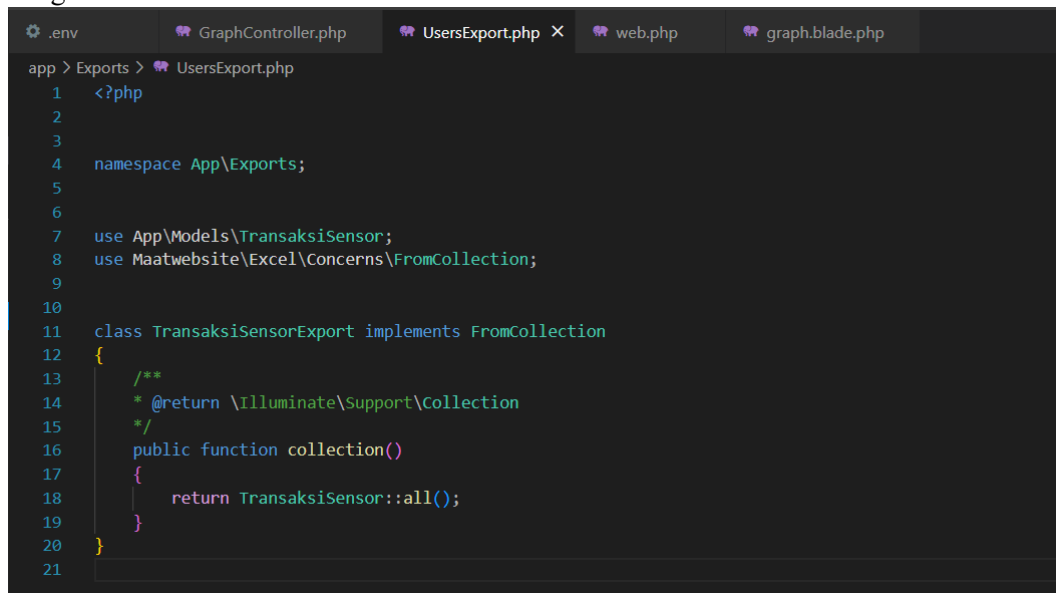
3. Tambahkan kode ke dalam GraphController.php.



4. Jalankan perintah php artisan make:export TransaksiSensorExport --model=TransaksiSensor untuk membuat file ekspor data.



5. Edit file TransaksiSensorExport.php agar dapat mengekspor data dari database dengan kode berikut



6. Tambahkan rute baru pada file web.php di folder routes untuk menampilkan grafik dengan kode berikut

```
.env  GraphController.php  UsersExport.php  web.php  graph.blade.php
routes > web.php
1  <?php
2
3
4  use Illuminate\Support\Facades\Route;
5  use App\Http\Controllers\GraphController;
6
7
8  Route::get('/', [GraphController::class, 'index'])->name('graph');
9  Route::get('/graph/export', [GraphController::class, 'exportToExcel'])->name('graph.export');
```

7. Buat file baru graph.blade.php di dalam folder resources/views, dan tambahkan kode untuk menampilkan data.

```
<!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/anima
te.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>

```

```

        <div class="card animate__animated animate__fadeIn
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

```



```

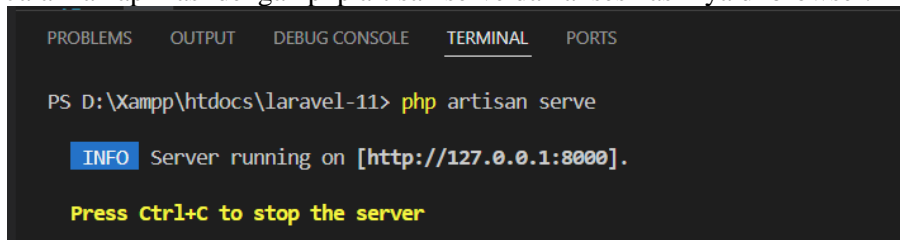
        document.querySelectorAll('.time-btn').forEach(btn => {
            btn.addEventListener('click', function() {
                document.querySelectorAll('.time-btn').forEach(b =>
b.classList.remove('active'));
                this.classList.add('active');

                chart.data.datasets.forEach(dataset => {
                    dataset.data = dataset.data.map(() =>
Math.random() * 100);
                });
                chart.update();
            });
        });

        window.addEventListener('resize', function() {
            chart.resize();
        });
    </script>
</body>
</html>

```

8. Jalankan aplikasi dengan php artisan serve dan akses hasilnya di browser.



```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS D:\Xampp\htdocs\laravel-11> php artisan serve

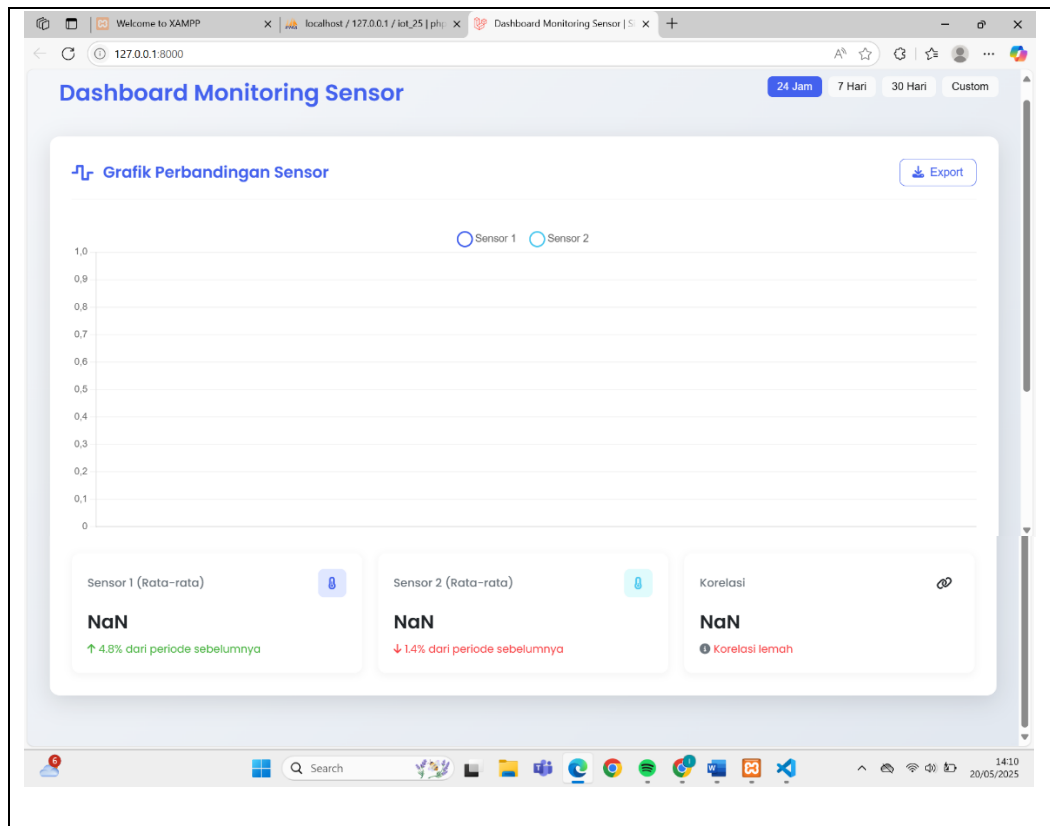
[INFO] Server running on [http://127.0.0.1:8000].

Press Ctrl+C to stop the server

```

3. Hasil dan Pembahasan

Web dashboard berhasil dibuat dan berfungsi untuk menampilkan data sensor suhu dan kelembaban dari perangkat IoT secara langsung. Data yang dikirim dari mikrokontroler ESP32 menggunakan HTTP diterima oleh backend Laravel, disimpan di database MySQL, lalu divisualisasikan melalui grafik pada halaman web.



Dari segi tampilan, dashboard menyajikan grafik yang diperbarui secara berkala, memungkinkan pengguna melihat perubahan data secara real-time. Laravel mempermudah pengembangan backend karena dukungan struktur MVC-nya yang jelas. REST API yang dibuat diuji menggunakan Postman dan memberikan respons yang akurat.

Praktikum ini membuktikan bahwa integrasi antara perangkat IoT dan web server menggunakan Laravel cukup efektif. Sistem ini bisa dikembangkan lebih lanjut untuk menambahkan fitur seperti kontrol perangkat jarak jauh atau notifikasi otomatis berdasarkan nilai sensor.