LAPORAN PRAKTIKUM
PERANCANGAN DAN PEMROGRAMAN WEB

MODUL 10

LARAVEL I

Oleh:

Dhiemas Tulus Ikhsan - 2311104046

PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK
DIREKTORAT KAMPUS PURWOKERTO
UNIVERSITAS TELKOM
2025

BAB I

PENDAHULUAN

1.1 Dasar Teori

Laravel adalah kerangka kerja (framework) berbasis PHP yang menggunakan arsitektur Model-View-Controller (MVC) untuk menciptakan pengembangan aplikasi web yang lebih sistematis, efisien, dan mudah dikelola. Sebagai sebuah framework, Laravel menawarkan sekumpulan pustaka, fungsi, dan kelas siap pakai yang memungkinkan pengembang membangun aplikasi tanpa harus menulis kode dari nol, sehingga proses kerja menjadi lebih cepat serta memiliki standar yang konsisten. Di dalamnya sudah tersedia berbagai fitur esensial seperti sistem routing, Blade Template Engine untuk tampilan, Eloquent ORM untuk manajemen database, hingga fitur keamanan dan validasi data. Melalui konsep MVC, Laravel memisahkan logika aplikasi ke dalam tiga bagian: Model untuk pengelolaan data, View untuk antarmuka pengguna, dan Controller sebagai pengatur alur logika serta penghubung keduanya. Pemisahan komponen ini bertujuan agar kode lebih terorganisir, mempermudah proses perbaikan (debugging), serta memperingkas pemeliharaan jangka panjang. Dengan alur kerja yang dimulai dari penerimaan request lewat routing hingga pemrosesan oleh controller, Laravel didukung oleh dokumentasi yang komprehensif dan komunitas yang luas, menjadikannya pilihan utama dalam pengembangan web dinamis saat ini.

1.2 Tujuan

1. Memahami konsep dasar framework web dan arsitektur MVC.
2. Memahami fungsi Laravel sebagai framework PHP modern.
3. Mampu melakukan instalasi Composer dan framework Laravel dengan benar.
4. Mampu menjalankan proyek Laravel pertama sebagai tahap awal pengembangan aplikasi web.

BAB II

HASIL PRAKTIKUM

2.1 GUIDED (Praktikum Terbimbing)

a. Route

Routing dalam Laravel berperan sebagai jembatan yang menghubungkan URL akses pengguna dengan logika pemrosesan di dalam aplikasi. Setiap permintaan HTTP yang masuk akan diarahkan terlebih dahulu melalui sistem route sebelum akhirnya diproses oleh controller atau langsung ditampilkan lewat view. Di Laravel, seluruh rute ini dikelola dan didefinisikan secara terpusat di dalam file-file yang berada di direktori routes/.

routes/web.php.

```
Route::get('/beranda', function () {
return "Halaman Beranda";
});
Route::get('/kendaraan/{jenis}', function ($jenis) {
return "Tampilkan data kendaraan dengan jenis $jenis";
});
Route::get('/produk', function () {
$arrProduk = [
"prod1" => "Televisi",
"prod2" => "Kipas Angin",
"prod3" => "Radio"
];
return view('produk', $arrProduk);
});
```

b. View

View adalah komponen MVC yang bertugas menyajikan tampilan atau antarmuka kepada pengguna. Dalam ekosistem Laravel, seluruh file view dikelola di dalam direktori resources/views dan memanfaatkan Blade Template Engine, sehingga setiap filenya menggunakan ekstensi khusus .blade.php.

```
Route::get('/mahasiswa', function () {
return view('mahasiswa');
});
resources\views\mahasiswa.blade.php
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Data Mahasiswa</title>
</head>
<body>
<h1>Data Mahasiswa</h1>
<ol>
@forelse ($mahasiswa as $val)
<li>{{ $val }}</li>
@empty
<div>Tidak ada data...</div>
@endforelse
</ol>
</body>
</html>
```

c. Blade

Blade merupakan mesin template bawaan Laravel yang dirancang untuk menyederhanakan penulisan kode PHP pada bagian tampilan (view). Dengan Blade, pengembang dapat menggunakan sintaks yang lebih ringkas dan bersih untuk menyajikan data, menjalankan perulangan (looping), serta mengatur logika pengkondisian.

```
Route::get('/produk', function () {
$produk = ['Televisi', 'Radio', 'Kipas Angin'];
return view('produk', ['produk' => $produk]);
});
resources\views\produk.blade.php
```

```
@extends('master')

@section('title','Data Produk')

@section('content')

Produk 1 : {{ $prod1; }}

<br>

Produk 2 : {{ $prod2; }}

<br>

Produk 2 : {{ $prod3; }}

@endsection
```

d. Controller

Controller berperan sebagai pengatur logika utama aplikasi sekaligus jembatan yang menghubungkan antara route dan view. Penggunaan controller sangat penting agar struktur kode tetap rapi, terorganisir, dan mudah dikelola secara berkelanjutan. Untuk membuat file controller secara otomatis, kamu cukup menjalankan perintah php artisan make:controller NamaController melalui terminal.

app\Http\Controllers\PageController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class PageController extends Controller

{

public function index()

{

return view('welcome');

}

public function tampil()

{

$arrMahasiswa = [

"Kholifahdina",

"Rahmat Taufik",

"Nita Fitrotunimah",
```

```
"Defrin Anggun Saputri"
];
return view('mahasiswa')->with('mahasiswa', $arrMahasiswa);
}
}
routes\web.php
use App\Http\Controllers\PageController;
Route::get('/', [PageController::class, 'index']);
Route::get('/mahasiswa', [PageController::class, 'mahasiswa']);
```

2.2 UNGUIDED (Tugas Mandiri)

a. TUGAS - Router

i. Buat minimal 5 route tanpa parameter

```
Route::get('/login', function () {
    return view('auth.login');
})->name('login');
```

```
Route::get('/dashboard',
[GuruDashboardController::class,
'index'])->name('dashboard');
```

```
Route::get('/data-pengguna',
[DataPenggunaController::class,
'index'])->name('data_pengguna');
```

```
oute::get('/data-presensi',
    [PresensiController::class, 'index']
    )->name('data_presensi');
```

```
Route::get('/riwayat-notifikasi', function () {
        return view('admin.riwayat_notifikasi');
    })->name('riwayat_notifikasi');
```

ii. Buat minimal 3 route dengan parameter

```
Route::get('/data-pengguna/guru/edit/{id}',
[GuruController::class, 'edit'])->name('guru.edit');
```

```
Route::get('/data-pengguna/guru/detail/{id}',
        [GuruController::class, 'show']
    )->name('guru.show');
```

```
Route::get('/profil-siswa/detail/{nis}',
[GuruProfilSiswaController::class,
'getDetail'])->name('profil_siswa.detail');
```

iii. Buat minimal 3 route dengan optional parameter

```
Route::get('/siswa/detail/{id?}',
[SiswaController::class, 'show']);
```

```
Route::get('/laporan/{bulan?}',
[LaporanController::class, 'index']);
```

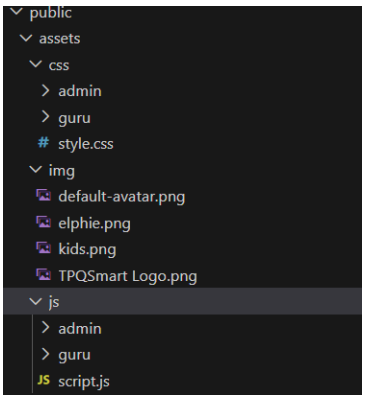
```
Route::get('/arsip/{kategori?}',
[PostController::class, 'archive']);
```

Route yang dibuat nantinya akan diimplementasikan pada tugas besar di mata kuliah teori.

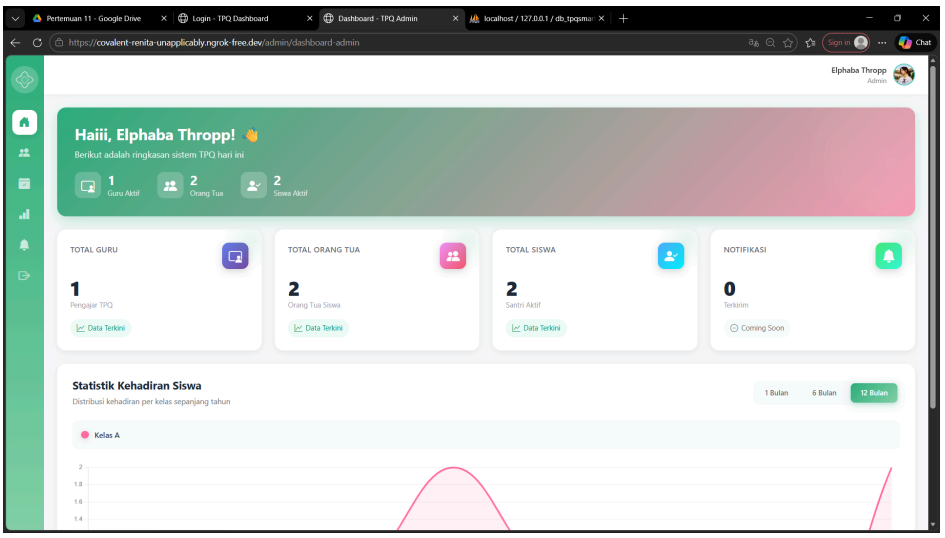
b. TUGAS - View

Pada pembelajaran PHP dasar kita pasti sudah mempelajari pengelolaan asset. Pada tugas, kali ini terapkan pengelolaan asset pada framework Laravel.

i. Buat folder asset yang didalamnya birisi folder css, js, img.



ii. Buat view yang menampilkan gambar yang terletak di folder img



iii. Buat view yang mengakses css dan javascript dari folder css dan js.

```
$(document).ready(function () {
    // 1. Sidebar Toggle
    $(document).on("click", ".hamburger-btn", function
() {
        const sidebar = $(".sidebar");
        sidebar.toggleClass("expanded");
                                const    isExpanded    =
sidebar.hasClass("expanded");
               localStorage.setItem("sidebarExpanded",
isExpanded);
    });

    // Load Sidebar State
      if (localStorage.getItem("sidebarExpanded") ===
"true") {
        $(".sidebar").addClass("expanded");
    }

    // 2. Mobile Menu Toggle
    $("#btnMenu").on("click", function (e) {
        e.stopPropagation();
        $("#sidebar").toggleClass("show");
    });

    // 3. Click Outside Sidebar
    $(document).on("click", function (e) {
         if (!$(e.target).closest(".sidebar, #btnMenu,
.hamburger-btn").length) {
            $("#sidebar").removeClass("show");
```

```
        }
    });

    // 4. Navigation Handler (untuk link biasa, bukan
logout)
    $(".nav-item")
        .not("#btnLogout")
        .on("click", function (e) {
            const href = $(this).attr("href");

            if (!href || href === "#") {
                e.preventDefault();
                    console.warn("Navigation item has
invalid href:", href);
            }
               // Biarkan browser navigate untuk href
yang valid
        });

    // 5. Logout Handler
    $("#btnLogout").on("click", function (e) {
        e.preventDefault();
        e.stopPropagation();
        $("#logoutOverlay").addClass("show");
    });

    // 6. Cancel Logout
    $("#cancelLogout").on("click", function (e) {
        e.preventDefault();
        $("#logoutOverlay").removeClass("show");
    });

    // 7. Initialize Chart
    renderPerformanceChart();

    // 8. Animate Stats Cards
    animateStats();

    // 9. Update Time (jika ada element #currentTime)
    if ($("#currentTime").length) {
        updateCurrentTime();
        setInterval(updateCurrentTime, 60000);
    }
```

```
    // 10. Chart Filter Buttons
    $(".btn-filter").on("click", function () {
        $(".btn-filter").removeClass("active");
        $(this).addClass("active");

        const range = $(this).data("range");
        updateChartLabels(range);
    });
});

// ========== CHART FUNCTIONS ==========
function renderPerformanceChart() {
                                const        ctx        =
document.getElementById("performanceChart");
    if (!ctx) {
        console.warn("Chart canvas not found");
        return;
    }

      // Ambil data dari window.dashboardData (diset
dari Blade)
     const hasData = window.dashboardData?.hasData ||
false;
     const labels = window.dashboardData?.chartLabels
|| [];
                             const       datasets       =
window.dashboardData?.chartDatasets || [];

       console.log("Chart Data:", { hasData, labels,
datasets });

    // Jika tidak ada data, tampilkan chart kosong
    if (!hasData || datasets.length === 0) {
        renderEmptyChart(ctx, labels);
        return;
    }

    // Transform datasets untuk Chart.js
     const colors = ["#FF6B9D", "#9D6BFF", "#6B9DFF",
"#FFB84D", "#4DFFB8"];
        const  chartDatasets  =  datasets.map((dataset,
index) => ({
```

```
        label: dataset.label,
        data: dataset.data,
        borderColor: colors[index % colors.length],
            backgroundColor: hexToRgba(colors[index %
colors.length], 0.1),
        borderWidth: 3,
        tension: 0.4,
        fill: true,
        pointRadius: 0,
        pointHoverRadius: 6,
    }));

    const config = {
        type: "line",
        data: {
            labels: labels,
            datasets: chartDatasets,
        },
        options: getChartOptions(),
    };

    window.performanceChart = new Chart(ctx, config);
}

function renderEmptyChart(ctx, labels) {
    const config = {
        type: "line",
        data: {
            labels:
                labels.length > 0
                    ? labels
                    : [
                          "JAN",
                          "FEB",
                          "MAR",
                          "APR",
                          "MAY",
                          "JUN",
                          "JUL",
                          "AUG",
                          "SEP",
                          "OCT",
                          "NOV",
```

```
                        "DEC",
                    ],
            datasets: [],
        },
        options: getChartOptions(),
    };

    window.performanceChart = new Chart(ctx, config);
    console.log("Empty chart rendered");
}

function getChartOptions() {
    return {
        responsive: true,
        maintainAspectRatio: false,
        interaction: {
            mode: "index",
            intersect: false,
        },
        plugins: {
            legend: {
                display: false,
            },
            tooltip: {
                enabled: true,
                backgroundColor: "rgba(0, 0, 0, 0.8)",
                padding: 12,
                  titleFont: { size: 13, weight: "bold"
},
                bodyFont: { size: 12 },
                callbacks: {
                    label: function (context) {
                                            let label =
context.dataset.label || "";
                        if (label) label += ": ";
                        if (context.parsed.y !== null)
{
                                            label += new
Intl.NumberFormat("id-ID").format(
                                    context.parsed.y
                            );
                        }
                        return label;
```

```
                    },
                },
            },
        },
        scales: {
            x: {
                grid: { display: false },
                ticks: {
                    font: { size: 11 },
                    color: "#888",
                },
            },
            y: {
                beginAtZero: true,
                grid: {
                    color: "rgba(0, 0, 0, 0.05)",
                    drawBorder: false,
                },
                ticks: {
                    font: { size: 11 },
                    color: "#888",
                    callback: function (value) {
                        return value >= 1000 ? value /
1000 + "k" : value;
                    },
                },
            },
        },
    };
}

function updateChartLabels(range) {
    if (!window.performanceChart) {
        console.warn("Chart not initialized");
        return;
    }

    let labels;
    if (range == 1) {
         labels = ["Week 1", "Week 2", "Week 3", "Week
4"];
    } else if (range == 6) {
```

```
          labels = ["JAN", "FEB", "MAR", "APR", "MAY",
"JUN"];
    } else {
        labels = [
            "JAN",
            "FEB",
            "MAR",
            "APR",
            "MAY",
            "JUN",
            "JUL",
            "AUG",
            "SEP",
            "OCT",
            "NOV",
            "DEC",
        ];
    }

    window.performanceChart.data.labels = labels;
    window.performanceChart.update();

    console.log("Chart labels updated:", labels);
}

// ========== UTILITY FUNCTIONS ==========
function animateStats() {
    $(".stat-card").each(function (index) {
        $(this).css({
            opacity: "0",
            transform: "translateY(20px)",
        });

        setTimeout(() => {
            $(this).css({
                opacity: "1",
                transform: "translateY(0)",
                transition: "all 0.5s ease",
            });
        }, index * 100);
    });
}
```

```
function updateCurrentTime() {
    const now = new Date();
       const options = { hour: "2-digit", minute:
"2-digit", hour12: false };
    const timeString = now.toLocaleTimeString("id-ID",
options);
    $("#currentTime").text(timeString);
}

function hexToRgba(hex, alpha) {
    const r = parseInt(hex.slice(1, 3), 16);
    const g = parseInt(hex.slice(3, 5), 16);
    const b = parseInt(hex.slice(5, 7), 16);
    return `rgba(${r}, ${g}, ${b}, ${alpha})`;
}

function showComingSoonAlert(featureName) {
    const alertHtml = `
      <div class="alert alert-info alert-dismissible
fade show position-fixed"
         style="top: 80px; right: 20px; z-index: 9999;
min-width: 300px;"
         role="alert">
      <i class="bi bi-info-circle-fill me-2"></i>
      <strong>${featureName}</strong> segera hadir!
            <button type="button" class="btn-close"
data-bs-dismiss="alert"></button>
    </div>`;

    $("body").append(alertHtml);

    setTimeout(() => {
        $(".alert").fadeOut(function () {
            $(this).remove();
        });
    }, 3000);
}

// Log untuk debugging
console.log("Dashboard Admin JS loaded successfully");
```

```css
/* ========== DASHBOARD ADMIN STYLING - COMPLETE VERSION ========== */
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

body {
     font-family: "Segoe UI", Tahoma, Geneva, Verdana, sans-serif;
    background: #f8f9fa;
    overflow-x: hidden;
}

/* ========== SIDEBAR ========== */
.sidebar {
    position: fixed;
    left: 0;
    top: 0;
    width: 70px;
    height: 100vh;
        background: linear-gradient(180deg, #2eaf7d, #83d1a7);
    display: flex;
    flex-direction: column;
    align-items: center;
    padding: 20px 0;
    z-index: 1000;
    transition: width 0.3s ease;
}

.sidebar:hover {
    width: 220px;
}

.logo-section {
    width: 50px;
    height: 50px;
    background: rgba(255, 255, 255, 0.2);
    border-radius: 50%;
    display: flex;
    align-items: center;
```

```
    justify-content: center;
    margin-bottom: 30px;
    overflow: hidden;
    padding: 8px;
}

.sidebar-logo {
    width: 100%;
    height: 100%;
    object-fit: contain;
}

.nav-menu {
    display: flex;
    flex-direction: column;
    gap: 10px;
    width: 100%;
    padding: 0 12px;
}

.nav-item {
    width: 46px;
    height: 46px;
    display: flex;
    align-items: center;
    gap: 12px;
    color: rgba(255, 255, 255, 0.7);
    text-decoration: none;
    border-radius: 12px;
    transition: all 0.3s ease;
    font-size: 20px;
    padding: 0 12px;
    overflow: hidden;
    white-space: nowrap;
    position: relative;
}

.sidebar:hover .nav-item {
    width: 196px;
    justify-content: flex-start;
}

.nav-text {
```

```
    font-size: 14px;
    font-weight: 500;
    opacity: 0;
    transition: opacity 0.3s ease;
}

.sidebar:hover .nav-text {
    opacity: 1;
}

.nav-item:hover {
    background: rgba(255, 255, 255, 0.1);
    color: white;
}

.nav-item.active {
    background: white;
    color: #2eaf7d;
    box-shadow: 0 4px 12px rgba(0, 0, 0, 0.1);
}

/* Ripple Effect */
.nav-item .ripple {
    position: absolute;
    border-radius: 50%;
    background: rgba(255, 255, 255, 0.4);
    width: 20px;
    height: 20px;
    animation: ripple-animation 0.6s ease-out;
    pointer-events: none;
}

@keyframes ripple-animation {
    to {
        transform: scale(4);
        opacity: 0;
    }
}

/* ========== MAIN CONTENT ========== */
.main-content {
    margin-left: 70px;
    min-height: 100vh;
```

```
    transition: margin-left 0.3s ease;
    width: calc(100% - 70px);
}

/* ========== HEADER ========== */
.top-header {
    background: white;
    padding: 16px 24px;
    display: flex;
    justify-content: space-between;
    align-items: center;
    box-shadow: 0 2px 8px rgba(0, 0, 0, 0.05);
    position: sticky;
    top: 0;
    z-index: 999;
}

.btn-menu {
    background: transparent;
    border: none;
    font-size: 24px;
    cursor: pointer;
    color: #333;
    padding: 8px;
    display: none;
}

.user-profile {
    display: flex;
    align-items: center;
    gap: 12px;
    margin-left: auto;
}

.user-info {
    display: flex;
    flex-direction: column;
    align-items: flex-end;
}

.user-name {
    font-weight: 600;
    font-size: 14px;
```

```
    color: #333;
}

.user-role {
    font-size: 12px;
    color: #888;
}

.user-avatar {
    width: 40px;
    height: 40px;
    border-radius: 50%;
    border: 2px solid #4ecdc4;
    object-fit: cover;
}

/* ========== STAT CARDS ========== */
.stat-card {
    background: white;
    border-radius: 16px;
    padding: 24px;
    box-shadow: 0 4px 15px rgba(0, 0, 0, 0.08);
    transition: all 0.3s ease;
    position: relative;
    overflow: hidden;
}

.stat-card::before {
    content: "";
    position: absolute;
    top: 0;
    right: 0;
    width: 100px;
    height: 100px;
    background: radial-gradient(
        circle,
        rgba(46, 175, 125, 0.1) 0%,
        transparent 70%
    );
    border-radius: 50%;
    transform: translate(30%, -30%);
}
```

```
.stat-card:hover {
    transform: translateY(-8px);
    box-shadow: 0 12px 30px rgba(0, 0, 0, 0.15);
}

.stat-header {
    display: flex;
    justify-content: space-between;
    align-items: flex-start;
    margin-bottom: 16px;
}

.stat-label {
    font-size: 14px;
    color: #6b7280;
    font-weight: 600;
    text-transform: uppercase;
    letter-spacing: 0.5px;
}

.stat-icon {
    width: 48px;
    height: 48px;
    border-radius: 12px;
    display: flex;
    align-items: center;
    justify-content: center;
    color: white;
    font-size: 24px;
    box-shadow: 0 4px 12px rgba(0, 0, 0, 0.15);
}

.stat-icon.bg-guru {
      background: linear-gradient(135deg, #667eea 0%,
#764ba2 100%);
}

.stat-icon.bg-wali {
      background: linear-gradient(135deg, #f093fb 0%,
#f5576c 100%);
}

.stat-icon.bg-siswa {
```

```
      background: linear-gradient(135deg, #4facfe 0%,
#00f2fe 100%);
}

.stat-icon.bg-notif {
      background: linear-gradient(135deg, #43e97b 0%,
#38f9d7 100%);
}

.stat-value-container {
    margin: 12px 0;
}

.stat-value {
    font-size: 36px;
    font-weight: 800;
    color: #1f2937;
    margin: 0;
    line-height: 1;
}

.stat-subtitle {
    font-size: 13px;
    color: #9ca3af;
    margin-top: 4px;
}

.stat-progress {
    margin-top: 16px;
}

.progress {
    height: 6px;
    border-radius: 10px;
    background: #f3f4f6;
}

.progress-bar {
    border-radius: 10px;
    transition: width 1s ease;
}

.progress-bar.bg-guru {
```

```
        background: linear-gradient(90deg, #667eea 0%,
#764ba2 100%);
}

.progress-bar.bg-wali {
        background: linear-gradient(90deg, #f093fb 0%,
#f5576c 100%);
}

.progress-bar.bg-siswa {
        background: linear-gradient(90deg, #4facfe 0%,
#00f2fe 100%);
}

.stat-trend {
    display: flex;
    align-items: center;
    gap: 6px;
    font-size: 13px;
    margin-top: 12px;
    padding: 6px 12px;
    background: #f0fdf4;
    border-radius: 20px;
    width: fit-content;
}

.stat-trend.up {
    color: #059669;
    background: #ecfdf5;
}

.stat-trend.neutral {
    color: #6b7280;
    background: #f9fafb;
}

.stat-trend i {
    font-size: 14px;
}

/* Enhanced Chart Card */

.chart-card {
```

```
    background: white;
    border-radius: 16px;
    padding: 28px;
    width: 100%;
    box-shadow: 0 4px 15px rgba(0, 0, 0, 0.08);
    position: relative;
}
.chart-container canvas {
    width: 100% !important; /* Paksa biar full */
}

.chart-header {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin-bottom: 24px;
    flex-wrap: wrap;
    gap: 16px;
}

.chart-title-section {
    flex: 1;
}

.chart-title {
    font-size: 20px;
    font-weight: 700;
    color: #1f2937;
    margin-bottom: 6px;
}

.chart-subtitle {
    font-size: 14px;
    color: #6b7280;
    margin: 0;
}

.chart-controls {
    display: flex;
    gap: 8px;
    background: #f9fafb;
    padding: 4px;
    border-radius: 12px;
```

```
}

.chart-controls .btn {
    padding: 8px 18px;
    border-radius: 8px;
    font-size: 13px;
    font-weight: 600;
    transition: all 0.3s ease;
    border: none;
    background: transparent;
    color: #6b7280;
}

.chart-controls .btn.active {
      background: linear-gradient(135deg, #2eaf7d 0%,
#83d1a7 100%);
    color: white;
    box-shadow: 0 4px 12px rgba(46, 175, 125, 0.3);
}

.chart-legend {
    display: flex;
    gap: 24px;
    margin-bottom: 24px;
    flex-wrap: wrap;
    padding: 16px;
    background: #f9fafb;
    border-radius: 12px;
}

.legend-item {
    display: flex;
    align-items: center;
    gap: 10px;
    font-size: 14px;
    color: #374151;
    font-weight: 500;
}

.legend-dot {
    width: 14px;
    height: 14px;
    border-radius: 50%;
```

```
    display: inline-block;
    box-shadow: 0 2px 8px rgba(0, 0, 0, 0.15);
}

/* Welcome Banner */
.welcome-banner {
      background: linear-gradient(135deg, #2eaf7d 0%,
#f799b2 100%);
    border-radius: 16px;
    padding: 32px;
    margin-bottom: 24px;
    color: white;
    box-shadow: 0 8px 25px rgba(46, 175, 125, 0.3);
    position: relative;
    overflow: hidden;
}

.welcome-banner::before {
    content: "";
    position: absolute;
    top: -50%;
    right: -10%;
    width: 400px;
    height: 400px;
    background: radial-gradient(
        circle,
        rgba(255, 255, 255, 0.1) 0%,
        transparent 70%
    );
    border-radius: 50%;
}

.welcome-content {
    position: relative;
    z-index: 1;
}

.welcome-title {
    font-size: 28px;
    font-weight: 700;
    margin-bottom: 8px;
}
```

```css
.welcome-subtitle {
    font-size: 16px;
    opacity: 0.9;
    margin-bottom: 20px;
}

.welcome-stats {
    display: flex;
    gap: 32px;
    flex-wrap: wrap;
}

.welcome-stat-item {
    display: flex;
    align-items: center;
    gap: 12px;
}

.welcome-stat-icon {
    width: 48px;
    height: 48px;
    background: rgba(255, 255, 255, 0.2);
    border-radius: 12px;
    display: flex;
    align-items: center;
    justify-content: center;
    font-size: 24px;
}

.welcome-stat-info h4 {
    font-size: 24px;
    font-weight: 700;
    margin: 0;
}

.welcome-stat-info p {
    font-size: 13px;
    margin: 0;
    opacity: 0.8;
}

/* Empty State */
.empty-chart-message {
```

```
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-50%, -50%);
    text-align: center;
    z-index: 10;
    pointer-events: none;
}

.empty-chart-message i {
    animation: pulse 2s ease-in-out infinite;
    color: #e5e7eb;
}

.empty-chart-message p {
    color: #9ca3af;
    margin-top: 16px;
    font-weight: 500;
}

@keyframes pulse {
    0%,
    100% {
        opacity: 0.4;
        transform: scale(1);
    }
    50% {
        opacity: 0.8;
        transform: scale(1.05);
    }
}

/* Animations */
.stat-card,
.chart-card {
    animation: fadeInUp 0.6s ease;
}

@keyframes fadeInUp {
    from {
        opacity: 0;
        transform: translateY(30px);
    }
```

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

/* Responsive */
@media (max-width: 768px) {
    .welcome-banner {
        padding: 24px;
    }

    .welcome-title {
        font-size: 22px;
    }

    .welcome-stats {
        gap: 20px;
    }

    .stat-value {
        font-size: 28px;
    }

    .chart-header {
        flex-direction: column;
        align-items: flex-start;
    }
}

/* Responsive Chart */
@media (max-width: 768px) {
    .chart-container {
        height: 300px;
    }
}
/* ========== LOGOUT MODAL ========== */
.logout-overlay {
    position: fixed;
    inset: 0;
    background: rgba(0, 0, 0, 0.4);
    display: flex;
    align-items: center;
```

```
    justify-content: center;
    z-index: 9999;
    visibility: hidden;
    opacity: 0;
    transition: all 0.3s ease;
}

.logout-overlay.show {
    visibility: visible;
    opacity: 1;
}

.logout-modal {
    background: #fff;
    padding: 28px 32px;
    border-radius: 16px;
    text-align: center;
    width: 320px;
    box-shadow: 0 10px 30px rgba(0, 0, 0, 0.2);
    transform: scale(0.9);
    transition: transform 0.3s ease;
}

.logout-overlay.show .logout-modal {
    transform: scale(1);
}

.logout-modal h5 {
    font-weight: 600;
    margin-bottom: 24px;
    line-height: 1.4;
    color: #333;
}

.logout-actions {
    display: flex;
    gap: 12px;
    justify-content: center;
}

.btn-logout {
        background:  linear-gradient(135deg,  #ff9f43,
#ee5253);
```

```
    border: none;
    color: white;
    padding: 10px 20px;
    border-radius: 10px;
    font-weight: 600;
    cursor: pointer;
    transition: all 0.3s ease;
}

.btn-logout:hover {
    transform: translateY(-2px);
    box-shadow: 0 4px 12px rgba(238, 82, 83, 0.3);
}

.btn-cancel {
    background: #2ec4b6;
    border: none;
    color: white;
    padding: 10px 20px;
    border-radius: 10px;
    font-weight: 600;
    cursor: pointer;
    transition: all 0.3s ease;
}

.btn-cancel:hover {
    transform: translateY(-2px);
    box-shadow: 0 4px 12px rgba(46, 196, 182, 0.3);
}

/* ========== COMING SOON NOTIFICATION ========== */
.coming-soon-notification {
    position: fixed;
    top: 80px;
    right: -300px;
    background: white;
    padding: 16px 20px;
    border-radius: 12px;
    box-shadow: 0 4px 12px rgba(0, 0, 0, 0.15);
    z-index: 9999;
    display: flex;
    align-items: center;
    gap: 12px;
```

```
    transition: right 0.3s ease;
}

.coming-soon-notification.show {
    right: 20px;
}

.notification-content {
    display: flex;
    align-items: center;
    gap: 12px;
}

.notification-content i {
    font-size: 24px;
    color: #3b82f6;
}

.notification-text {
    display: flex;
    flex-direction: column;
}

.notification-text strong {
    font-size: 14px;
    color: #333;
}

.notification-text span {
    font-size: 12px;
    color: #888;
}

/* Overlay untuk mobile */
.sidebar.show::before {
    content: "";
    position: fixed;
    top: 0;
    left: 280px;
    right: 0;
    bottom: 0;
    background: rgba(0, 0, 0, 0.5);
    z-index: -1;
```

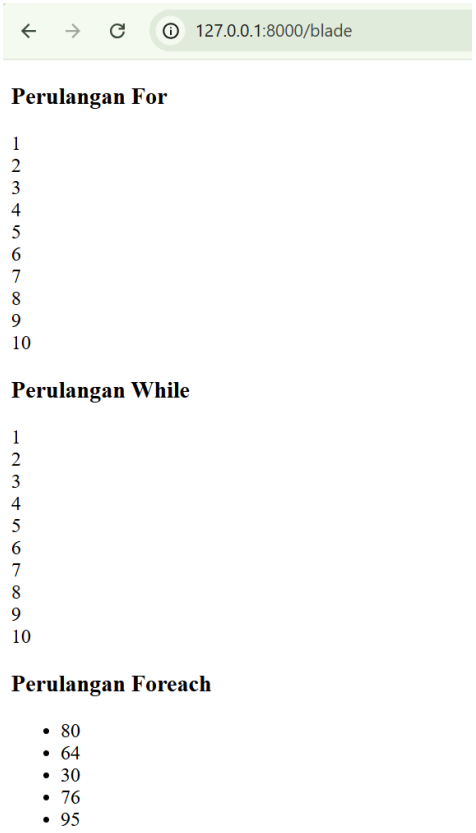
```
}
```

c. TUGAS - Blade Template Engine

Kerjakan tugas berikut menggunakan blade

i. Buat perulangan for untuk menampilkan bilangan 1 s.d 10

```
Langkah 1: Membuat Route Blade
Route::get('/blade-loop', function () {
$nilai = [80, 64, 30, 76, 95];
return view('blade', ['nilai' => $nilai]);
});
Langkah 2: Membuat View Blade
File: resources/views/blade.blade.php
<!DOCTYPE html>
<html>
<head>
<title>Blade Loop</title>
</head>
<body>
<h3>Perulangan For</h3>
@for ($i = 1; $i <= 10; $i++)
{{ $i }} <br>
@endfor
<h3>Perulangan While</h3>
@php $j = 1; @endphp
@while ($j <= 10)
{{ $j }} <br>
@php $j++; @endphp
@endwhile
<h3>Perulangan Foreach</h3>
<ul>
@foreach ($nilai as $n)
<li>{{ $n }}</li>
@endforeach
</ul>
</body>
</html>
```



ii. Buat perulangan while untuk menampilkan bilangan 1 s.d 10

```
Langkah 1: Membuat Controller. php artisan make:controller
MahasiswaController
Langkah 2: Mengisi Controller
File: app/Http/Controllers/MahasiswaController.php
<?php
namespace App\Http\Controllers;
class MahasiswaController extends Controller
{
public function index()
{
$mahasiswa = ['Aay', 'Tiur', 'Anis', 'Tulus'];
return view('mahasiswa', ['mahasiswa' => $mahasiswa]);
}
}
Langkah 3: Membuat Route ke Controller.
Tambahkan pada routes/web.php: use
App\Http\Controllers\MahasiswaController;
Route::get('/anak', [MahasiswaController::class, 'index']);
Langkah 4: Membuat View Mahasiswa
File: resources/views/mahasiswa.blade.php
<!DOCTYPE html>
<html>
<head>
```

```
<title>Data Mahasiswa</title>
</head>
<body>
<h2>Daftar Mahasiswa</h2>
<ul>
@foreach ($mahasiswa as $mhs)
<li>{{ $mhs }}</li>
@endforeach
</ul>
</body>
</html>
```

iii. Buat perulangan foreach untuk menampilkan nilai pada route berikut

```php
// Route untuk perulangan FOREACH
Route::get('/blade-foreach', function () {
    $nilai = [80, 64, 30, 76, 95];
    return view('blade-foreach', ['nilai' => $nilai]);
});
```

d. TUGAS - Controller

i. Buat root yang mengarah ke controller

```php
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Route;
use App\Http\Controllers\AuthController;
use App\Http\Controllers\Admin\SiswaController;
use App\Http\Controllers\Admin\GuruController;
use App\Http\Controllers\Admin\DataPenggunaController;
// ✅ TAMBAHKAN INI
use
App\Http\Controllers\Admin\AdminDashboardController;
// ✅ TAMBAHKAN INI
use App\Http\Controllers\Admin\PresensiController; //
✅ TAMBAHKAN INI
```

```
use
App\Http\Controllers\Admin\LaporanEvaluasiController;
// ✅ TAMBAHKAN INI
use App\Http\Controllers\Guru\GuruDashboardController;
use
App\Http\Controllers\Guru\GuruProfilSiswaController;
// ✅ TAMBAH INI
use  App\Http\Controllers\Guru\GuruPresensiController;
// ✅ TAMBAH INI
use
App\Http\Controllers\Guru\GuruPerkembanganController;
// ✅ TAMBAH INI
use
App\Http\Controllers\Guru\GuruLaporanEvaluasiControlle
r; // ✅ TAMBAH INI
use
App\Http\Controllers\Admin\RiwayatNotifikasiController
;
```

ii. Tampilkan template pada view menggunakan controller

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Hash;
use App\Models\User;
use App\Models\Siswa;

class AuthController extends Controller
{
    // ============================================
    // 1. LOGIN WEB (untuk Admin & Guru)
    // ============================================
    public function login(Request $request)
    {
        $credentials = $request->validate([
            'username' => 'required',
            'password' => 'required'
        ]);
```

```
        if (Auth::attempt($credentials)) {
            $user = Auth::user();

            // Redirect based on role
            if ($user->role === 'admin') {
                                                    return
redirect()->route('admin.dashboard_admin');
            } else if ($user->role === 'guru') {
                                                    return
redirect()->route('guru.dashboard');
            }

            return redirect('/');
        }

        return back()->withErrors([
                'username' => 'Username atau password
salah.',
        ]);
    }

    // ============================================
    // 2. LOGIN API (untuk Mobile App)
    // ============================================
    public function loginApi(Request $request)
    {
                    \Log::info('Login  API  Request:',
$request->all());

        $request->validate([
            'username' => 'required',
            'password' => 'required'
        ]);

        $username = $request->username;
        $password = $request->password;

        // ===== CEK LOGIN SEBAGAI GURU/ADMIN =====
                    $user  =  User::where('username',
$username)->first();
```

```
                if ($user && Hash::check($password,
$user->password)) {
                \Log::info('Login success as User:',
['user_id' => $user->id, 'role' => $user->role]);

            return response()->json([
                'success' => true,
                'message' => 'Login berhasil',
                'data' => [
                    'user_id' => $user->id,
                    'username' => $user->username,
                    'nama' => $user->name,
                    'role' => $user->role,
                     'token' => 'user_' . $user->id //
Simple token untuk testing
                ]
            ], 200);
        }

        // ===== CEK LOGIN SEBAGAI ORANG TUA =====
           // Login pakai no_hp (username) dan no_hp
(password)
                    $siswa = Siswa::where('no_hp',
$username)->first();

        if ($siswa && $password === $siswa->no_hp) {
                \Log::info('Login success as Ortu:',
['siswa_id' => $siswa->id]);

            return response()->json([
                'success' => true,
                'message' => 'Login berhasil',
                'data' => [
                    'user_id' => $siswa->id,
                    'username' => $siswa->no_hp,
                            'nama' => 'Orang Tua ' .
$siswa->nama_lengkap,
                    'role' => 'ortu',
                    'siswa_id' => $siswa->id,
                                'nama_siswa' =>
$siswa->nama_lengkap,
                    'token' => 'ortu_' . $siswa->id
                ]
```

```
            ], 200);
        }

         \Log::warning('Login failed:', ['username' =>
$username]);

        return response()->json([
            'success' => false,
                'message' => 'Username atau password
salah'
        ], 401);
    }

    // ============================================
    // 3. LOGOUT
    // ============================================
    public function logout(Request $request)
    {
        Auth::logout();
        return redirect('/login');
    }
}
```

BAB III
PENUTUP

KESIMPULAN

Dalam arsitektur MVC Laravel, Controller berfungsi sebagai pemisah antara logika bisnis dengan route dan view, di mana route hanya memetakan URL sementara controller mengelola pemrosesan data. Pada proyek ini, empat controller dibuat menggunakan perintah php artisan make:controller di direktori app/Http/Controllers untuk peran yang berbeda. PageController mengelola halaman umum seperti beranda dan login melalui method return view(). GuruController menangani fitur seperti presensi dan profil siswa dengan dukungan parameter dinamis seperti $id atau $kelas. AdminController bertanggung jawab atas manajemen sistem secara menyeluruh termasuk laporan evaluasi dengan parameter opsional, sedangkan OrangTuaController fokus pada data spesifik anak yang diasuh. Alur kerjanya dimulai saat rute dipanggil melalui sintaks Route::get('/url', [NamaController::class, 'namaMethod']), di mana Laravel akan menjalankan fungsi terkait, memproses data (saat ini masih data dummy sebelum terhubung ke Model), dan mengembalikan view kepada pengguna. Struktur ini menerapkan prinsip separation of concerns yang membuat aplikasi TPQSmart lebih modular, mudah dipelihara, dan memudahkan pengaturan izin akses (otorisasi) sesuai peran masing-masing pengguna.