LAMPIRAN

Source Code Halaman Dasboard

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Models\Obat;
use App\Models\PeriodePermintaan;
use App\Models\MetrikEvaluasi;
use App\Models\MonteCarlo;
use App\Models\HasilError;
use App\Models\LogProses;
use Illuminate\Support\Facades\DB;

class DashboardController extends Controller
{
    public function index()
    {
        $obats = Obat::all();
        $totalData = [];

        foreach ($obats as $obat) {
            $columnName = $this->getColumnName($obat->nama_obat);
            $total = PeriodePermintaan::sum($columnName);
            $totalData[] = [
                'nama_obat' => $obat->nama_obat,
                'total' => $total
            ];
        }

        // Data untuk pie chart
        $pieChartData = [];
        foreach ($obats as $obat) {
            $columnName = $this->getColumnName($obat->nama_obat);
            $total = PeriodePermintaan::sum($columnName);
            $pieChartData[] = [
                'name' => $obat->nama_obat,
                'y' => $total
            ];
        }

        // Tambahkan data statistik lainnya
        $totalPrediksi = MonteCarlo::distinct('id_obat', 'skenario')->count();
        $totalError = HasilError::count();
        $logAktivitas = LogProses::orderBy('created_at', 'desc')->limit(5)->get();

        return view('dashboard', compact('totalData', 'pieChartData', 'obats', 'totalPrediksi',
'totalError', 'logAktivitas'));
    }

    private function getColumnName($namaObat)
    {
        $columnMap = [
            'Clopidogrel 75 Mg' => 'clopidogrel_75_mg',
            'Candesartan 8 Mg' => 'candesartan_8_mg',
```

```
            'Isosorbid Dinitrate 5 Mg' => 'isosorbid_dinitrate_5_mg',
            'Nitrokaf Retard 2.5 Mg' => 'nitrokaf_retard_25_mg'
        ];

        return $columnMap[$namaObat] ?? '';
    }
}
@extends('layouts.app')
@section('title', 'Dashboard')
@section('content')
<div class="row">
    <div class="col-xl-3 col-md-6 mb-4">
        <div class="card border-start border-primary border-4 h-100">
            <div class="card-body">
                <div class="d-flex align-items-center">
                    <div class="flex-grow-1">
                        <div class="text-uppercase text-primary fw-bold small mb-1">Clopidogrel
75 Mg</div>
                        <div class="fs-4 fw-bold">{{ $totalData[0]['total'] ?? 0 }}</div>
                        <div class="text-muted">Total Data</div>
                    </div>
                    <div class="ms-3">
                        <div class="bg-primary bg-opacity-10 p-3 rounded-circle">
                            <i class="fas fa-pills text-primary fa-lg"></i>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
    <div class="col-xl-3 col-md-6 mb-4">
        <div class="card border-start border-success border-4 h-100">
            <div class="card-body">
                <div class="d-flex align-items-center">
                    <div class="flex-grow-1">
                        <div class="text-uppercase text-success fw-bold small mb-1">Candesartan
8 Mg</div>
                        <div class="fs-4 fw-bold">{{ $totalData[1]['total'] ?? 0 }}</div>
                        <div class="text-muted">Total Data</div>
                    </div>
                    <div class="ms-3">
                        <div class="bg-success bg-opacity-10 p-3 rounded-circle">
                            <i class="fas fa-heartbeat text-success fa-lg"></i>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
        <div class="col-xl-3 col-md-6 mb-4">
        <div class="card border-start border-warning border-4 h-100">
            <div class="card-body">
                <div class="d-flex align-items-center">
                    <div class="flex-grow-1">
                        <div class="text-uppercase text-warning fw-bold small mb-1">Isosorbid
```

```
Dinitrate 5 Mg</div>
                    <div class="fs-4 fw-bold">{{ $totalData[2]['total'] ?? 0 }}</div>
                    <div class="text-muted">Total Data</div>
                  </div>
                  <div class="ms-3">
                    <div class="bg-warning bg-opacity-10 p-3 rounded-circle">
                      <i class="fas fa-heart text-warning fa-lg"></i>
                    </div>
                  </div>
                </div>
              </div>
            </div>
          </div>
            <div class="col-xl-3 col-md-6 mb-4">
            <div class="card border-start border-info border-4 h-100">
              <div class="card-body">
                <div class="d-flex align-items-center">
                  <div class="flex-grow-1">
                    <div class="text-uppercase text-info fw-bold small mb-1">Nitrokaf Retard
2.5 Mg</div>
                    <div class="fs-4 fw-bold">{{ $totalData[3]['total'] ?? 0 }}</div>
                    <div class="text-muted">Total Data</div>
                  </div>
                  <div class="ms-3">
                    <div class="bg-info bg-opacity-10 p-3 rounded-circle">
                      <i class="fas fa-stethoscope text-info fa-lg"></i>
                    </div>
                  </div>
                </div>
              </div>
            </div>
          </div>
        </div>
        <div class="row mt-4">
          <div class="col-lg-8 mb-4">
            <div class="card shadow-sm h-100">
              <div class="card-header bg-white border-bottom-0 py-3">
                <h5 class="card-title mb-0 d-flex align-items-center">
                  <i class="fas fa-chart-line text-primary me-2"></i>
                  Metode Monte Carlo & Informasi Obat
                </h5>
              </div>
              <div class="card-body pt-0">
                <div class="mb-4">
                  <h6 class="text-primary mb-3 fw-bold d-flex align-items-center">
                    <i class="fas fa-calculator me-2"></i>
                    Metode Monte Carlo
                  </h6>
                  <div class="card border-light mb-3">
                    <div class="card-body">
                      <p class="mb-0">
                        Metode Monte Carlo adalah teknik simulasi komputer yang
menggunakan sampling acak untuk memperkirakan hasil numerik. Dalam konteks prediksi
permintaan obat, metode ini menggunakan data historis untuk membangun distribusi
probabilitas permintaan, kemudian menggunakan bilangan acak untuk simulasi permintaan
```

```
masa depan.
                    </p>
                  </div>
                </div>

                <h6 class="text-primary mb-3 fw-bold d-flex align-items-center">
                  <i class="fas fa-list-ol me-2"></i>
                  Langkah-langkah Prediksi Permintaan Obat
                </h6>
                <div class="row">
                  <div class="col-md-6">
                    <ul class="list-group list-group-flush">
                      <li class="list-group-item d-flex align-items-center">
                        <span class="badge bg-primary rounded-circle me-3">1</span>
                        Mengumpulkan data historis permintaan obat
                      </li>
                      <li class="list-group-item d-flex align-items-center">
                        <span class="badge bg-primary rounded-circle me-3">2</span>
                        Menghitung frekuensi setiap nilai permintaan
                      </li>
                      <li class="list-group-item d-flex align-items-center">
                        <span class="badge bg-primary rounded-circle me-3">3</span>
                        Menghitung distribusi probabilitas
                      </li>
                      <li class="list-group-item d-flex align-items-center">
                        <span class="badge bg-primary rounded-circle me-3">4</span>
                        Menghitung distribusi probabilitas kumulatif
                      </li>
                    </ul>
                  </div>
                  <div class="col-md-6">
                    <ul class="list-group list-group-flush">
                      <li class="list-group-item d-flex align-items-center">
                        <span class="badge bg-primary rounded-circle me-3">5</span>
                        Menentukan interval bilangan acak
                      </li>
                      <li class="list-group-item d-flex align-items-center">
                        <span class="badge bg-primary rounded-circle me-3">6</span>
                        Memilih bilangan acak untuk simulasi
                      </li>
                      <li class="list-group-item d-flex align-items-center">
                        <span class="badge bg-primary rounded-circle me-3">7</span>
                        Mengulang proses simulasi berkali-kali
                      </li>
                      <li class="list-group-item d-flex align-items-center">
                        <span class="badge bg-primary rounded-circle me-3">8</span>
                        Menghitung rata-rata hasil simulasi
                      </li>
                    </ul>
                  </div>
                </div>
              </div>
              <hr>
              <div>
                <h6 class="text-primary mb-3 fw-bold d-flex align-items-center">
```

```
                    <i class="fas fa-heartbeat me-2"></i>
                    Jenis Obat Jantung
                </h6>
                <!-- SLIDER/CAROUSEL DENGAN BACKGROUND GAMBAR -->
                <div class="obat-slider-container">
                    <div class="obat-slider-track" id="obatSliderTrack">
                        <!-- Slide 1: Clopidogrel 75 Mg -->
                        <div class="obat-slide active" data-slide="1">
                            <div class="obat-card">
                                <div class="obat-image" style="background-image:
url('https://drotafarma.com.ve/wp-content/uploads/2024/03/clopidogrel-75-mg-
drotafarma--2048x1365.png');"></div>
                                <div class="obat-content">
                                    <div class="obat-header">
                                        <div class="obat-icon">
                                            <i class="fas fa-pills"></i>
                                        </div>
                                        <div class="obat-name">Clopidogrel</div>
                                        <div class="obat-dosis">75 Mg</div>
                                    </div>
                                    <div class="obat-body">
                                        <div class="obat-description">
                                            <p><strong>Kategori:</strong> Antiplatelet</p>
                                            <p class="mb-0">Obat antiplatelet untuk mencegah
pembentukan gumpalan darah pada pasien dengan riwayat serangan jantung atau
stroke.</p>
                                        </div>
                                    </div>
                                </div>
                            </div>
                        </div>
                        <!-- Slide 2: Candesartan 8 Mg -->
                        <div class="obat-slide" data-slide="2">
                            <div class="obat-card">
                                <div class="obat-image" style="background-image:
url('https://coopidrogas.vtexassets.com/arquivos/ids/36831599/candesartan-8-mg-30-
tabletas-mkm13999.jpg?v=638413494379730000');"></div>
                                <div class="obat-content">
                                    <div class="obat-header">
                                        <div class="obat-icon">
                                            <i class="fas fa-heartbeat"></i>
                                        </div>
                                        <div class="obat-name">Candesartan</div>
                                        <div class="obat-dosis">8 Mg</div>
                                    </div>
                                    <div class="obat-body">
                                        <div class="obat-description">
                                            <p><strong>Kategori:</strong> ARB (Angiotensin II
Receptor Blocker)</p>
                                            <p class="mb-0">Mengobati tekanan darah tinggi dan gagal
jantung dengan memblokir efek hormon angiotensin II.</p>
                                        </div>
                                    </div>
                                </div>
                            </div>
```

```
                    </div>
                    <!-- Slide 3: Isosorbid Dinitrate 5 Mg -->
                    <div class="obat-slide" data-slide="3">
                      <div class="obat-card">
                        <div class="obat-image" style="background-image: url('https://res-
4.cloudinary.com/dk0z4ums3/image/upload/c_scale,h_500,w_500/v1/production/pharmac
y/products/1659780885_6231c77089446a1af45d8ec5_isosorbide_5_mg_10_tablet');"></d
iv>
                        <div class="obat-content">
                          <div class="obat-header">
                            <div class="obat-icon">
                              <i class="fas fa-heart"></i>
                            </div>
                            <div class="obat-name">Isosorbid Dinitrate</div>
                            <div class="obat-dosis">5 Mg</div>
                          </div>
                          <div class="obat-body">
                            <div class="obat-description">
                              <p><strong>Kategori:</strong> Nitrat</p>
                              <p class="mb-0">Mencegah nyeri dada (angina) dengan
melebarkan pembuluh darah, sehingga meningkatkan aliran darah ke jantung.</p>
                            </div>
                          </div>
                        </div>
                      </div>
                    </div>
                    <!-- Slide 4: Nitrokaf Retard 2.5 Mg -->
                    <div class="obat-slide" data-slide="4">
                      <div class="obat-card">
                        <div class="obat-image" style="background-image:
url('https://d2qjkwm11akmwu.cloudfront.net/products/142621_30-12-2021_11-28-
58.png');"></div>
                        <div class="obat-content">
                          <div class="obat-header">
                            <div class="obat-icon">
                              <i class="fas fa-stethoscope"></i>
                            </div>
                            <div class="obat-name">Nitrokaf Retard</div>
                            <div class="obat-dosis">2.5 Mg</div>
                          </div>
                          <div class="obat-body">
                            <div class="obat-description">
                              <p><strong>Kategori:</strong> Nitrat Retard</p>
                              <p class="mb-0">Mengobati dan mencegah nyeri dada
(angina) dengan melebarkan pembuluh darah di jantung.</p>
                            </div>
                          </div>
                        </div>
                      </div>
                    </div>
                  </div>
                    <!-- Slider Controls -->
                  <div class="slider-controls">
                    <button class="slider-btn" id="prevBtn" aria-label="Previous slide">
                      <i class="fas fa-chevron-left"></i>
```

```
                    </button>
                                    <div class="slider-counter">
                      <span id="currentSlide">1</span> / <span id="totalSlides">4</span>
                    </div>
                    <div class="slider-dots" id="sliderDots">
                      <!-- Dots will be generated by JavaScript -->
                    </div>
                    <button class="slider-btn" id="nextBtn" aria-label="Next slide">
                      <i class="fas fa-chevron-right"></i>
                    </button>
                  </div>
                </div>
              </div>
            </div>
          </div>
          <div class="col-lg-4 mb-4">
            <div class="row">
              <!-- Pie Chart Card -->
              <div class="col-12 mb-4">
                <div class="card shadow-sm h-100">
                  <div class="card-header bg-white border-bottom-0 py-3">
                    <h5 class="card-title mb-0 d-flex align-items-center">
                      <i class="fas fa-chart-pie text-success me-2"></i>
                      Distribusi Data Obat
                    </h5>
                  </div>
                  <div class="card-body pt-0">
                    <div class="d-flex justify-content-center">
                      <div style="width: 240px; height: 240px;">
                        <canvas id="pieChart"></canvas>
                      </div>
                    </div>
                    <div class="mt-4">
                      <div class="table-responsive">
                        <table class="table table-sm table-borderless">
                          <tbody>
                            @foreach($pieChartData as $index => $data)
                            @php
                              $colors = ['#FF6384', '#36A2EB', '#FFCE56', '#4BC0C0'];
                            @endphp
                            <tr>
                              <td>
                                <span class="d-inline-block me-2" style="width: 12px;
height: 12px; background-color: {{ $colors[$index] }};"></span>
                                <span class="small">{{ $data['name'] }}</span>
                              </td>
                              <td class="text-end fw-bold small">{{ $data['y'] }}</td>
                            </tr>
                            @endforeach
                            <tr class="border-top">
                              <td class="fw-bold small">Total Data</td>
                              <td class="text-end fw-bold small">
                                {{ array_sum(array_column($pieChartData, 'y')) }}
                              </td>
```

```
                        </tr>
                      </tbody>
                    </table>
                  </div>
                </div>
              </div>
            </div>
          </div>
          <!-- Log Aktivitas -->
          <div class="col-12">
            <div class="card shadow-sm h-100">
              <div class="card-header bg-white border-bottom-0 py-3">
                <h5 class="card-title mb-0 d-flex align-items-center">
                  <i class="fas fa-history text-warning me-2"></i>
                  Aktivitas Terbaru
                </h5>
              </div>
              <div class="card-body pt-0">
                <div class="timeline">
                  @forelse($logAktivitas as $log)
                  <div class="timeline-item mb-3">
                    <div class="d-flex">
                      <div class="flex-shrink-0">
                        @if($log->status == 'SUCCESS')
                        <div class="bg-success rounded-circle p-2">
                          <i class="fas fa-check text-white"></i>
                        </div>
                        @elseif($log->status == 'ERROR')
                        <div class="bg-danger rounded-circle p-2">
                          <i class="fas fa-times text-white"></i>
                        </div>
                        @else
                        <div class="bg-warning rounded-circle p-2">
                          <i class="fas fa-sync text-white"></i>
                        </div>
                        @endif
                      </div>
                      <div class="flex-grow-1 ms-3">
                        <h6 class="mb-1 small">{{ $log->proses }}</h6>
                        <p class="mb-1 text-muted small">{{ Str::limit($log->pesan, 50)
}}</p>
                        <small class="text-muted">{{ \Carbon\Carbon::parse($log-
>created_at)->diffForHumans() }}</small>
                      </div>
                    </div>
                  </div>
                  @empty
                  <div class="text-center text-muted py-3">
                    <i class="fas fa-history fa-lg mb-2"></i>
                    <p class="small mb-0">Belum ada aktivitas</p>
                  </div>
                  @endforelse
                </div>
              </div>
            </div>
```

```
            </div>
        </div>
    </div>
</div>
@endsection
@section('scripts')
<script>
    document.addEventListener('DOMContentLoaded', function() {
        // Pie Chart
        const ctx = document.getElementById('pieChart').getContext('2d');
        const pieChart = new Chart(ctx, {
            type: 'doughnut',
            data: {
                labels: @json(array_column($pieChartData, 'name')),
                datasets: [{
                    data: @json(array_column($pieChartData, 'y')),
                    backgroundColor: [
                        'rgba(255, 99, 132, 0.8)',
                        'rgba(54, 162, 235, 0.8)',
                        'rgba(255, 206, 86, 0.8)',
                        'rgba(75, 192, 192, 0.8)'
                    ],
                    borderColor: [
                        'rgba(255, 99, 132, 1)',
                        'rgba(54, 162, 235, 1)',
                        'rgba(255, 206, 86, 1)',
                        'rgba(75, 192, 192, 1)'
                    ],
                    borderWidth: 1,
                    hoverOffset: 15
                }]
            },
            options: {
                responsive: true,
                maintainAspectRatio: false,
                plugins: {
                    legend: {
                        display: false
                    },
                    tooltip: {
                        callbacks: {
                            label: function(context) {
                                let label = context.label || '';
                                if (label) {
                                    label += ': ';
                                }
                                const value = context.raw;
                                const total = context.dataset.data.reduce((a, b) => a + b, 0);
                                const percentage = Math.round((value / total) * 100);
                                return `${label}${value} data (${percentage}%)`;
                            }
                        }
                    }
                }
            }
```

```
});
// SLIDER FUNCTIONALITY
function initObatSlider() {
    const track = document.getElementById('obatSliderTrack');
    const slides = document.querySelectorAll('.obat-slide');
    const prevBtn = document.getElementById('prevBtn');
    const nextBtn = document.getElementById('nextBtn');
    const currentSlideEl = document.getElementById('currentSlide');
    const totalSlidesEl = document.getElementById('totalSlides');
    const dotsContainer = document.getElementById('sliderDots');
            let currentSlide = 0;
    const totalSlides = slides.length;
            // Set total slides
    totalSlidesEl.textContent = totalSlides;
            // Create dots
    slides.forEach((_, index) => {
        const dot = document.createElement('div');
        dot.className = `slider-dot ${index === 0 ? 'active' : ''}`;
        dot.dataset.index = index;
        dot.addEventListener('click', () => goToSlide(index));
        dotsContainer.appendChild(dot);
    });
            // Update slide position
    function updateSlidePosition() {
        track.style.transform = `translateX(-${currentSlide * 100}%)`;
                // Update active class on slides
        slides.forEach((slide, index) => {
            slide.classList.toggle('active', index === currentSlide);
        });
                // Update current slide indicator
        currentSlideEl.textContent = currentSlide + 1;
                // Update active dot
        document.querySelectorAll('.slider-dot').forEach((dot, index) => {
            dot.classList.toggle('active', index === currentSlide);
        });
                // Update button states
        prevBtn.disabled = currentSlide === 0;
        nextBtn.disabled = currentSlide === totalSlides - 1;
                // Add fade animation
        const activeSlide = slides[currentSlide];
        activeSlide.style.animation = 'none';
        setTimeout(() => {
            activeSlide.style.animation = 'fadeIn 0.5s ease';
        }, 10);
    }
            // Go to specific slide
    function goToSlide(index) {
        if (index >= 0 && index < totalSlides) {
            currentSlide = index;
            updateSlidePosition();
        }
    }
            // Next slide
    function nextSlide() {
        if (currentSlide < totalSlides - 1) {
```

```
            currentSlide++;
            updateSlidePosition();
        }
    }
            // Previous slide
    function prevSlide() {
        if (currentSlide > 0) {
            currentSlide--;
            updateSlidePosition();
        }
    }
            // Event listeners for buttons
    prevBtn.addEventListener('click', prevSlide);
    nextBtn.addEventListener('click', nextSlide);
            // Keyboard navigation
    document.addEventListener('keydown', (e) => {
        if (e.key === 'ArrowLeft') prevSlide();
        if (e.key === 'ArrowRight') nextSlide();
    });
            // Swipe functionality for mobile
    let startX = 0;
    let endX = 0;
            track.addEventListener('touchstart', (e) => {
        startX = e.touches[0].clientX;
    });
            track.addEventListener('touchmove', (e) => {
        endX = e.touches[0].clientX;
    });
            track.addEventListener('touchend', () => {
        const diff = startX - endX;
        const threshold = 50;
        if (Math.abs(diff) > threshold) {
            if (diff > 0) {
            // Swipe left
            nextSlide();
            } else {
            // Swipe right
            prevSlide();
            }
        }
    });
            // Auto slide every 10 seconds
    let autoSlideInterval = setInterval(nextSlide, 10000);
            // Pause auto-slide on hover
    track.addEventListener('mouseenter', () => {
        clearInterval(autoSlideInterval);
    });
            track.addEventListener('mouseleave', () => {
        autoSlideInterval = setInterval(nextSlide, 10000);
    });
            // Initialize
    updateSlidePosition();
}
        // Initialize slider
initObatSlider();
```

```
            // Handle image loading errors
        function handleImageErrors() {
            const images = document.querySelectorAll('.obat-image');
            const fallbackColors = [
                'linear-gradient(135deg, #1a5fb4 0%, #3584e4 100%)',
                'linear-gradient(135deg, #26a269 0%, #33d17a 100%)',
                'linear-gradient(135deg, #c64600 0%, #ff7800 100%)',
                'linear-gradient(135deg, #613583 0%, #9141ac 100%)'
            ];
              images.forEach((img, index) => {
                // Check if image loaded successfully
                const tempImg = new Image();
                tempImg.src = img.style.backgroundImage.replace('url("', '').replace('")', '');
                tempImg.onerror = () => {
                    // Replace with gradient if image fails to load
                    img.style.backgroundImage = 'none';
                    img.style.background = fallbackColors[index % fallbackColors.length];
                };
            });
        }
            handleImageErrors();
    });
</script>
<style>
    /* Custom styling untuk dashboard */
    .card {
        border: 1px solid rgba(0,0,0,.125);
        transition: transform 0.2s ease-in-out;
    }
        .card:hover {
        transform: translateY(-2px);
    }
        .card-title {
        color: #2c3e50;
        font-weight: 600;
    }
        .border-start {
        border-left-width: 4px !important;
    }
        .list-group-item {
        border: none;
        padding: 0.75rem 0;
        background: transparent;
    }
        .badge.bg-primary.rounded-circle {
        width: 24px;
        height: 24px;
        display: flex;
        align-items: center;
        justify-content: center;
        font-size: 0.75rem;
    }
        /* Styling untuk konten slider */
    .obat-description p {
        margin-bottom: 0.5rem;
```

```
    line-height: 1.5;
}
    .obat-description p strong {
    color: #fff;
}
    /* STYLING SLIDER BARU */
.obat-slider-container {
    position: relative;
    overflow: hidden;
    margin: 30px 0;
    border-radius: 16px;
    box-shadow: 0 5px 15px rgba(0,0,0,0.1);
}
    .obat-slider-track {
    display: flex;
    transition: transform 0.5s ease-in-out;
}
    .obat-slide {
    flex: 0 0 100%;
    min-width: 100%;
    box-sizing: border-box;
    padding: 5px;
}
    .obat-card {
    position: relative;
    border-radius: 12px;
    overflow: hidden;
    height: 280px;
}
    .obat-image {
    position: absolute;
    top: 0;
    left: 0;
    right: 0;
    bottom: 0;
    background-size: cover;
    background-position: center;
    filter: brightness(0.8);
    transition: transform 0.5s ease;
}
    .obat-card:hover .obat-image {
    transform: scale(1.05);
}
    .obat-content {
    position: relative;
    z-index: 2;
    height: 100%;
    display: flex;
    flex-direction: column;
    background: linear-gradient(to bottom, rgba(0,0,0,0.3) 0%, rgba(0,0,0,0.7) 100%);
    color: white;
}
    .obat-header {
    padding: 20px 20px 10px;
    text-align: center;
```

```
    flex-shrink: 0;
}
  .obat-icon {
    font-size: 2rem;
    margin-bottom: 10px;
    color: white;
}
    .obat-name {
    font-size: 1.3rem;
    font-weight: 700;
    margin-bottom: 5px;
}

.obat-dosis {
    font-size: 1rem;
    opacity: 0.95;
    font-weight: 600;
}

.obat-body {
    padding: 10px 20px 20px;
    flex-grow: 1;
    display: flex;
    align-items: center;
}

.obat-description {
    font-size: 0.9rem;
    line-height: 1.5;
    color: rgba(255, 255, 255, 0.95);
    margin-bottom: 0;
}

/* Slider controls */
.slider-controls {
    display: flex;
    justify-content: center;
    align-items: center;
    margin-top: 15px;
    gap: 15px;
}

.slider-btn {
    background-color: var(--secondary-color);
    color: white;
    border: none;
    width: 40px;
    height: 40px;
    border-radius: 50%;
    display: flex;
    align-items: center;
    justify-content: center;
    font-size: 1rem;
    cursor: pointer;
    transition: all 0.3s;
```

```
    box-shadow: 0 4px 8px rgba(13, 110, 253, 0.2);
}

.slider-btn:hover {
    background-color: #0b5ed7;
    transform: scale(1.1);
}

.slider-btn:disabled {
    background-color: #ccc;
    cursor: not-allowed;
    transform: none;
    box-shadow: none;
}

.slider-dots {
    display: flex;
    gap: 8px;
}

.slider-dot {
    width: 10px;
    height: 10px;
    border-radius: 50%;
    background-color: #ddd;
    cursor: pointer;
    transition: all 0.3s;
}

.slider-dot.active {
    background-color: var(--secondary-color);
    transform: scale(1.2);
}
    .slider-counter {
    font-weight: 600;
    color: var(--primary-color);
    font-size: 0.9rem;
    min-width: 60px;
    text-align: center;
}

/* Animation for slide change */
@keyframes fadeIn {
    from { opacity: 0.8; }
    to { opacity: 1; }
}
.obat-slide.active {
    animation: fadeIn 0.5s ease;
}
/* Timeline styling */
.timeline-item:not(:last-child) {
    padding-bottom: 15px;
    border-bottom: 1px solid #eee;
    margin-bottom: 15px;
}
```

```
        @media (max-width: 768px) {
         .card-body .row > [class*="col-"] {
            margin-bottom: 1rem;
         }
              .card-body .row > [class*="col-"]:last-child {
            margin-bottom: 0;
         }
              .obat-card {
            height: 250px;
         }
              .obat-header {
            padding: 15px 15px 5px;
         }
              .obat-body {
            padding: 5px 15px 15px;
         }
              .obat-name {
            font-size: 1.2rem;
         }
              .obat-dosis {
            font-size: 0.9rem;
         }
              .obat-icon {
            font-size: 1.8rem;
         }
              .slider-btn {
            width: 35px;
            height: 35px;
         }
              .slider-counter {
            min-width: 50px;
            font-size: 0.8rem;
         }
      }
    </style>
    @endsection
```

Source Code Halaman Data set

```
@extends('layouts.app')
@section('title', 'Data Set & Prediksi Obat Jantung')
@section('content')
<div class="container-fluid">
   <div class="row">
      <div class="col-12">
         {{-- FILTER TAHUN --}}
         <form method="GET" action="{{ route('data.index') }}" class="row g-2 align-items-
center mb-3">
            <div class="col-auto">
               <label for="tahun" class="col-form-label">Filter Tahun</label>
            </div>
            <div class="col-auto">
               <select name="tahun" id="tahun" class="form-select"
onchange="this.form.submit()">
                  <option value="" {{ empty($selectedYear) ? 'selected' : '' }}>Semua</option>
                  @foreach($availableYears as $y)
```

```
                    <option value="{{ $y }}" {{ (int)$selectedYear === (int)$y ? 'selected' : '' }}>
                      {{ $y }}
                    </option>
                  @endforeach
                </select>
              </div>
            </form>
            {{-- KARTU DATA SET --}}
            <div class="card shadow">
              <div class="card-header bg-primary text-white d-flex justify-content-between align-
items-center">
                <h5 class="mb-0">
                  <i class="fas fa-pills me-2"></i>Data Set Permintaan Obat Jantung
                </h5>
                <div class="d-flex gap-2">
                  <a href="{{ route('data.create') }}" class="btn btn-light btn-sm">
                    <i class="fas fa-plus me-2"></i> Tambah Data
                  </a>
                  <a href="{{ route('data.import.form') }}" class="btn btn-light btn-sm">
                    <i class="fas fa-file-excel me-2"></i> Import Excel
                  </a>
                  {{-- tombol hapus semua data set --}}
                  <form action="{{ route('data.clear') }}" method="POST"
                    onsubmit="return confirm('Hapus SEMUA data (harian, bulanan,
simulasi)?');">
                    @csrf
                    <button type="submit" class="btn btn-danger btn-sm">
                      <i class="fas fa-trash-alt me-2"></i> Reset
                    </button>
                  </form>
                </div>
              </div>
              <div class="card-body">
                {{-- pesan sukses & error --}}
                @if(session('success'))
                  <div class="alert alert-success alert-dismissible fade show" role="alert">
                    <i class="fas fa-check-circle me-2"></i>{{ session('success') }}
                    <button type="button" class="btn-close" data-bs-dismiss="alert"></button>
                  </div>
                @endif
                @if(session('error'))
                  <div class="alert alert-danger alert-dismissible fade show" role="alert">
                    <i class="fas fa-exclamation-circle me-2"></i>{{ session('error') }}
                    <button type="button" class="btn-close" data-bs-dismiss="alert"></button>
                  </div>
                @endif
                {{-- INFO ALUR SISTEM --}}
                <div class="alert alert-info">
                  <i class="fas fa-info-circle me-2"></i>
                  Alur sistem:
                  <ol class="mb-0">
                    <li><strong>Import Dataset</strong> harian dari Excel (menu di atas).</li>
                    <li>Dataset otomatis terakumulasi menjadi <strong>data
bulanan</strong>.</li>
                    <li>Buka menu <strong>Monte Carlo</strong> di sidebar untuk memilih
```

```
skenario dan menjalankan simulasi.</li>
                    <li>Hasil simulasi dan error dapat dilihat di menu <strong>Hasil Error / Rekap</strong>.</li>
                </ol>
            </div>
            {{-- TOMBOL LANJUT KE MONTE CARLO --}}
            <div class="mb-3 text-end">
                <a href="{{ route('monte-carlo.index') }}" class="btn btn-primary btn-sm">
                    <i class="fas fa-forward me-1"></i> Lanjut ke Simulasi Monte Carlo
                </a>
            </div>
            {{-- RINGKASAN DATA SET HARIAN --}}
            <div class="card shadow-sm mt-3">
                <div class="card-header bg-light">
                    <h6 class="mb-0">
                        <i class="fas fa-table me-2"></i>Ringkasan Data Set (Harian)
                    </h6>
                </div>
                <div class="card-body">
                    @php
                        $rowsHarian = $detailHarian ?? collect();
                    @endphp
                    <div class="table-responsive">
                        <table class="table table-bordered table-hover mb-0 align-middle">
                            <thead class="table-light">
                                <tr>
                                    <th style="width: 80px;" class="text-center">No</th>
                                    <th style="width: 140px;" class="text-center">Tanggal</th>
                                    @foreach($obats as $obat)
                                        <th class="text-center">
                                            Permintaan Obat Jantung<br>
                                            <small class="text-muted">{{ $obat->nama_obat }}</small>
                                        </th>
                                    @endforeach
                                    <th style="width: 170px;" class="text-center">Aksi</th>
                                </tr>
                            </thead>
                            <tbody>
                                @forelse($rowsHarian as $index => $row)
                                    @php
                                        $tglKey = \Carbon\Carbon::parse($row->tanggal)->format('Y-m-d');
                                    @endphp
                                    <tr>
                                        <td class="text-center">{{ $index + 1 }}</td>
                                        <td class="text-center">{{ \Carbon\Carbon::parse($row->tanggal)->format('d/m/Y') }}</td>
                                        @foreach($obats as $obat)
                                            @php
                                                $colName = $columnMapById[$obat->id_obat] ?? null;
                                                $nilai   = $colName ? ($row->{$colName} ?? 0) : 0;
                                            @endphp
                                            <td class="text-center">{{ $nilai }}</td>
                                        @endforeach
                                        <td class="text-center">
```

```
                                    <a href="{{ route('data.harian.edit', $tglKey) }}" class="btn
btn-warning btn-sm">
                                        Edit
                                    </a>
                                    <button type="button"
                                            class="btn btn-danger btn-sm js-btn-delete"
                                            data-bs-toggle="modal"
                                            data-bs-target="#modalDeleteHarian"
                                            data-tanggal="{{ $tglKey }}"
                                            data-action="{{ route('data.harian.destroy', $tglKey) }}">
                                        Hapus
                                    </button>
                                </td>
                            </tr>
                            @empty
                                <tr>
                                    <td colspan="{{ 3 + $obats->count() }}" class="text-center">
                                        Belum ada data permintaan harian. Silakan import Excel
terlebih dahulu.
                                    </td>
                                </tr>
                            @endforelse
                        </tbody>
                    </table>
                </div>
                <p class="mt-3 text-center text-muted small mb-0">
                    Tabel Data SET (Harian) — sesuai data di Excel.
                </p>
            </div>
        </div>
    </div>
</div>
{{-- MODAL KONFIRMASI HAPUS (Bootstrap) --}}
<div class="modal fade" id="modalDeleteHarian" tabindex="-1" aria-hidden="true">
    <div class="modal-dialog modal-dialog-centered">
        <div class="modal-content">
            <div class="modal-header bg-danger text-white">
                <h5 class="modal-title">Konfirmasi Hapus</h5>
                <button type="button" class="btn-close btn-close-white" data-bs-
dismiss="modal"></button>
            </div>
            <div class="modal-body">
                Yakin hapus data tanggal <strong id="deleteTanggalText">-</strong>?
                <div class="text-muted small mt-2">
                    Data bulanan untuk bulan terkait akan ikut ter-update.
                </div>
            </div>
            <div class="modal-footer">
                <button type="button" class="btn btn-secondary" data-bs-
dismiss="modal">Batal</button>
                <form id="formDeleteHarian" method="POST" action="#">
            @csrf
                    @method('DELETE')
                    <button type="submit" class="btn btn-danger">Ya, Hapus</button>
                </form>
```

```
                    </div>
                </div>
            </div>
        </div>
    </div>
</div>
@endsection
@section('scripts')
<script>
document.addEventListener('click', function (e) {
    const btn = e.target.closest('.js-btn-delete');
    if (!btn) return;
    const tanggal = btn.getAttribute('data-tanggal');
    const action  = btn.getAttribute('data-action');
    document.getElementById('deleteTanggalText').textContent = tanggal;
    document.getElementById('formDeleteHarian').setAttribute('action', action);
});
</script>
<style>
    .card:hover {
        box-shadow: 0 4px 15px rgba(0, 0, 0, .1);
        transition: box-shadow 0.3s ease;
    }
</style>
@endsection
```

Source Code Halaman *Monte Carlo*

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Models\Obat;
use App\Models\PeriodePermintaan;
use Illuminate\Support\Facades\DB;
use Illuminate\Support\Facades\Schema;
use Illuminate\Support\Facades\Log;
use Carbon\Carbon;
use PhpOffice\PhpSpreadsheet\IOFactory;
use PhpOffice\PhpSpreadsheet\Cell\Coordinate;
use App\Models\MetrikEvaluasi;
class MonteCarloController extends Controller
{
    private $obatMapping = [
        'Clopidogrel 75 Mg'        => 'clopidogrel_75_mg',
        'Candesartan 8 Mg'         => 'candesartan_8_mg',
        'Isosorbid Dinitrate 5 Mg' => 'isosorbid_dinitrate_5_mg',
        'Nitrokaf Retard 2.5 Mg'   => 'nitrokaf_retard_25_mg',
    ];
    private function getExcelTemplatePath(string $skenario): ?string
    {
        $base = storage_path('app/montecarlo');
        if ($skenario === 'Skenario 1') return $base . DIRECTORY_SEPARATOR .
'SKNENARIO 1 FIX HARIAN.xlsx';
        if ($skenario === 'Skenario 2') return $base . DIRECTORY_SEPARATOR . 'SKENARIO
2 FIX BULANAN.xlsx';
```

```
        if ($skenario === 'Skenario 3') return $base . DIRECTORY_SEPARATOR . 'SKENARIO
3 FIX TAHUNAN.xlsx';
        return null;
    }
    public function index(\Illuminate\Http\Request $request)
    {
        $obats = Obat::orderBy('id_obat')->get();
        // Ambil dari query kalau ada, kalau tidak ambil dari session
        $selectedObatId = $request->query('obat_id', session('mc_selected_obat_id'));
        $selectedSkenario = $request->query('skenario', session('mc_selected_skenario'));
        // Rekap permanen
        $rekap = session('mc_rekap');
        return view('monte-carlo.index', compact('obats', 'rekap', 'selectedObatId',
'selectedSkenario'));
    }

    public function run(Request $request)
    {
        $request->validate([
            'obat_id'  => 'required|exists:tb_obat,id_obat',
            'skenario' => 'required|in:Skenario 1,Skenario 2,Skenario 3',
        ]);
        try {
            $idObat   = (int) $request->obat_id;
            $skenario = $request->input('skenario', 'Skenario 1');

            $obat = Obat::findOrFail($idObat);

            $columnName = $this->getColumnName($obat->nama_obat);
            if (empty($columnName)) {
                throw new \Exception('Kolom database tidak ditemukan untuk obat: ' . $obat-
>nama_obat);
            }
            $series = $this->getHistoricalSeries($skenario, $columnName);
            if (count($series) < 2) {
                throw new \Exception('Data historis minimal 2 periode. Data tersedia: ' .
count($series) . ' periode.');
            }
            [$trainingSeries, $ujiSeries] = $this->splitTrainingAndTestData($skenario, $series);
            if (count($trainingSeries) === 0) throw new \Exception('Data training tidak tersedia
untuk skenario ini.');
            if (count($ujiSeries) === 0) throw new \Exception('Data uji tidak tersedia untuk
skenario ini.');

            $dataTraining   = array_map(fn($o) => (int) $o->nilai, $trainingSeries);
            $dataUji        = array_map(fn($o) => (int) $o->nilai, $ujiSeries);
            $jumlahSimulasi = count($ujiSeries);

            // ====== PARAMETER ITERASI
            $targetMape = 10.0;  // <10 sangat baik, <20 baik, <50 cukup, >50 buruk
            $maxTry     = 1000;   // batas maksimal percobaan agar tidak lama
            // ================================================
            // Hitung error
            $calcErrorOnly = function (array $simulasiDetail, array $ujiSeries): array {
                $n = min(count($simulasiDetail), count($ujiSeries));
```

```
            if ($n === 0) return ['MAD' => 0, 'MSE' => 0, 'MAPE' => 0];
            $sumMAD = 0;
            $sumMSE = 0;
            $sumMAPE = 0;
            for ($i = 0; $i < $n; $i++) {
                $prediksi = (int) $simulasiDetail[$i]['permintaan'];
                $aktual   = (int) $ujiSeries[$i]->nilai;
                $mad  = abs($prediksi - $aktual);
                $mse  = ($prediksi - $aktual) ** 2;
                $mape = $aktual > 0 ? abs(($prediksi - $aktual) / $aktual) * 100 : 0;
                $sumMAD  += $mad;
                $sumMSE  += $mse;
                $sumMAPE += $mape;
            }

            return [
                'MAD'  => $sumMAD / $n,
                'MSE'  => $sumMSE / $n,
                'MAPE' => $sumMAPE / $n,
            ];
        };
        $bestHasil = null;
        $bestErr   = null;
        $bestTry   = 0;
        for ($t = 1; $t <= $maxTry; $t++) {
            // Penting: kirim [] agar bilangan acak pakai random_int() -> MAPE bisa berubah tiap
iterasi
            $hasilTmp = $this->runMonteCarloSimulation(
                $idObat,
                $skenario,
                $dataTraining,
                $jumlahSimulasi,
                [], // <-- ini yang membedakan: jangan pakai bilanganAcakExcel saat iterasi
                $trainingSeries
            );
            $errTmp = $calcErrorOnly($hasilTmp['simulasi_detail'], $ujiSeries);
            if ($bestErr === null || $errTmp['MAPE'] < $bestErr['MAPE']) {
                $bestHasil = $hasilTmp;
                $bestErr   = $errTmp;
                $bestTry   = $t;
            }
            if ($errTmp['MAPE'] < $targetMape) {
                break; // sudah masuk target
            }
        }
        if ($bestHasil === null) {
            throw new \Exception('Simulasi gagal menghasilkan hasil.');
        }
        // Simpan ke DB hanya 1x
        $hasilSimulasi = $bestHasil;

        $errorRingkas = $this->calculateErrorAndSaveMany(
            $idObat,
            $skenario,
            $hasilSimulasi['simulasi_detail'],
```

```
            $ujiSeries
        );
        // $this->saveMetrikEvaluasi($idObat, $skenario, $errorRingkas);
        $errorRingkas['kriteria_mape'] = $this->getMapeKriteria((float)
$errorRingkas['MAPE']);
        $errorRingkas['iterasi_terpilih'] = $bestTry;
        $rekap = [
            'obat'            => $obat->nama_obat,
            'id_obat'         => $idObat,
            'skenario'        => $skenario,
            'jumlah_simulasi' => $jumlahSimulasi,
            'periode_prediksi' => count($ujiSeries),
            'data_training'   => $dataTraining,
            'data_uji'        => $dataUji,
            'distribusi'      => $hasilSimulasi['distribusi'],
            'interval'        => $hasilSimulasi['interval'],
            'simulasi'        => $hasilSimulasi['simulasi_detail'],
            'statistik'       => $hasilSimulasi['statistik'],
            'error_ringkas'   => $errorRingkas,
        ];
        // simpan permanen
        session()->put('mc_rekap', $rekap);
        session()->put('mc_selected_obat_id', $idObat);
        session()->put('mc_selected_skenario', $skenario);
        return redirect()
            ->route('monte-carlo.index')
            ->with('success', 'Simulasi Monte Carlo berhasil dijalankan.');
        // ->with('rekap', $rekap)
        // ->with('selected_obat_id', $idObat);
    } catch (\Exception $e) {
        Log::error('Error simulasi Monte Carlo', [
            'error' => $e->getMessage(),
            'trace' => $e->getTraceAsString(),
        ]);
        return redirect()
            ->route('monte-carlo.index')
            ->with('error', 'Gagal menjalankan simulasi: ' . $e->getMessage());
    }
    return redirect()
        ->route('monte-carlo.index', [
            'obat_id'  => $idObat,
            'skenario' => $skenario,
        ])
        ->with('success', 'Simulasi Monte Carlo berhasil dijalankan.');
}
 private function splitTrainingAndTestData(string $skenario, array $series): array
 {
    $trainingSeries = [];
    $ujiSeries = [];
    if ($skenario === 'Skenario 1') {
        foreach ($series as $item) {
            if ($item->periode_tahun == 2024) {
                if ($item->periode_bulan == 12) $ujiSeries[] = $item;
                else $trainingSeries[] = $item;
            }
```

```
        }
    } elseif ($skenario === 'Skenario 2') {
        foreach ($series as $item) {
            if ($item->periode_tahun == 2021) $trainingSeries[] = $item;
            elseif ($item->periode_tahun == 2022) $ujiSeries[] = $item;
        }
    } elseif ($skenario === 'Skenario 3') {
        // FIX: Skenario 3 = bulanan
        foreach ($series as $item) {
            if ($item->periode_tahun >= 2021 && $item->periode_tahun <= 2023)
$trainingSeries[] = $item; // 36 bulan
            elseif ($item->periode_tahun == 2024) $ujiSeries[] = $item; // 12 bulan
        }
    }
    return [$trainingSeries, $ujiSeries];
}
private function getColumnName($namaObat)
{
    $namaObat = trim($namaObat);

    if (isset($this->obatMapping[$namaObat])) return $this->obatMapping[$namaObat];

    $columnName = strtolower($namaObat);
    $columnName = str_replace([' ', '-', '.'], '_', $columnName);
    $columnName = preg_replace('/[^a-z0-9_]/', '', $columnName);
    return $columnName;
}

private function getHistoricalSeries(string $skenario, string $columnName): array
{
    if ($skenario === 'Skenario 1') {
        if (!Schema::hasTable('tb_permintaan_obat_harian')) {
            throw new \Exception('Tabel tb_permintaan_obat_harian belum dibuat.');
        }
        $rows = DB::table('tb_permintaan_obat_harian')
            ->select('tanggal', DB::raw($columnName . ' as nilai'))
            ->whereYear('tanggal', 2024)
            ->whereNotNull($columnName)
            ->where($columnName, '>', 0)
            ->orderBy('tanggal')
            ->get();
        $series = [];
        foreach ($rows as $row) {
            $tgl = Carbon::parse($row->tanggal);
            $series[] = (object) [
                'periode_tahun' => $tgl->year,
                'periode_bulan' => $tgl->month,
                'label'         => $tgl->format('Y-m-d'),
                'nilai'         => (int) $row->nilai,
            ];
        }
        return $series;
    }
    if ($skenario === 'Skenario 2') {
        if (!Schema::hasTable('tb_periode_permintaan')) {
```

```
            throw new \Exception('Tabel tb_periode_permintaan belum dibuat.');
        }
        $rows = PeriodePermintaan::select(
            'periode_tahun',
            'periode_bulan',
            DB::raw($columnName . ' as nilai')
        )
            ->whereIn('periode_tahun', [2021, 2022])
            ->whereNotNull($columnName)
            ->where($columnName, '>', 0)
            ->orderBy('periode_tahun')
            ->orderBy('periode_bulan')
            ->get();
        $series = [];
        foreach ($rows as $row) {
            $label = sprintf('%04d-%02d', $row->periode_tahun, $row->periode_bulan);
            $series[] = (object) [
                'periode_tahun' => (int) $row->periode_tahun,
                'periode_bulan' => (int) $row->periode_bulan,
                'label'         => $label,
                'nilai'         => (int) $row->nilai,
            ];
        }
        return $series;
    }

    if ($skenario === 'Skenario 3') {
        if (!Schema::hasTable('tb_periode_permintaan')) {
            throw new \Exception('Tabel tb_periode_permintaan belum dibuat.');
        }
        // FIX UTAMA: Skenario 3
        $rows = PeriodePermintaan::select(
            'periode_tahun',
            'periode_bulan',
            DB::raw($columnName . ' as nilai')
        )
            ->whereBetween('periode_tahun', [2021, 2024])
            ->whereNotNull($columnName)
            ->where($columnName, '>', 0)
            ->orderBy('periode_tahun')
            ->orderBy('periode_bulan')
            ->get();
        $series = [];
        foreach ($rows as $row) {
            $label = sprintf('%04d-%02d', $row->periode_tahun, $row->periode_bulan);
            $series[] = (object) [
                'periode_tahun' => (int) $row->periode_tahun,
                'periode_bulan' => (int) $row->periode_bulan,
                'label'         => $label,
                'nilai'         => (int) $row->nilai,
            ];
        }
        return $series;
    }
    throw new \Exception('Skenario tidak valid: ' . $skenario);
```

```
    }
    private function normalizeText(string $s): string
    {
        $s = mb_strtolower($s);
        $s = preg_replace('/[^a-z0-9]/', '', $s);
        return $s ?? '';
    }
    private function readBilanganAcakFromExcel(?string $path, string $namaObat, int
$jumlahSimulasi, string $skenario): array
    {
        try {
            if (!$path || !file_exists($path)) return [];
            $spreadsheet = IOFactory::load($path);
            $sheet = $spreadsheet->getActiveSheet();
            $highestRow = $sheet->getHighestRow();
            $highestCol = $sheet->getHighestColumn();
            $highestColIndex = Coordinate::columnIndexFromString($highestCol);
            $maxRand = ($skenario === 'Skenario 1') ? 100 : 100;
            // cari header "bilangan acak"
            $headerRow = null;
            $headerCol = null;
            for ($r = 1; $r <= $highestRow; $r++) {
                for ($c = 1; $c <= $highestColIndex; $c++) {
                    $v = $sheet->getCellByColumnAndRow($c, $r)->getValue();
                    if (is_string($v) && stripos($v, 'bilangan acak') !== false) {
                        $headerRow = $r;
                        $headerCol = $c;
                        break 2;
                    }
                }
            }
            if (!$headerRow || !$headerCol) return [];
            // coba cari kolom obat di sekitar header
            $target = $this->normalizeText($namaObat);
            $colObat = null;
            for ($r = $headerRow; $r <= min($headerRow + 4, $highestRow); $r++) {
                for ($c = max(1, $headerCol - 15); $c <= min($headerCol + 15, $highestColIndex);
$c++) {
                    $v = $sheet->getCellByColumnAndRow($c, $r)->getValue();
                    if (!is_string($v) || trim($v) === '') continue;

                    if ($this->normalizeText($v) === $target) {
                        $colObat = $c;
                        break 2;
                    }
                }
            }
            if (!$colObat) $colObat = $headerCol;
            $rowStart = null;
            for ($r = 1; $r <= $highestRow; $r++) {
                for ($col = 1; $col <= 3; $col++) {
                    $v = $sheet->getCellByColumnAndRow($col, $r)->getValue();
                    if (is_string($v) && stripos($v, 'januari') !== false) {
                        $rowStart = $r;
                        break 2;
```

```
                }
            }
        }
        if (!$rowStart) return [];
        $nums = [];
        for ($r = $rowStart; $r < $rowStart + $jumlahSimulasi; $r++) {
            $v = $sheet->getCellByColumnAndRow($colObat, $r)->getCalculatedValue();
            if ($v === null || $v === '') continue;
            $n = (int) $v;
            if ($n < 0) $n = 0;
            if ($n > $maxRand) $n = $maxRand;
           $nums[] = $n;
            if (count($nums) >= $jumlahSimulasi) break;
        }
        return $nums;
    } catch (\Throwable $e) {
        Log::warning('Gagal baca bilangan acak dari excel', [
            'file'  => $path,
            'error' => $e->getMessage(),
        ]);
        return [];
    }
  }
 // mencari nilai interval bilangan acak skenario 1 2 dan 3
   private function buildIntervals(array $distribusi, int $maxRange): array //pemangggilan
semua skenario
   {
      $interval = [];
      $start = 0;
      foreach ($distribusi as $row) {
         if ($start > $maxRange) break;
         // bagian interval awal dan akhir
         $end = (int) floor($row['kumulatif'] * $maxRange); //rumus interval
         if ($end < 0) $end = 0;
         if ($end > $maxRange) $end = $maxRange;
         if ($end < $start) $end = $start;
         $interval[] = [  //hasil interval
           'nilai'          => $row['nilai'],
           'interval_awal'  => $start, //awal
           'interval_akhir' => $end,   //akhir
         ];
         $start = $end + 1; //lanjut ke interval berikutnya
      }
      if (!empty($interval)) {
         $interval[count($interval) - 1]['interval_akhir'] = $maxRange;
      }
      return $interval;
   }
   private function runMonteCarloSimulation(
      int $idObat,
      string $skenario,
      array $dataTraining,
      int $jumlahSimulasi,
      array $bilanganAcakExcel = [],
      array $trainingSeries = []
```

```
    ): array {
        // Skenario perhitungan 2 & 3
        if ($skenario === 'Skenario 2' || $skenario === 'Skenario 3') {
            $distribusi = [];
            $total = 0;
            // perhitungan frekuensi skenario 2 dan 3
            foreach ($trainingSeries as $row) {
                $total += (int) $row->nilai;
            }
//  Distribusi Probabilitas skenario 2 dan 3
            foreach ($trainingSeries as $row) {
                $nilai = (int) $row->nilai;
                $prob  = ($total > 0) ? ($nilai / $total) : 0; // distribusi probabilitas skenario 2 dan 3
                $distribusi[] = [ //bgian distribusi probabilitas
                    'nilai'       => $nilai,
                    'frekuensi'   => $nilai,
                    'probabilitas' => $prob,
                    'label'       => $row->label ?? null,
                ];
            }
// distribusi probabilitas kumulatif skenario 2 dan 3
            $kumulatif = 0.0;
            foreach ($distribusi as $i => $row) { //perhitungan distribusi prob.kumulatif
                $kumulatif += $row['probabilitas'];
                $distribusi[$i]['kumulatif'] = $kumulatif;
            }
// pembangkitan bil.acak
            $maxRand = 100; // S2 & S3
            $interval = $this->buildIntervals($distribusi, $maxRand);
            $simulasi = [];
            $hasil = [];
// perhitungan pembangkitan
            for ($i = 0; $i < $jumlahSimulasi; $i++) {
                $rand = $bilanganAcakExcel[$i] ?? random_int(0, $maxRand);
                if ($rand < 0) $rand = 0;
                if ($rand > $maxRand) $rand = $maxRand;
                $nilaiTerpilih = $interval[0]['nilai'];
                foreach ($interval as $row) {
                    if ($rand >= $row['interval_awal'] && $rand <= $row['interval_akhir']) {
                        $nilaiTerpilih = $row['nilai'];
                        break;
                    }
                }
        // hasil simulasi skenario 2 dan 3
                $simulasi[] = [
                    'periode'   => $i + 1,
                    'bil_acak'  => $rand,
                    'permintaan' => $nilaiTerpilih,
                ];
                $hasil[] = $nilaiTerpilih;
            }
            $statistik = [
                'jumlah_simulasi' => count($hasil),
                'rata_rata'       => count($hasil) ? array_sum($hasil) / count($hasil) : 0,
                'min'             => count($hasil) ? min($hasil) : 0,
```

```
            'max'             => count($hasil) ? max($hasil) : 0,
            'standar_deviasi' => $this->calculateStandardDeviation($hasil),
        ];
        return [
            'distribusi'      => $distribusi,
            'interval'        => $interval,
            'simulasi_detail' => $simulasi,
            'statistik'       => $statistik,
        ];
    }
    // Skenario 1: frekuensi berdasarkan kemunculan nilai
    $frekuensi = array_count_values($dataTraining);
    ksort($frekuensi);
    $total = array_sum($frekuensi);
    $distribusi = [];
    // distribusi probabilitas skenario 1
    foreach ($frekuensi as $nilai => $freq) {
        $distribusi[] = [ // bgian distribusi probabilitas skenario 1
            'nilai'        => (int) $nilai,
            'frekuensi'    => (int) $freq,
            'probabilitas' => ($total > 0) ? ($freq / $total) : 0.0, //ini perhitungan probabilitas
sknario 1
        ];
    }
  // Distribusi probabilitas kumulatif skenario 1
    $kumulatif = 0.0;
    foreach ($distribusi as $i => $row) { //perhitungan prob.kumulatif
        $kumulatif += $row['probabilitas'];
        $distribusi[$i]['kumulatif'] = $kumulatif;
    }
    // Setelah distribusi probabilitas dibuat (semua skenario)
    $maxRand = 100;  //skenario 1
    $interval = $this->buildIntervals($distribusi, $maxRand); //memanggil method
    $simulasi = [];
    $hasil = [];
    //pembangkitan bilangan acak (untuk Skenario 1, 2, dan 3)
    for ($i = 0; $i < $jumlahSimulasi; $i++) { //pembangkitan bil.acak
        $rand = $bilanganAcakExcel[$i] ?? random_int(0, $maxRand);
        if ($rand < 0) $rand = 0;
        if ($rand > $maxRand) $rand = $maxRand;
        //pencocokan interval
        $nilaiTerpilih = $interval[0]['nilai'];
        foreach ($interval as $row) {
            if ($rand >= $row['interval_awal'] && $rand <= $row['interval_akhir']) {
                $nilaiTerpilih = $row['nilai'];
                break;
            }
        }
            // hasil simulasi skenario 1
        $simulasi[] = [
            'periode'    => $i + 1,
            'bil_acak'   => $rand, //hasil bil.acak disimpan
            'permintaan' => $nilaiTerpilih,
        ];
        $hasil[] = $nilaiTerpilih;
```

```
        }
        $statistik = [
            'jumlah_simulasi' => count($hasil),
            'rata_rata'       => count($hasil) ? array_sum($hasil) / count($hasil) : 0,
            'min'             => count($hasil) ? min($hasil) : 0,
            'max'             => count($hasil) ? max($hasil) : 0,
            'standar_deviasi' => $this->calculateStandardDeviation($hasil),
        ];
        return [
            'distribusi'      => $distribusi,
            'interval'        => $interval,
            'simulasi_detail' => $simulasi,
            'statistik'       => $statistik,
        ];
    }
    //hasil perhitungan MAPE MSE DAN MAD
    private function calculateErrorAndSaveMany(
        int $idObat,
        string $skenario,
        array $simulasiDetail,
        array $ujiSeries
    ): array { // perhitungan error
        if (!Schema::hasTable('tb_hasil_error')) {
            return ['MAD' => 0, 'MSE' => 0, 'MAPE' => 0];
        }
        DB::table('tb_hasil_error')
            ->where('id_obat', $idObat)
            ->where('skenario', $skenario)
            ->delete();
        $n = min(count($simulasiDetail), count($ujiSeries));
        $sumMAD  = 0;
        $sumMSE  = 0;
        $sumMAPE = 0;
        for ($i = 0; $i < $n; $i++) {
            $prediksi = (int) $simulasiDetail[$i]['permintaan'];
            $aktual   = (int) $ujiSeries[$i]->nilai;
            $mad  = abs($prediksi - $aktual); //MAD
            $mse  = pow($prediksi - $aktual, 2); //MSE
            $mape = $aktual > 0 ? abs(($prediksi - $aktual) / $aktual) * 100 : 0; //MAPE
            $sumMAD  += $mad;
            $sumMSE  += $mse;
            $sumMAPE += $mape;
            $periodeTahun = property_exists($ujiSeries[$i], 'periode_tahun') ? $ujiSeries[$i]-
>periode_tahun : null;
            $periodeBulan = property_exists($ujiSeries[$i], 'periode_bulan') ? $ujiSeries[$i]-
>periode_bulan : null;
            DB::table('tb_hasil_error')->insert([
                'id_obat'       => $idObat,
                'skenario'      => $skenario,
                'data_prediksi' => $prediksi,
                'data_aktual'   => $aktual,
                'AD'            => $mad,
                'SE'            => $mse,
                'APE'           => $mape,
                'periode_tahun' => $periodeTahun,
```

```
            'periode_bulan' => $periodeBulan,
            'created_at'    => now(),
            'updated_at'    => now(),
        ]);
    }
    if ($n === 0) {
        return ['MAD' => 0, 'MSE' => 0, 'MAPE' => 0];
    }
    $mseAvg = $sumMSE / $n;
    return [
        'MAD'  => $sumMAD / $n,
        'MSE'  => $mseAvg,
        'RMSE' => sqrt($mseAvg),
        'MAPE' => $sumMAPE / $n,
    ];
}
private function calculateStandardDeviation(array $data): float
{
    $n = count($data);
    if ($n < 2) return 0.0;
    $mean = array_sum($data) / $n;
    $sumSquares = 0.0;
    foreach ($data as $value) {
        $sumSquares += pow($value - $mean, 2);
    }
    return sqrt($sumSquares / ($n - 1));
}
private function getMapeKriteria(float $mape): string
{
    if ($mape < 10)  return 'Kemampuan prediksi sangat baik';
    if ($mape < 20)  return 'Kemampuan prediksi baik';
    if ($mape < 50)  return 'Kemampuan prediksi cukup';
    return 'Kemampuan prediksi buruk';
}
private function calculateErrorOnly(array $simulasiDetail, array $ujiSeries): array
{
    $n = min(count($simulasiDetail), count($ujiSeries));
    if ($n === 0) return ['MAD' => 0, 'MSE' => 0, 'MAPE' => 0];
    $sumMAD = 0;
    $sumMSE = 0;
    $sumMAPE = 0;
    for ($i = 0; $i < $n; $i++) {
        $prediksi = (int) $simulasiDetail[$i]['permintaan'];
        $aktual   = (int) $ujiSeries[$i]->nilai;
        $mad  = abs($prediksi - $aktual);
        $mse  = ($prediksi - $aktual) ** 2;
        $mape = $aktual > 0 ? abs(($prediksi - $aktual) / $aktual) * 100 : 0;
        $sumMAD += $mad;
        $sumMSE += $mse;
        $sumMAPE += $mape;
    }
    return ['MAD' => $sumMAD / $n, 'MSE' => $sumMSE / $n, 'MAPE' => $sumMAPE /
$n];
}
public function analisisKomparatif()
```

```
{
    //  pastikan tb_metrik_evaluasi terisi dari tb_hasil_error
    $this->backfillMetrikEvaluasiDariError();
    $evaluations = MetrikEvaluasi::with('obat')
        ->orderBy('id_obat')
        ->orderBy('skenario')
        ->get()
        ->groupBy('id_obat');
    $statistics = [];
    foreach ($evaluations as $idObat => $scenarios) {
        $obat = Obat::find($idObat);
        $bestScenario = $scenarios->sortBy('MAPE')->first();
        $statistics[] = [
            'obat' => $obat ? $obat->nama_obat : '-',
            'obat_id' => $idObat,
            'scenarios' => $scenarios,
            'best_scenario' => $bestScenario,
            'avg_mape' => $scenarios->avg('MAPE'),
            'avg_mad' => $scenarios->avg('MAD'),
            'avg_mse' => $scenarios->avg('MSE'),
        ];
    }
    return view('hasil.analisis', compact('evaluations', 'statistics'));
}
}
<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
class MonteCarlo extends Model
{
    use HasFactory;
    protected $table = 'tb_monte_carlo';
    protected $primaryKey = 'id';
      // Sesuaikan dengan struktur tabel SEBENARNYA
    protected $fillable = [
        'id_obat',
        'skenario',
        'nilai_frekuensi',
        'frekuensi',  // TAMBAHKAN ini jika kolom sudah ditambahkan
        'distribusi_probabilitas',
        'distribusi_kumulatif',
        'interval_bil_acak_awal',
        'interval_bil_acak_akhir',
        'simulasi_permintaan'
        // HAPUS 'pembangkitan_acak' jika tidak ada di tabel
    ];
    protected $casts = [
        'nilai_frekuensi' => 'decimal:2',
        'distribusi_probabilitas' => 'decimal:15',
        'distribusi_kumulatif' => 'decimal:15',
        'frekuensi' => 'integer',
        'simulasi_permintaan' => 'integer'
    ];
    public function obat()
```

```
    {
        return $this->belongsTo(Obat::class, 'id_obat', 'id_obat');
    }
}
```

Source Code Halaman *Hasil Error*

```
<?php
namespace App\Http\Controllers;
use App\Models\HasilError;
use App\Models\MetrikEvaluasi;
use App\Models\MonteCarlo;
use App\Models\Obat;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
class HasilController extends Controller
{
    public function index(Request $request)
    {
        // daftar obat untuk dropdown
        $obats = DB::table('tb_obat')
            ->select('id_obat', 'nama_obat')
            ->orderBy('id_obat')
            ->get();
        // default: obat pertama, skenario 1
        $selectedObatId = $request->get('obat_id', $obats->first()->id_obat ?? null);
        $selectedSkenario = $request->get('skenario', 'Skenario 1');
        $metrik = null;
        $errorRows = collect();
        $ringkas = null;
        if (!$selectedObatId) {
            return view('hasil.index', compact(
                'obats',
                'selectedObatId',
                'selectedSkenario',
                'metrik',
                'errorRows',
                'ringkas'
            ));
        }
        // apakah tabel error punya kolom "tanggal"?
        $hasTanggal = DB::getSchemaBuilder()->hasColumn('tb_hasil_error', 'tanggal');
        if ($selectedSkenario === 'Skenario 1') {
            // Jika tb_hasil_error sudah ada kolom tanggal, pakai langsung
            if ($hasTanggal) {
                $errorRows = DB::table('tb_hasil_error as he')
                    ->where('he.id_obat', $selectedObatId)
                    ->where('he.skenario', $selectedSkenario)
                    ->select('he.*', DB::raw('he.tanggal as tanggal_tampil'))
                    ->orderBy('he.tanggal', 'asc')
                    ->get();
            } else {
                /**
                 * Jika belum ada kolom tanggal:
                 * - Pasangkan urutan baris tb_hasil_error dengan urutan tanggal di
```

```
tb_permintaan_obat_harian
             * - Kunci pasangan: (periode_tahun, periode_bulan, row_number)
             */
            // Subquery error: beri nomor urut per (tahun, bulan)
            $he = DB::table('tb_hasil_error as he')
               ->selectRaw("
                  he.*,
                  ROW_NUMBER() OVER (
                     PARTITION BY he.periode_tahun, he.periode_bulan
                     ORDER BY he.id
                  ) as rn
               ")
               ->where('he.id_obat', $selectedObatId)
               ->where('he.skenario', $selectedSkenario);
            // Subquery harian: beri nomor urut per (tahun, bulan)
            $h = DB::table('tb_permintaan_obat_harian as h')
               ->selectRaw("
                  h.tanggal,
                  YEAR(h.tanggal)  as periode_tahun,
                  MONTH(h.tanggal) as periode_bulan,
                  ROW_NUMBER() OVER (
                     PARTITION BY YEAR(h.tanggal), MONTH(h.tanggal)
                     ORDER BY h.tanggal
                  ) as rn
               ");
            $errorRows = DB::query()
               ->fromSub($he, 'he')
               ->joinSub($h, 'h', function ($join) {
                  $join->on('he.periode_tahun', '=', 'h.periode_tahun')
                     ->on('he.periode_bulan', '=', 'h.periode_bulan')
                     ->on('he.rn', '=', 'h.rn');
               })
               ->select('he.*', DB::raw('h.tanggal as tanggal_tampil'))
               ->orderBy('h.tanggal', 'asc')
               ->get();
         }
      } else {
         // Skenario 2/3: tampilkan periode (tahun-bulan) sebagai tanggal_tampil
         $errorRows = DB::table('tb_hasil_error as he')
            ->where('he.id_obat', $selectedObatId)
            ->where('he.skenario', $selectedSkenario)
            ->select(
               'he.*',
               DB::raw("CONCAT(he.periode_tahun, '-', LPAD(he.periode_bulan, 2, '0')) as
tanggal_tampil")
            )
            ->orderBy('he.periode_tahun', 'asc')
            ->orderBy('he.periode_bulan', 'asc')
            ->get();
      }
      $this->syncMetrikEvaluasiFromErrorRows($selectedObatId, $selectedSkenario,
$errorRows);
      $metrik = $this->getMetrikWithNamaObat($selectedObatId, $selectedSkenario);

      $ringkas = $this->buildRingkasanMetrik($metrik, $errorRows);
```

```
    return view('hasil.index', compact(
        'obats',
        'selectedObatId',
        'selectedSkenario',
        'metrik',
        'errorRows',
        'ringkas'
    ));
}
public function show(Request $request)
{
    // NOTE: route lama kamu pakai id_obat, skenario
    $request->validate([
        'id_obat' => 'required|exists:tb_obat,id_obat',
        'skenario' => 'required|in:Skenario 1,Skenario 2,Skenario 3',
    ]);

    $idObat = (int) $request->id_obat;
    $skenario = (string) $request->skenario;

    $obats = Obat::all();
    $obat = Obat::findOrFail($idObat);

    $errorRows = $this->getErrorRowsWithTanggalTampil($idObat, $skenario);
    $this->syncMetrikEvaluasiFromErrorRows($idObat, $skenario, $errorRows);
    $metrik = $this->getMetrikWithNamaObat($idObat, $skenario);
    $ringkas = $this->buildRingkasanMetrik($metrik, $errorRows);
    // kalau kamu masih pakai results untuk view tertentu, tetap disediakan
    $results = $this->getPredictionResults($idObat, $skenario);
    return view('hasil.index', compact(
        'obats',
        'obat',
        'results',
        'idObat',
        'skenario',
        'metrik',
        'errorRows',
        'ringkas'
    ) + [
        'selectedObatId' => $idObat,
        'selectedSkenario' => $skenario,
    ]);
}
public function detailSimulasi($id_obat, $skenario)
{
    $obat = Obat::findOrFail($id_obat);
    $monteCarloData = MonteCarlo::where('id_obat', $id_obat)
        ->where('skenario', $skenario)
        ->whereNotNull('frekuensi')
        ->orderBy('id')
        ->get();
    $errorData = HasilError::where('id_obat', $id_obat)
        ->where('skenario', $skenario)
        ->orderBy('created_at', 'desc')
```

```
            ->limit(10)
            ->get();
        $metrics = MetrikEvaluasi::where('id_obat', $id_obat)
            ->where('skenario', $skenario)
            ->first();
        $latestPrediction = MonteCarlo::where('id_obat', $id_obat)
            ->where('skenario', $skenario)
            ->whereNotNull('simulasi_permintaan')
            ->orderBy('created_at', 'desc')
            ->first();
        $stats = [
            'total_distribusi' => $monteCarloData->count(),
            'prediksi_terakhir' => $latestPrediction ? $latestPrediction->simulasi_permintaan : 0,
            'total_error' => $errorData->count(),
            'kategori_akurasi' => $metrics ? $metrics->kategori_akurasi : 'Belum dievaluasi',
        ];
        return view('hasil.simulasi', compact(
            'obat',
            'skenario',
            'monteCarloData',
            'errorData',
            'metrics',
            'stats',
            'latestPrediction'
        ));
    }
    public function analisisKomparatif()
    {
        // Auto-backfill dari tb_hasil_error -> tb_metrik_evaluasi (MSE tetap AVG(SE), bukan
RMSE)
        $this->backfillMetrikEvaluasiDariError();
        $evaluations = MetrikEvaluasi::with('obat')
            ->orderBy('id_obat')
            ->orderBy('skenario')
            ->get()
            ->groupBy('id_obat');
        $statistics = [];
        foreach ($evaluations as $idObat => $scenarios) {
            $obat = Obat::find($idObat);
            // keyBy skenario (trim biar aman dari spasi)
            $bySkenario = $scenarios->keyBy(fn ($x) => trim((string)$x->skenario));
            $s1 = $bySkenario->get('Skenario 1');
            $s2 = $bySkenario->get('Skenario 2');
            $s3 = $bySkenario->get('Skenario 3');
            $best = $scenarios->sortBy('MAPE')->first();
            $statistics[] = [
                'obat' => $obat ? $obat->nama_obat : '-',
                'obat_id' => $idObat,
                's1' => $s1,
                's2' => $s2,
                's3' => $s3,
                'best' => $best,
            ];
        }
        return view('hasil.analisis', compact('statistics'));
```

```
    }
    public function generateReport(Request $request)
    {
        $request->validate([
            'format' => 'nullable|in:pdf,excel',
            'id_obat' => 'nullable|exists:tb_obat,id_obat',
        ]);
        $format = $request->format ?? 'pdf';
        $idObat = $request->id_obat;
        $query = DB::table('tb_metrik_evaluasi as me')
            ->join('tb_obat as o', 'me.id_obat', '=', 'o.id_obat')
            ->select('o.nama_obat', 'me.*');
        if ($idObat) {
            $query->where('me.id_obat', $idObat);
        }
        $results = $query
            ->orderBy('o.nama_obat')
            ->orderBy('me.skenario')
            ->get();
        $stats = [
            'total_obat' => $results->unique('id_obat')->count(),
            'total_evaluasi' => $results->count(),
            'rata_mape' => $results->avg('MAPE'),
            'terbaik_mape' => $results->sortBy('MAPE')->first(),
            'terburuk_mape' => $results->sortByDesc('MAPE')->first(),
        ];
        return view('hasil.laporan', compact('results', 'stats', 'format', 'idObat'));
    }
    private function getMetrikWithNamaObat(int $idObat, string $skenario)
    {
        return DB::table('tb_metrik_evaluasi as me')
            ->join('tb_obat as o', 'me.id_obat', '=', 'o.id_obat')
            ->select('me.*', 'o.nama_obat')
            ->where('me.id_obat', $idObat)
            ->where('me.skenario', $skenario)
            ->first();
    }
    private function getErrorRowsWithTanggalTampil(int $idObat, string $skenario)
    {
        $hasTanggal = DB::getSchemaBuilder()->hasColumn('tb_hasil_error', 'tanggal');
        if ($skenario === 'Skenario 1') {
            if ($hasTanggal) {
                return DB::table('tb_hasil_error as he')
                    ->where('he.id_obat', $idObat)
                    ->where('he.skenario', $skenario)
                    ->select('he.*', DB::raw('he.tanggal as tanggal_tampil'))
                    ->orderBy('he.tanggal', 'asc')
                    ->get();
            }
            // fallback kalau tidak ada kolom tanggal: tampilkan created_at (lebih aman &
konsisten)
            return DB::table('tb_hasil_error as he')
                ->where('he.id_obat', $idObat)
                ->where('he.skenario', $skenario)
                ->select('he.*', DB::raw('DATE(he.created_at) as tanggal_tampil'))
```

```
            ->orderBy('he.created_at', 'asc')
            ->get();
        }
        if ($skenario === 'Skenario 3') {
            // Tahunan: tampilkan tahun saja
            return DB::table('tb_hasil_error as he')
                ->where('he.id_obat', $idObat)
                ->where('he.skenario', $skenario)
                ->select('he.*', DB::raw("CAST(he.periode_tahun AS CHAR) as tanggal_tampil"))
                ->orderBy('he.periode_tahun', 'asc')
                ->get();
        }
        // Skenario 2 (bulanan): YYYY-MM
        return DB::table('tb_hasil_error as he')
            ->where('he.id_obat', $idObat)
            ->where('he.skenario', $skenario)
            ->select('he.*', DB::raw("CONCAT(he.periode_tahun, '-', LPAD(he.periode_bulan, 2,
'0')) as tanggal_tampil"))
            ->orderBy('he.periode_tahun', 'asc')
            ->orderBy('he.periode_bulan', 'asc')
            ->get();
    }
    private function syncMetrikEvaluasiFromErrorRows(int $idObat, string $skenario,
$errorRows): void
    {
        if (!$errorRows || $errorRows->count() === 0) return;
        $mad = (float) $errorRows->avg('AD');
        $mse = (float) $errorRows->avg('SE');   // MSE = AVG(SE)
        $mape = (float) $errorRows->avg('APE');
        DB::table('tb_metrik_evaluasi')->updateOrInsert(
            ['id_obat' => $idObat, 'skenario' => $skenario],
            [
                'MAD' => $mad,
                'MSE' => $mse,
                'MAPE' => $mape,
                'kategori_akurasi' => $this->getMapeKriteria($mape),
                'updated_at' => now(),
                'created_at' => now(),
            ]
        );
    }
    private function backfillMetrikEvaluasiDariError(): void
    {
        $rows = DB::table('tb_hasil_error')
            ->selectRaw('id_obat, skenario, AVG(AD) as MAD, AVG(SE) as MSE, AVG(APE) as
MAPE')
            ->groupBy('id_obat', 'skenario')
            ->get();
        foreach ($rows as $r) {
            $mape = (float) ($r->MAPE ?? 0);
            DB::table('tb_metrik_evaluasi')->updateOrInsert(
                ['id_obat' => $r->id_obat, 'skenario' => $r->skenario],
                [
                    'MAD' => (float) ($r->MAD ?? 0),
                    'MSE' => (float) ($r->MSE ?? 0), // jangan sqrt
```

```
                'MAPE' => $mape,
                'kategori_akurasi' => $this->getMapeKriteria($mape),
                'updated_at' => now(),
                'created_at' => now(),
            ]
        );
    }
}
private function buildRingkasanMetrik($metrik, $errorRows): array
{
    $avgMAD  = $errorRows->count() ? (float) $errorRows->avg('AD') : 0.0;
    $avgMSE  = $errorRows->count() ? (float) $errorRows->avg('SE') : 0.0;
    $avgMAPE = $errorRows->count() ? (float) $errorRows->avg('APE') : 0.0;
    // Aman kalau $metrik null
    $mad  = ($metrik && $metrik->MAD !== null) ? (float) $metrik->MAD : $avgMAD;
    $mse  = ($metrik && $metrik->MSE !== null) ? (float) $metrik->MSE : $avgMSE;
    $mape = ($metrik && $metrik->MAPE !== null) ? (float) $metrik->MAPE : $avgMAPE;
    // RMSE dihitung dari MSE (untuk tampilan)
    $rmse = $mse > 0 ? sqrt($mse) : 0.0;
    $kategori = ($metrik && !empty($metrik->kategori_akurasi))
        ? $metrik->kategori_akurasi
        : $this->getMapeKriteria($mape);
    return [
        'mad' => $mad,
        'mse' => $mse,
        'rmse' => $rmse,
        'mape' => $mape,
        'kriteria' => $kategori,
    ];
}
private function getMapeKriteria(float $mape): string
{
    if ($mape <= 10) return 'Sangat Baik';
    if ($mape <= 20) return 'Baik';
    if ($mape <= 50) return 'Cukup';
    return 'Buruk';
}
private function getPredictionResults($idObat, $skenario)
{
    return DB::table('tb_monte_carlo as mc')
        ->join('tb_obat as o', 'mc.id_obat', '=', 'o.id_obat')
        ->leftJoin('tb_hasil_error as he', function ($join) use ($idObat, $skenario) {
            $join->on('mc.id_obat', '=', 'he.id_obat')
                ->on('mc.skenario', '=', 'he.skenario')
                ->where('he.id_obat', $idObat)
                ->where('he.skenario', $skenario);
        })
        ->leftJoin('tb_metrik_evaluasi as me', function ($join) use ($idObat, $skenario) {
            $join->on('mc.id_obat', '=', 'me.id_obat')
                ->on('mc.skenario', '=', 'me.skenario')
                ->where('me.id_obat', $idObat)
                ->where('me.skenario', $skenario);
        })
        ->where('mc.id_obat', $idObat)
        ->where('mc.skenario', $skenario)
```

```
            ->whereNotNull('mc.simulasi_permintaan')
            ->select(
                'o.nama_obat',
                'mc.skenario',
                'mc.simulasi_permintaan as prediksi',
                'he.data_aktual',
                'he.AD',
                'he.SE',
                'he.APE',
                'me.MAPE',
                'me.MSE',
                'me.MAD',
                'me.kategori_akurasi',
                'mc.created_at as tanggal_prediksi'
            )
            ->orderBy('mc.created_at', 'desc')
            ->get();
    }
}
<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
class HasilError extends Model
{
    use HasFactory;
    protected $table = 'tb_hasil_error';
    protected $primaryKey = 'id';
        public $timestamps = true;
    const CREATED_AT = 'created_at';
    const UPDATED_AT = null;
        protected $fillable = [
        'id_obat',
        'skenario',
        'data_prediksi',
        'data_aktual',
        'AD',
        'SE',
        'APE',
        'simulasi',
        'hitung_error',
        'created_at'
    ];
        protected $casts = [
        'AD' => 'decimal:2',
        'SE' => 'decimal:2',
        'APE' => 'decimal:4',
        'hitung_error' => 'decimal:4'
    ];
        public function obat()
    {
        return $this->belongsTo(Obat::class, 'id_obat', 'id_obat');
    }
}
```

Source Code Halaman Tentang Kami

```
@extends('layouts.app')
@section('title', 'Tentang Kami'
@section('content')
<div class="mc-container">
   {{-- BARIS ATAS: TITLE --}}
   <div class="mc-top-bar">
      <div class="mc-top-left">
         <h5 class="mb-0" style="font-size:14px;font-weight:700;">
            TENTANG KAMI
         </h5>
      </div>
   </div>
   {{-- BLOK PROFIL APLIKASI --}}
   <div class="mc-block">
      <div class="mc-block-header">
         <span>Profil Aplikasi</span>
      </div>
      <div class="mt-2" style="font-size:12px;line-height:1.6;">
         <p>
            <strong>{{ $appInfo['nama_aplikasi'] ?? 'Sistem Prediksi Permintaan Obat Jantung'
}}</strong>
            merupakan aplikasi berbasis web yang digunakan untuk
            memprediksi kebutuhan obat jantung pada
            <strong>{{ $appInfo['rumah_sakit'] ?? 'RSUD Ibnu Sina' }}</strong>.
         </p>
         <p>
            Aplikasi ini menggunakan metode
            <strong>{{ $appInfo['metode'] ?? 'Monte Carlo Frekuensi–Range' }}</strong>
            untuk membangkitkan permintaan obat secara acak berdasarkan
            distribusi historis. Hasil prediksi kemudian dievaluasi dengan
            metrik <strong>MAD</strong>, <strong>MSE</strong>, dan
<strong>MAPE</strong>.
         </p>
         <p class="mb-0">
            Versi Aplikasi: <strong>{{ $appInfo['versi'] ?? '1.0.0' }}</strong>
         </p>
      </div>
   </div>
   {{-- BLOK TUJUAN DAN MANFAAT --}}
   <div class="mc-block">
      <div class="mc-block-header">
         <span>Tujuan &amp;amp; Manfaat</span>
      </div>
      <ul style="font-size:12px;margin-top:6px;padding-left:18px;">
         <li>Membantu depo farmasi memprediksi kebutuhan obat jantung setiap bulan.</li>
         <li>Mengurangi risiko kekurangan stok (stockout) maupun kelebihan stok.</li>
         <li>Menyediakan dashboard sederhana untuk mengelola dataset harian dan
bulanan.</li>
         <li>Memberikan evaluasi akurasi prediksi menggunakan MAD, MSE, dan MAPE.</li>
      </ul>
   </div>
   {{-- BLOK FITUR UTAMA --}}
   <div class="mc-block">
      <div class="mc-block-header">
```

```
        <span>Fitur Utama</span>
      </div>
      <div class="mt-2" style="font-size:12px;">
        <ol style="padding-left:18px;">
          <li>
            <strong>Data Set</strong>
            Mengelola data permintaan obat jantung:
            <ul style="padding-left:16px;margin-top:2px;">
              <li>Input manual data harian.</li>
              <li>Import data harian dari file Excel.</li>
              <li>Akumulasi otomatis menjadi data bulanan.</li>
            </ul>
          </li>
          <li class="mt-1">
            <strong>Simulasi Monte Carlo</strong>
            Memilih obat, skenario, jumlah simulasi, dan periode prediksi,
            kemudian menjalankan simulasi Monte Carlo frekuensi–range.
          </li>
          <li class="mt-1">
            <strong>Hasil Error / Rekap</strong>
            Menampilkan ringkasan metrik MAD, MSE, dan MAPE
            serta tabel error per periode untuk setiap kombinasi obat dan skenario.
          </li>
          <li class="mt-1">
            <strong>Analisis & Laporan</strong> (opsional)
            Menyediakan analisis komparatif antar skenario dan laporan
            rekapitulasi metrik evaluasi.
          </li>
          <li class="mt-1">
            <strong>Reset Data</strong>
            Menghapus seluruh dataset, hasil simulasi, dan log proses
            untuk mengulang percobaan dari awal.
          </li>
        </ol>
      </div>
    </div>
    {{-- BLOK ALUR PENGGUNAAN SINGKAT --}}
    <div class="mc-block">
      <div class="mc-block-header">
        <span>Alur Penggunaan Singkat</span>
      </div>
      <div style="font-size:12px;margin-top:6px;">
        <ol style="padding-left:18px;">
          <li>
            Buka menu <strong>Data Set</strong> untuk:
            <ul style="padding-left:16px;margin-top:2px;">
              <li>Import data harian dari Excel, atau</li>
              <li>Input manual data harian per tanggal.</li>
            </ul>
          </li>
          <li>
            Sistem otomatis mengakumulasi data harian menjadi
            <strong>data bulanan</strong> di tabel <code>tb_periode_permintaan</code>.
          </li>
          <li>
```

```
                Buka menu <strong>Monte Carlo</strong>, pilih obat dan skenario,
                lalu tentukan:
                <ul style="padding-left:16px;margin-top:2px;">
                  <li><em>Jumlah simulasi</em> (misal: 1000)</li>
                  <li><em>Periode prediksi</em> (misal: 1 bulan ke depan)</li>
                </ul>
                lalu klik <strong>Hitung</strong>.
              </li>
              <li>
                Hasil distribusi, interval bilangan acak, dan prediksi akan tampil
                di tabel Monte Carlo. Sekaligus tersimpan ke database.
              </li>
              <li>
                Buka menu <strong>Hasil Error</strong> untuk melihat nilai
                MAD, MSE, MAPE, dan kategori akurasi per obat &amp;amp; skenario.
              </li>
            </ol>
          </div>
        </div>
        {{-- TOMBOL KEMBALI --}}
        <div class="mc-bottom-btn">
          <a href="{{ route('dashboard') }}" class="mc-btn mc-btn-outline">
            Kembali ke Dashboard
          </a>
        </div>
      </div>
      @endsection
      {{-- STYLE SEDERHANA (pakai mc-* sama seperti halaman Monte Carlo) --}}
      <style>
        .mc-container {
          background: var(--light-bg, #f8fafc);
          color: var(--dark-text, #2d3748);
          padding: 16px 20px 30px 20px;
          border-radius: 12px;
          border: 1px solid var(--border-color, #e2e8f0);
          box-shadow: 0 4px 12px rgba(0,0,0,0.04);
        }
        .mc-top-bar {
          display: flex;
          justify-content: space-between;
          align-items: center;
          margin-bottom: 8px;
        }
        .mc-block {
          border: 1px solid var(--border-color, #e2e8f0);
          padding: 8px 10px 10px 10px;
          margin-bottom: 12px;
          background: #ffffff;
          border-radius: 10px;
        }
        .mc-block-header span {
          font-size: 13px;
          font-weight: 600;
          color: var(--primary-color, #1a5fb4);
        }
```

```
    .mc-btn {
        border: 1px solid var(--secondary-color, #0d6efd);
        background: white;
        color: var(--secondary-color, #0d6efd);
        padding: 4px 16px;
        font-size: 12px;
        text-decoration: none;
        border-radius: 6px;
        transition: all 0.2s ease;
    }
    .mc-btn-outline {
        background: transparent;
    }
    .mc-btn:hover {
        background: var(--secondary-color, #0d6efd);
        color: #fff;
        text-decoration: none;
        box-shadow: 0 2px 6px rgba(13,110,253,0.3);
    }
    .mc-bottom-btn {
        margin-top: 8px;
        display: flex;
        justify-content: flex-start;
        gap: 8px;
    }
</style>
```