

Laporan Modul 2: Laravel Fundamentals

Mata Kuliah: Workshop Web Lanjut

Nama: Maila Aziza **NIM:** 2024573010024 **Kelas:** TI 2C

Abstrak

Laporan ini bertujuan untuk membahas tentang fondasi utama dalam membangun aplikasi laravel. Laporan ini untuk memahami routing, middleware, penanganan request/response, controller, view, dan Blade templating sangat penting untuk mulai mengembangkan dengan Laravel secara efektif.

1. Dasar Teori

- Apa itu MVC (Model, View, Controller). MVC adalah singkatan dari Model-View-Controller, sebuah pola arsitektur perangkat lunak yang digunakan untuk memisahkan concern dalam aplikasi web: a. Model: Mempresentasikan data dan logika bisnis. Dalam Laravel, biasanya ditangani oleh Eloquent models yang berinteraksi dengan database. b. View: Menangani lapisan presentasi, yaitu HTML yang ditampilkan kepada pengguna. Laravel menggunakan Blade sebagai template engine untuk view. c. Controller: Mengatur alur antara model dan view. Controller menerima input dari pengguna melalui route, memprosesnya, dan mengembalikan view atau resppons yang sesuai.
- Konsep Routing di Laravel. Routing di laravel mendefinisikan bagaimana aplikasi merespons permintaan yang masuk. Route di larael didefinisikan di file routes/web.php untuk permintaan web dan di routes/api.php untuk route API. Routing memetakan URI ke aksi controller atau closure.
- Fungsi Middleware. Middleware menyediakan mekanisme untuk memfilter permintaan HTTP yang masuk ke aplikasi. Middleware bertindak sebagai lapisan yang berada di antara siklus request dan response.
- Bagaimana cara Laravel menangani Request dan Response. Laravel mempermudah dalam menangani HTTP request dan menghasilkan response. Kita dapat mengakses data request menggunakan objek Request dan juga dapat dengan mudah mengakses parameter query, data form, dan payload JSON.
- Peran Controller dan View. Controller digunakan untuk mengatur logika aplikasi. File controller berada di dalam app/Http/Controllers. View disimpan di dalam resources/views. Contoh file view sederhana bisa dibuat di resources/views/home.blade.php.
- Fungsi Blade Templating Engine. Blade merupakan template engine yang disediakan oleh laravel. Contoh penggunaan template engine blade:

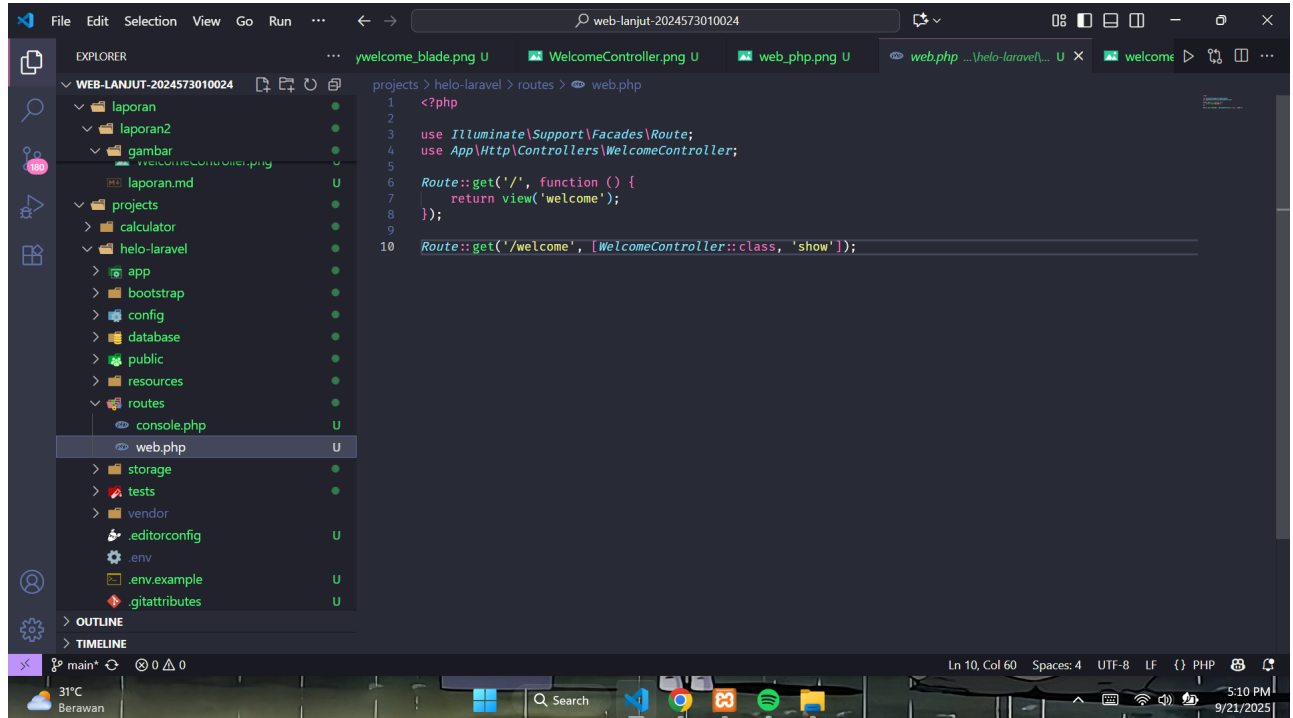
```
<h1>
    Hello, {{$name}}
</h1>
```

2. Langkah-Langkah Praktikum

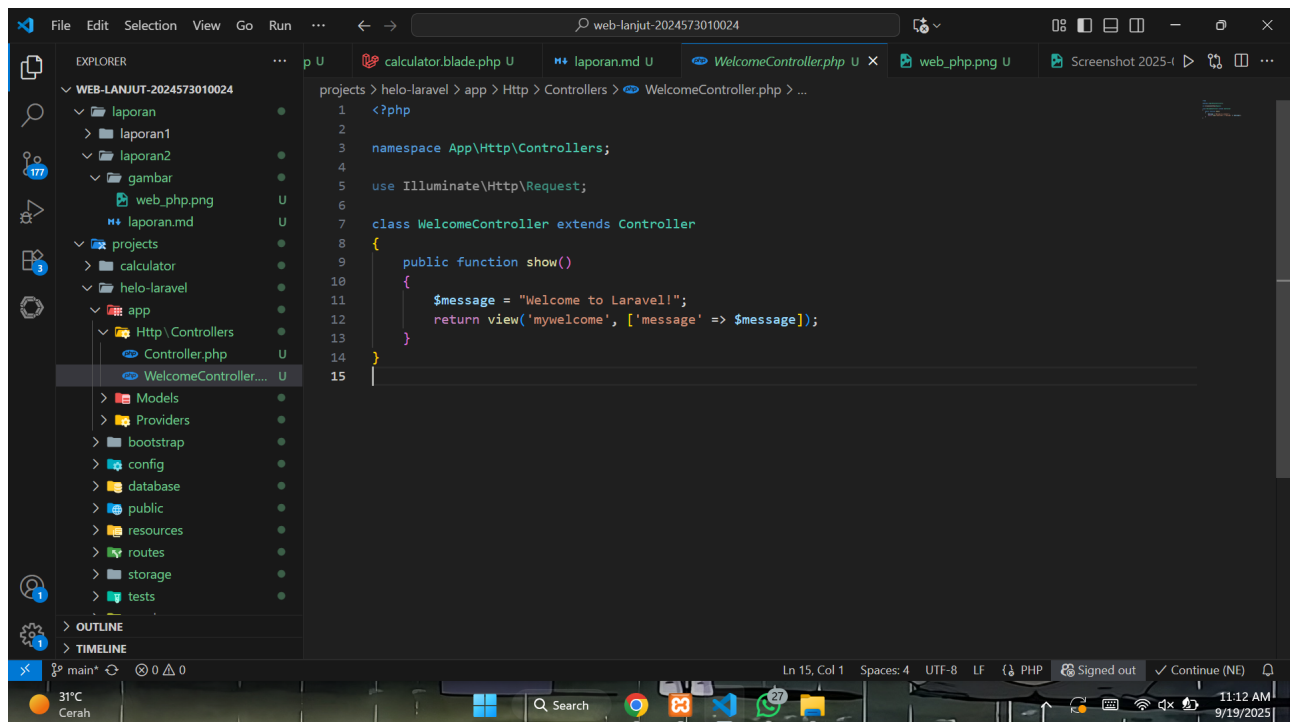
Tuliskan langkah-langkah yang sudah dilakukan, sertakan potongan kode dan screenshot hasil.

2.1 Praktikum 1 – Route, Controller, dan Blade View

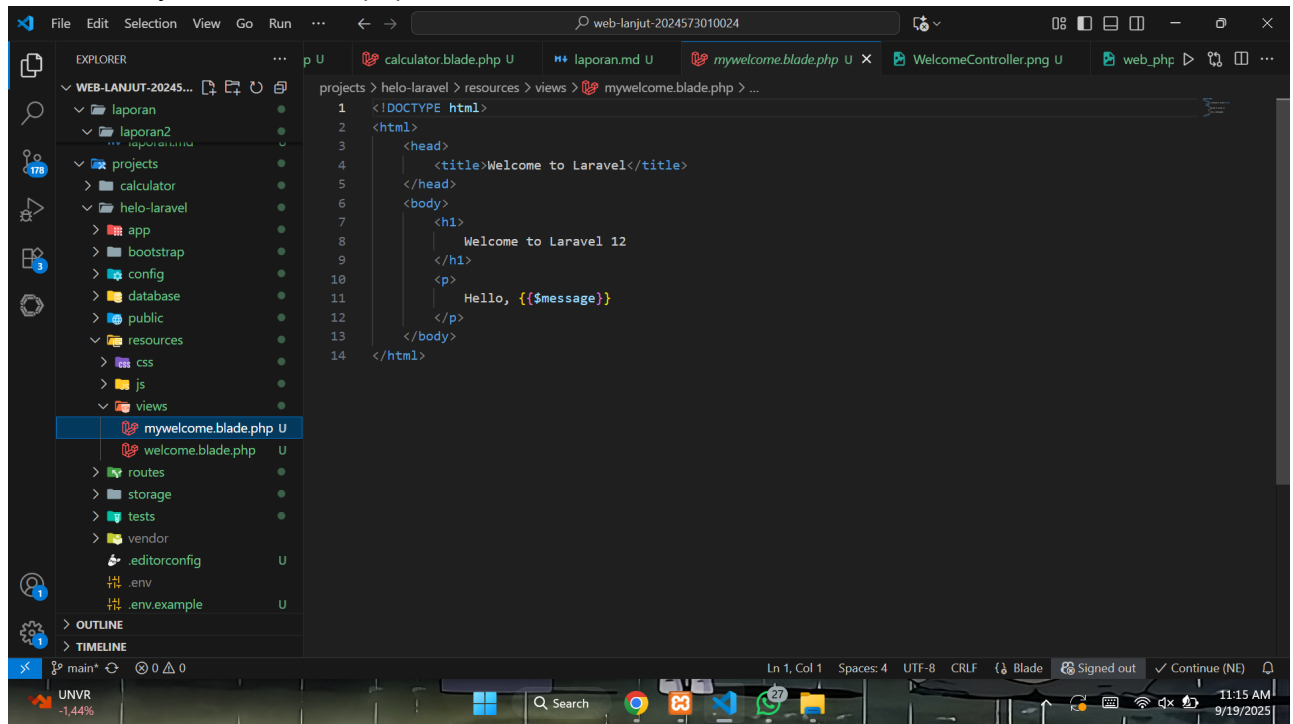
- Tambahkan route pada routes/web.php.



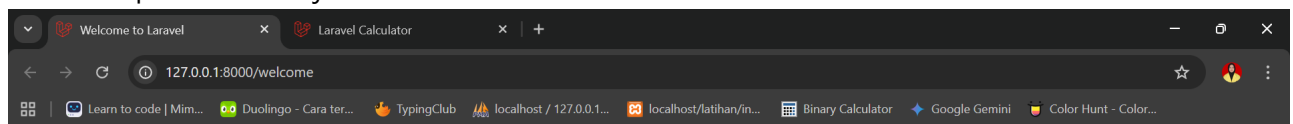
- Buat controller WelcomeController.



- Buat view mywelcome.blade.php.



- Jalankan aplikasi dan tunjukkan hasil di browser. Screenshot Hasil:



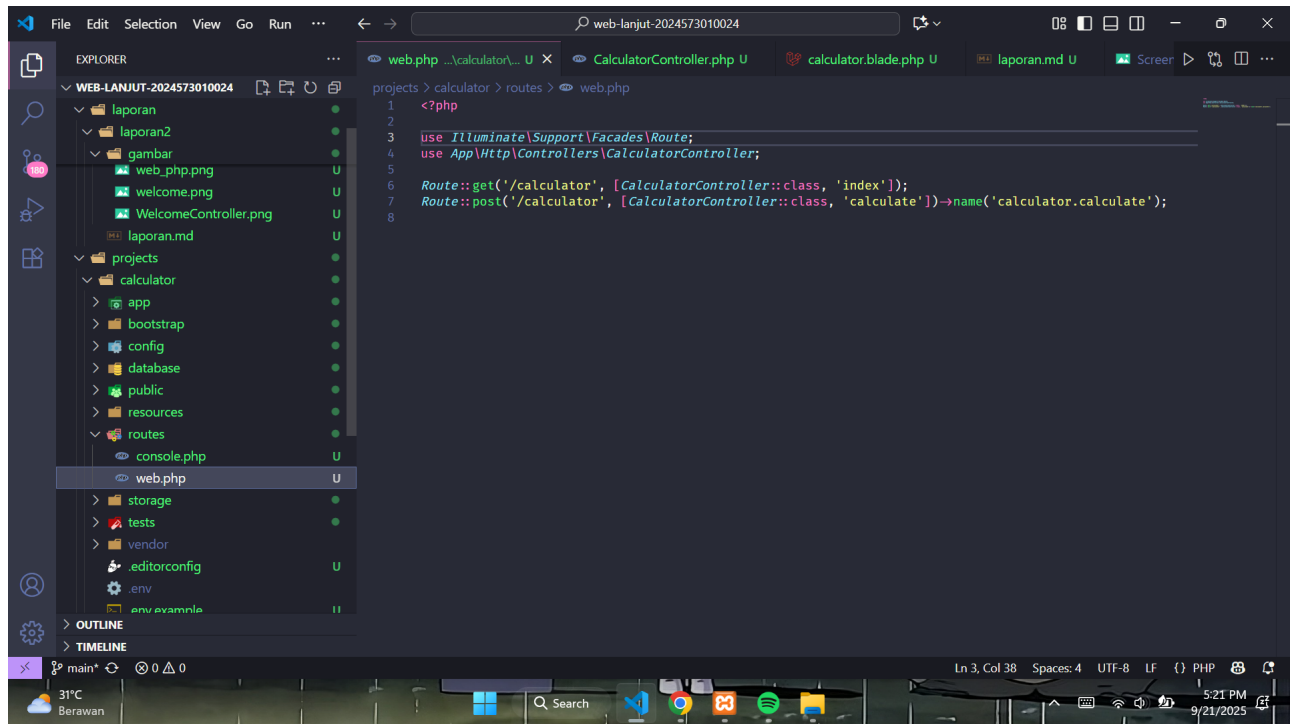
Welcome to Laravel 12

Hello, Welcome to Laravel!



2.2 Praktikum 2 – Membuat Aplikasi Sederhana "Calculator"

- Tambahkan route untuk kalkulator.

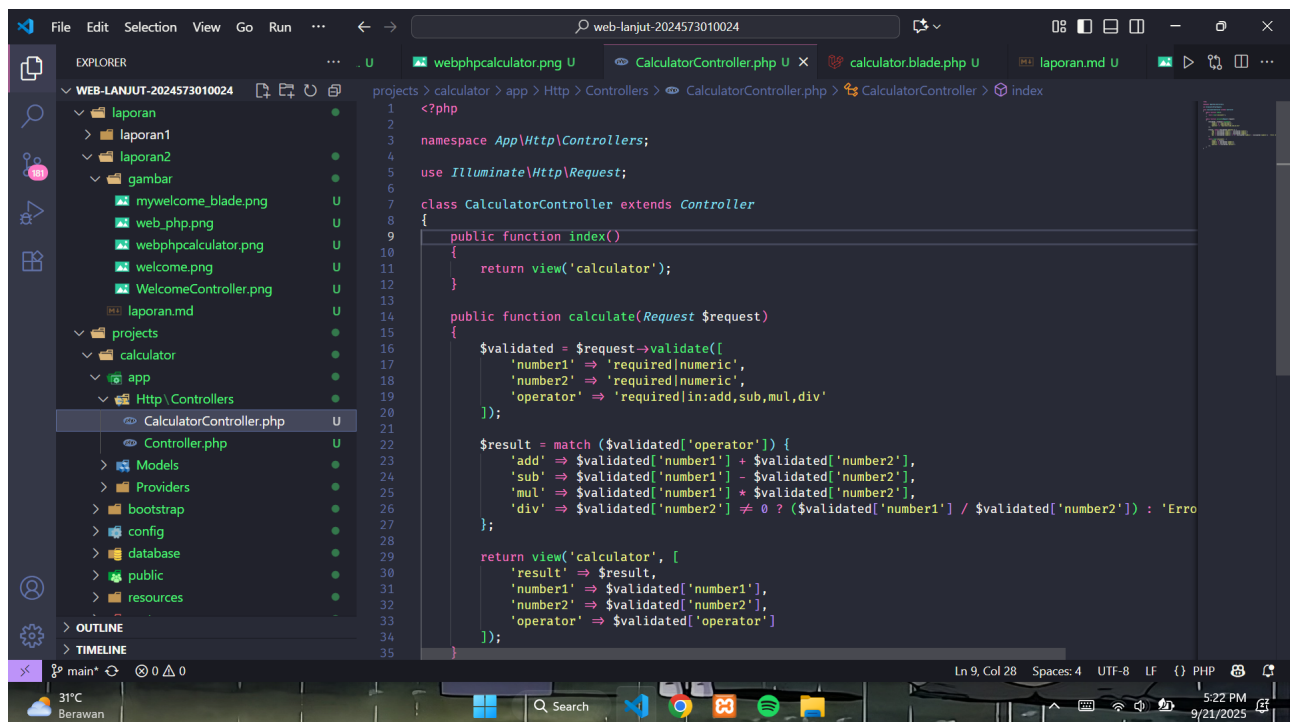


The screenshot shows the Visual Studio Code editor with the Explorer sidebar on the left displaying the project structure. The file `web.php` under the `routes` directory is selected. The main editor displays the content of `web.php`, which includes the following PHP code:

```
1 <?php
2
3 use Illuminate\Support\Facades\Route;
4 use App\Http\Controllers\CalculatorController;
5
6 Route::get('/calculator', [CalculatorController::class, 'index']);
7 Route::post('/calculator', [CalculatorController::class, 'calculate'])->name('calculator.calculate');
```

The status bar at the bottom indicates the current position is Line 3, Column 38, with 4 spaces, UTF-8 encoding, LF line endings, and PHP syntax highlighting.

- Buat controller CalculatorController.



The screenshot shows the Visual Studio Code editor with the Explorer sidebar on the left. The file `CalculatorController.php` under the `Http/Controllers` directory is selected. The main editor displays the content of `CalculatorController.php`, which includes the following PHP code:

```
1 <?php
2
3 namespace App\Http\Controllers;
4 use Illuminate\Http\Request;
5
6 class CalculatorController extends Controller
7 {
8     public function index()
9     {
10         return view('calculator');
11     }
12
13     public function calculate(Request $request)
14     {
15         $validated = $request->validate([
16             'number1' => 'required|numeric',
17             'number2' => 'required|numeric',
18             'operator' => 'required|in:add,sub,mul,div'
19         ]);
20
21         $result = match ($validated['operator']) {
22             'add' => $validated['number1'] + $validated['number2'],
23             'sub' => $validated['number1'] - $validated['number2'],
24             'mul' => $validated['number1'] * $validated['number2'],
25             'div' => $validated['number2'] != 0 ? ($validated['number1'] / $validated['number2']) : 'Error'
26         };
27
28         return view('calculator', [
29             'result' => $result,
30             'number1' => $validated['number1'],
31             'number2' => $validated['number2'],
32             'operator' => $validated['operator']
33         ]);
34     }
35 }
```

The status bar at the bottom indicates the current position is Line 9, Column 28, with 4 spaces, UTF-8 encoding, LF line endings, and PHP syntax highlighting.

- Tambahkan view calculator.blade.php.

```

<title>Laravel Calculator</title>
</head>
<body>
<h1>Kalkulator Sederhana</h1>

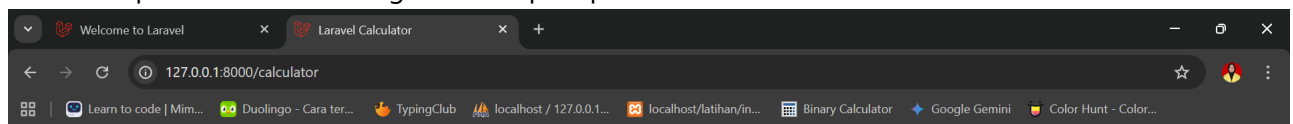
@if ($errors->any())
<div style="color: red">
<ul>
@foreach ($errors->all() as $error)
<li>{{ $error }}</li>
@endforeach
</ul>
</div>
@endif

<form method="POST" action="{{ route('calculator.calculate') }}">
@csrf
<input type="number" name="number1" value="{{ old('number1', $number1 ?? '') }}" placeholder="
<select name="operator" required>
<option value="add" {{ ($operator ?? '') == 'add' ? 'selected' : '' }}>+</option>
<option value="sub" {{ ($operator ?? '') == 'sub' ? 'selected' : '' }}>-</option>
<option value="mul" {{ ($operator ?? '') == 'mul' ? 'selected' : '' }}>*</option>
<option value="div" {{ ($operator ?? '') == 'div' ? 'selected' : '' }}>/</option>
</select>

<input type="number" name="number2" value="{{ old('number2', $number2 ?? '') }}" placeholder="
<button type="submit">Hitung</button>
</form>
@isset($result)
<h3>Hasil: {{ $result }}</h3>
@endisset
</body>
</html>

```

- Jalankan aplikasi dan coba dengan beberapa input berbeda. Screenshot Hasil:

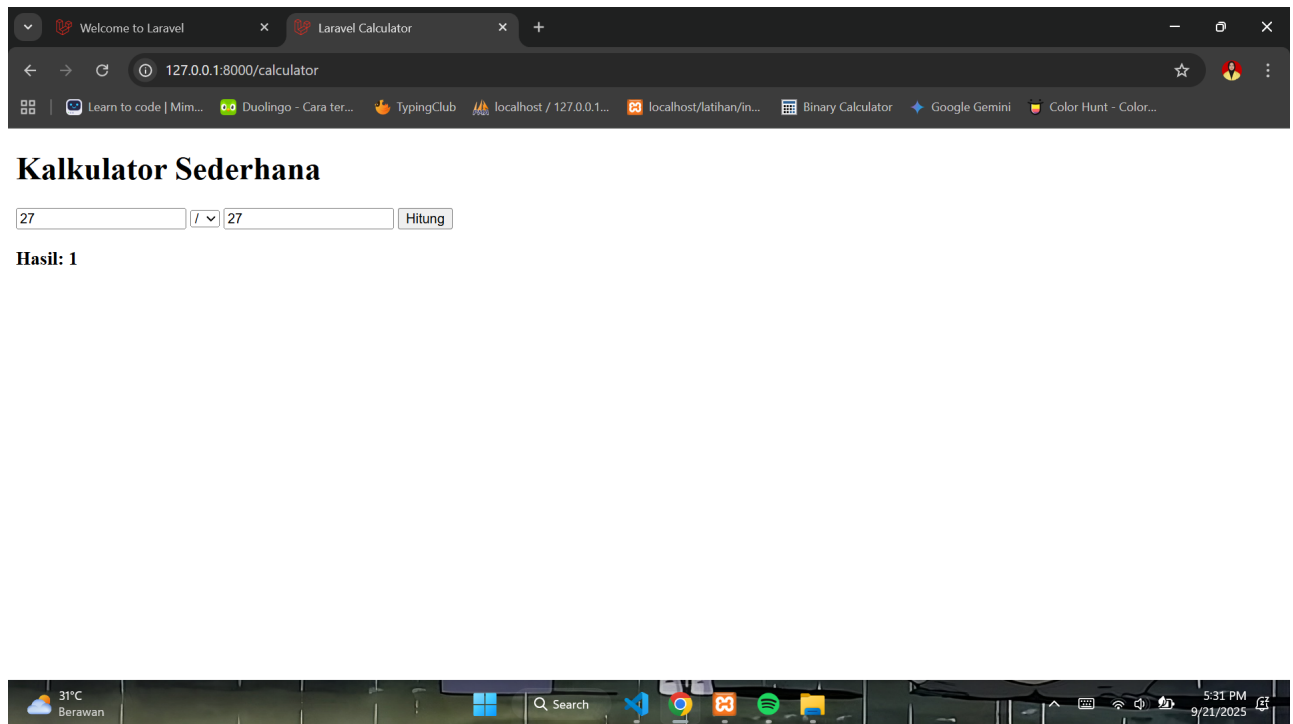


Kalkulator Sederhana

27 * 5 Hitung

Hasil: 135





3. Hasil dan Pembahasan

- Apakah aplikasi berjalan sesuai harapan? Aplikasi sudah berjalan sesuai yang diharapkan. Fitur-fiturnya juga sudah dapat digunakan, didalam aplikasi sudah terdapat operasi aritmatika seperti penjumlahan, pengurangan, perkalian dan pembagian.
- Apa yang terjadi jika ada input yang salah (misalnya pembagian dengan 0)? Jika pembagian dilakukan dengan input 0, maka hasilnya adalah: **Hasil: Error: Division by 0**
- Bagaimana validasi input bekerja di Laravel? Laravel Validasi Input dan penanganan kesalahan adalah dua aspek krusial dalam membangun aplikasi web yang aman, handal, dan user-friendly. Cara paling umum dan direkomendasikan untuk melakukan validasi di Laravel adalah dengan menggunakan metode `validate()` yang tersedia pada objek Request di dalam controller kamu. Metode ini secara otomatis akan mengalihkan pengguna kembali ke lokasi sebelumnya jika validasi gagal, dan akan menyimpan semua pesan kesalahan dalam session. Untuk validasi yang lebih kompleks atau ketika kamu ingin memisahkan logika validasi dari controller untuk menjaga kebersihan kode, Laravel Form Request adalah pilihan terbaik. Form Request adalah kelas khusus yang dapat kamu buat menggunakan Laravel Artisan.
- Apa peran masing-masing komponen (Route, Controller, View) dalam program yang dibuat? a. Route: Route yang menerima request dari user lewat URL, lalu menentukan request tersebut dibawa ke controller mana. b. Controller: Bagian yang mengatur logika. misalnya menghitung penjumlahan, perkalian atau mengambil data dari database. c. View: View adalah tampilan untuk user. Berisi file HTML + Blade syntax untuk menampilkan data yang dikirim controller.

4. Kesimpulan

Dari praktikum ini dapat diambil kesimpulan bahwa dengan adanya MVC sangat membantu dalam pengembangan aplikasi. MVC membantu memisahkan logika aplikasi tampilan dan pengaturan sehingga

lebih terstruktur. Pada praktikum kali ini dapat disimpulkan juga mengenai alur kerja laravel mulai dari menerima request hingga memberi response.

5. Referensi

Laravel Fundamentals — <https://hackmd.io/@mohdrzu/B1zwKEK5xe> Laravel Validasi Input dan Error Handling
— <https://neon.web.id/onphpid/laravel-validasi-input-dan-error-handling.html>
