

Laporan Modul 3: Laravel Controller

Mata Kuliah: Workshop Web Lanjut

Nama: Fathan Mubina

NIM: 2024573010058

Kelas: TI-2C

Abstrak

Dalam Praktikum Modul 3, mahasiswa mempelajari konsep dan aplikasi controller dalam framework Laravel 12 yang merupakan bagian dari arsitektur Model-View-Controller (MVC). Controller memiliki fungsi dalam mengatur dan mengelola alur logika dari suatu aplikasi, menghubungkan model dan view, serta menangani request dan response dari pengguna. Praktikum ini terdiri dari 3 bagian, yaitu: (1) menangani request dan response view menggunakan controller secara dasar, (2) mengelompokkan routes menggunakan route group, dan (3) menerapkan prefix dan namespace untuk pengorganisasian controller dalam struktur proyek Laravel. Dengan ini mahasiswa diharapkan paham cara membuat controller, mendefinisikan route dan menampilkan view yang diinginkan. Praktikum ini membuktikan bahwa controller memudahkan pengelolaan kode program agar lebih terstruktur, modular, dan lebih mudah untuk dikembangkan.

1. Dasar Teori

1. Apa Itu Controller

- Salah satu yang terpenting dari komponen arsitektur Model-View-Controller (MVC) dalam framework Laravel adalah controller. Controller berfungsi untuk merancang alur logika dari sebuah aplikasi dengan menjembatani antara model yang mengurus data dan database dengan view yang menyajikan antarmuka bagi pengguna. Menggunakan controller akan membuat penataan kode program menjadi lebih terorganisasi dan lebih mudah dibaca dengan pemisahan antara logika bisnis dan antarmuka.

2. Jelaskan fungsi utama Controller dalam arsitektur MVC pada Laravel.

Dalam arsitektur MVC, controller memiliki beberapafungsi utama, yaitu:

- Mengatur Alur Data – Controller menerima request dari pengguna, memprosesnya (jika perlu menggunakan model), dan mengembalikan response yang sesuai melalui view.
- Menangani Logika Aplikasi – Segala proses logika seperti validasi, pengambilan data, maupun pemrosesan hasil dilakukan dalam controller agar kode pada view dan model tetap bersih.
- Menghubungkan Komponen MVC – Controller menjadi penghubung antara view dan model sehingga komunikasi data berjalan dua arah secara efisien.
- Meningkatkan Pemeliharaan Kode – Dengan pembagian tanggung jawab yang jelas, perubahan pada logika aplikasi tidak mempengaruhi tampilan (view) secara langsung.

3. Sebutkan dan jelaskan tiga jenis controller yang ada di Laravel.

- Basic Controllers (Controller Dasar)

- “Controller” dasar difungsikan untuk menjalankan logika sederhana seperti mengembalikan view atau data tertentu. Biasanya untuk membuat controller ini dilakukan perintah: `php artisan make:controller NamaController`. Controller dasar juga dapat memiliki beberapa metode untuk menangani berbagai rute dalam satu file.
 - Resource Controllers (Controller Sumber Daya)
 - Resource controller digunakan untuk mengelola data yang memiliki operasi CRUD (Create, Read, Update, Delete). Laravel menyediakan cara otomatis untuk membuat metode bawaan seperti `index`, `create`, `store`, `show`, `edit`, `update`, dan `destroy`.
Perintahnya yaitu:
`php artisan make:controller NamaController --resource`.
Jenis controller ini membantu mempercepat pengembangan aplikasi berbasis sumber daya seperti manajemen pengguna, produk, atau artikel.
 - Invokable Controllers (Controller yang Dapat Dipanggil)
 - Controller ini hanya terdapat satu metode `__invoke()`, digunakan saat satu rute hanya membutuhkan satu aksi. Invokable controller ini cocok untuk fungsi-fungsi yang sederhana dan spesifik. Contoh membuatnya:
`php artisan make:controller NamaController --invokable`
-

2. Langkah-Langkah Praktikum

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

2.1 Praktikum 1 – Menangani Request dan Response View di laravel 12

1. Langkah 1 : Buat Dan Buka Proyek Laravel

- `composer create-project laravel/laravel:^12.0.3 lab-view`
- `cd lab-view`
- `code .`

2. Langkah 2: Buat sebuah Controller

`php artisan make:controller DemoController` Kemudian isi dengan kode berikut:

```
1  <?php
2
3  namespace App\Http\Controllers;
4  use Illuminate\Http\Request;
5
6  class DemonController extends Controller
7  {
8      // Simple data passing
9      public function hello()
10     {
11         $name = 'Laravel Learner';
12         return view('hello', ['name' => $name]);
13     }
14
15     // Parameterized route
16     public function greet($name)
17     {
18         return view('greet', ['name' => ucfirst($name)]);
19     }
20
21     // Query string
22     public function search(Request $request)
23     {
24         $keyword = $request -> query('q', 'none');
25         return view('search', ['keyword' => $keyword]);
26     }
27 }
28
```

3. Langkah 3: Definisikan Rute

Edit routes/web.php:

Kemudian Isi Dengan Code Berikut:

```
1  <?php
2
3  use Illuminate\Support\Facades\Route;
4  use App\Http\Controllers\DemonController;
5
6  Route::get('/hello', [DemonController::class, 'hello']);
7  Route::get('/greet/{naem}', [DemonController::class, 'greet']);
8  Route::get('/search', [DemonController::class, 'search']);
9  Route::get('/', function () {
10     return view('welcome');
11 });
12
```

4. Langkah 4: Buat View Sederhana

Buat file Blade berikut di resources/views/: Hello.blade.php

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title>Hello</title>
5  </head>
6  <body>
7      <h1>Hello, {{ $name }}!</h1>
8  </body>
9  </html>
```

Greet.blade.php



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <title>Greet</title>
5 </head>
6 <body>
7     <h1>Nice to meet you, {{ $name }}!</h1>
8 </body>
9 </html>
```

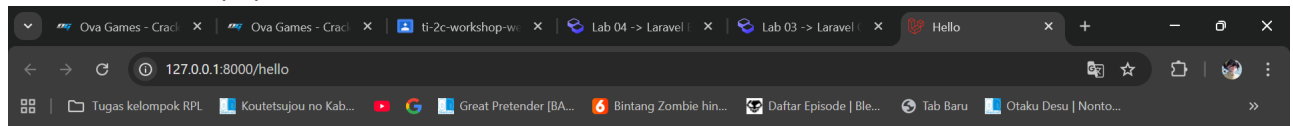
Search.blade.php



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <title>Search</title>
5 </head>
6 <body>
7     <h1>You searched for: <strong>{{ $keyword }}</strong></h1>
8 </body>
9 </html>
```

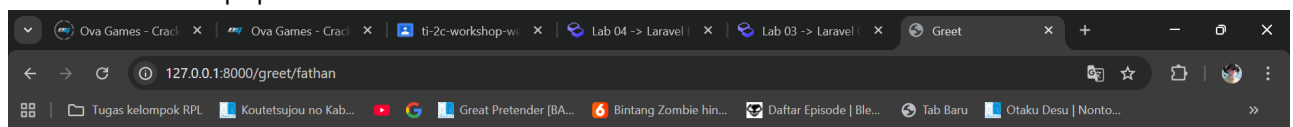
Screenshot Hasil:

- Hasil Hello.blade.php



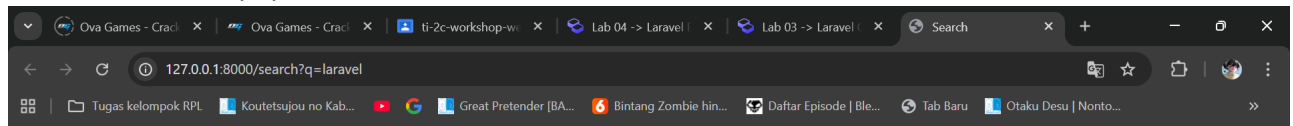
Hello, Laravel Learner!

-
- Hasil Greet.blade.php



Nice to meet you, Fathan!

- Hasil Search.blade.php



You searched for: laravel

2.2 Praktikum 2 – Menggunakan Group Route

1. Langkah 1 : Buat Dan Buka Proyek Laravel'

- `composer create-project laravel/laravel:^12.0.3 lab-group`
- `cd lab-group`
- `code .`

2. Langkah 2 : Buat Sebuah Controller php artisan make:controller PageController Kemudian Isi Dengan Code Berikut :



```
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6
7  class PageController extends Controller
8  {
9      public function home()
10     {
11         $message = "Welcome to the homepage.";
12         return view('pages.home', compact('message'));
13     }
14
15     public function about()
16     {
17         $message = "This is the about page.";
18         return view('pages.about', compact('message'));
19     }
20
21     public function contact()
22     {
23         $message = "Reach us through the contact page.";
24         return view('pages.contact', compact('message'));
25     }
26 }
27
```


3. Langkah 3 : Definisikan Rute Yang Dikelompokkan Kemudian Isi Dengan Code Berikut :


A screenshot of a code editor with a dark background and light-colored text. The code is PHP and defines routes for a web application. It includes the Laravel Route facade and a PageController. The routes are grouped under a single controller. The code is as follows:

```
1  <?php
2
3  use Illuminate\Support\Facades\Route;
4  use App\Http\Controllers\PageController;
5
6  Route::controller(PageController::class)->group(function () {
7      Route::get('/home', 'home')->name('home');
8      Route::get('/about', 'about')->name('about');
9      Route::get('/contact', 'contact')->name('contact');
10 });
11 Route::get('/', function () {
12     return view('welcome');
13 });
14
```

4. Langkah 4 : Buat View Sederhana


- Buat Folder : resources/views/pages/
- Selanjutnya Buat File - File Berikut Di dalam Pages/

■ Home.blade.php



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <title>Home</title>
5 </head>
6 <body>
7     <h1>{{ $message }}!</h1>
8 </body>
9 </html>
```

■ about.blade.php

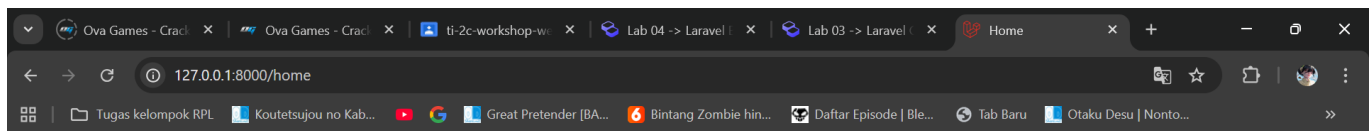


```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <title>About</title>
5 </head>
6 <body>
7     <h1>{{ $message }}!</h1>
8 </body>
9 </html>
```

■ contact.blade.php

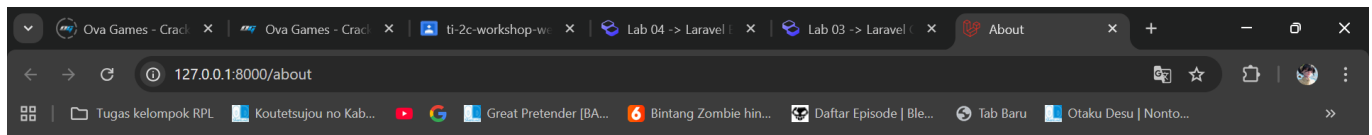
```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <title>Contact</title>
5 </head>
6 <body>
7     <h1>{{ $message }}!</h1>
8 </body>
9 </html>
```

Screenshot Hasil: Hasil Home:



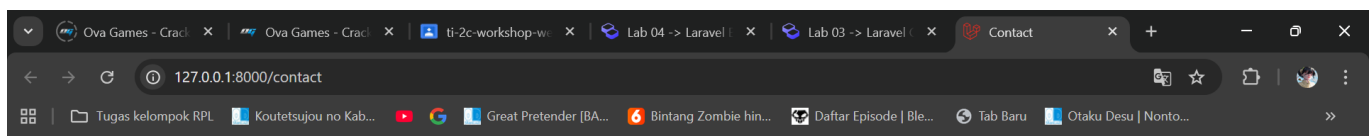
Welcome to the homepage.!

Hasil About:



This is the about page.!

Hasil Contact:



Reach us through the contact page.!

2.3 Praktikum 3 – Pengelompokan Prefix Dengan NameSpace Rute Di Laravel 12

1. Langkah 1: Buat Dan Buka Proyek Laravel

buka bash dan ketik teks berikut ini di dalam bash

- `composer create-project laravel/laravel:^12.0.3 lab-prefix`
- `cd lab-prefix`
- `code .`

2. Langkah 2: Buat Controller Dengan Namespace

buat controller berikut ini menggunakan bash

- php artisan make:controller Admin/DashboardController
- php artisan make:controller Admin/UserController

3. Langkah 3: Definisikan Kelompok Rute Dengan Prefix Dan Namespace Controller Isi file route/web.php Dengan Code Berikut ini

```
1  <?php
2
3  use Illuminate\Support\Facades\Route;
4  use App\Http\Controllers\Admin\DashboardController;
5  use App\Http\Controllers\Admin\UserController;
6
7  Route::prefix('admin')->group(function() {
8      Route::get('/dashboard', [DashboardController::class, 'index'])->name('admin.dashboard');
9      Route::get('/users', [UserController::class, 'index'])->name('admin.users');
10     Route::get('/users/{id}', [UserController::class, 'show'])->name('admin.users.show');
11 });
12 Route::get('/', function () {
13     return view('welcome');
14 });
15
```

4. Langkah 4: Tambahkan Aksi Ke Controller masukkan Code Berikut ini di dalam file DashboardController.php

```
1  <?php
2
3  namespace App\Http\Controllers\Admin;
4
5  use App\Http\Controllers\Controller;
6  use Illuminate\Http\Request;
7
8  class DashboardController extends Controller
9  {
10     public function index()
11     {
12         return view('admin.dashboard', ['message' => 'Welcome to Admin Dashboard']);
13     }
14 }
15
```

Selanjutnya, masukkan Code Berikut ini di dalam file UsersController.php

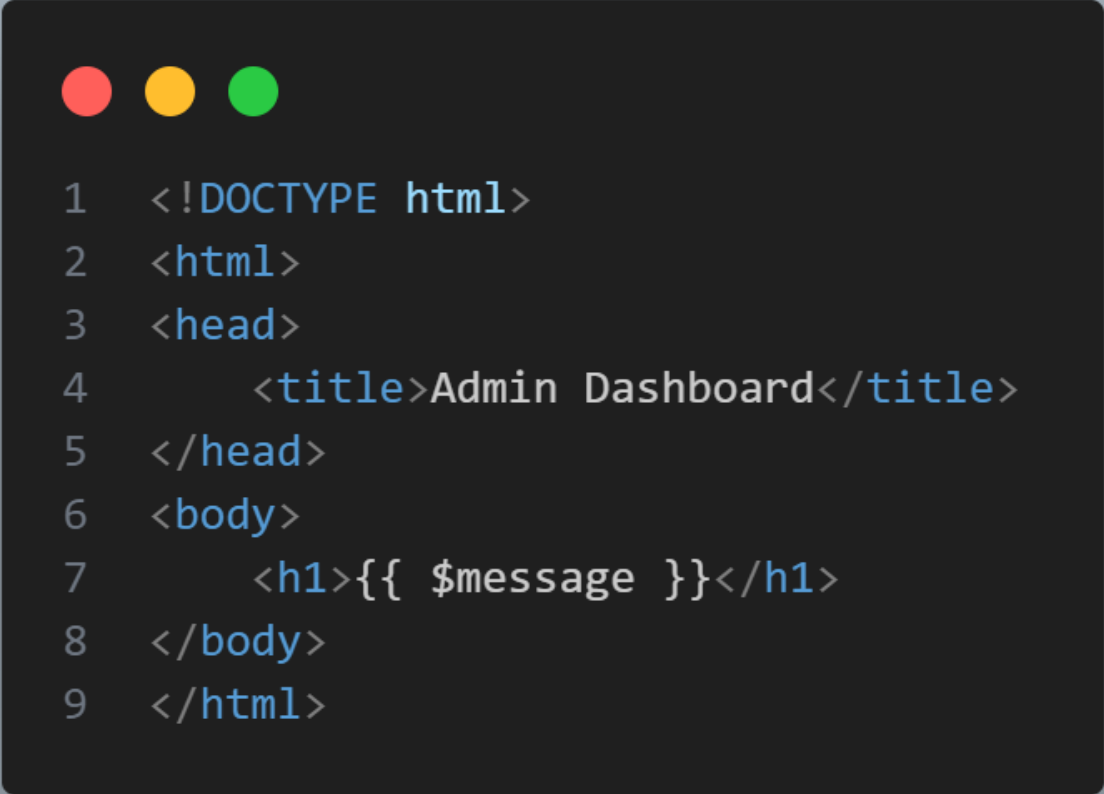
```
1  <?php
2
3  namespace App\Http\Controllers\Admin;
4
5  use App\Http\Controllers\Controller;
6  use Illuminate\Http\Request;
7
8  class UserController extends Controller
9  {
10     public function index()
11     {
12         $users = ['Ria', 'Lie', 'jon'];
13         return view('admin.users.index', compact('users'));
14     }
15
16     public function show($id)
17     {
18         $user = "User #". $id;
19         return view('admin.users.show', compact('user'));
20     }
21 }
22
```

5. Langkah 5: Buat View Sederhana

Buat file berikut ini di dalam resources/views/admin/:

- dashboard.blade.php
Kemudian buat folder users di dalam admin serta file berikut ini:
- users/index.blade.php

Masukkan Code Berikut pada file Dashboard.blade.php



```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title>Admin Dashboard</title>
5  </head>
6  <body>
7      <h1>{{ $message }}</h1>
8  </body>
9  </html>
```

Selanjutnya, membuat folder users di dalam resources/views/admin/.

buat file di resources/Views/admin/users/index.blade.php, ini adalah bahasa HTML yang sederhana untuk menampilkan daftar pengguna: Kemudian masukkan index.blade.php Dengan code berikut ini:



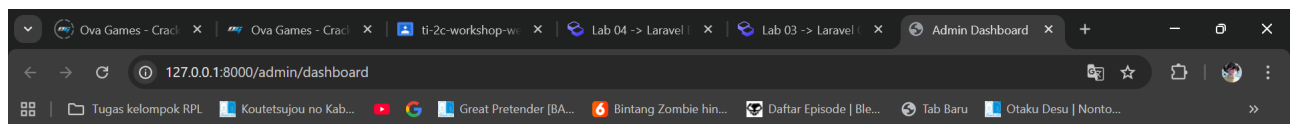
```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title>User</title>
5  </head>
6  <body>
7      <h1>User List</h1>
8      <ul>
9          @foreach ($users as $user)
10             <li>{{ $user }}</li>
11          @endforeach
12      </ul>
13  </body>
14  </html>
```

Setelah itu Bikin file Show.blade.php isi dengan code berikut

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title>User Detail</title>
5  </head>
6  <body>
7      <h1>Details for: {{ $user }}</h1>
8  </body>
9  </html>
```

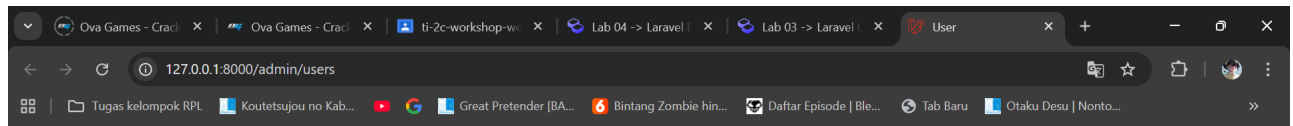
Screenshot Hasil:

- Hasil dari Admin Dashboard



Welcome to Admin Dashboard

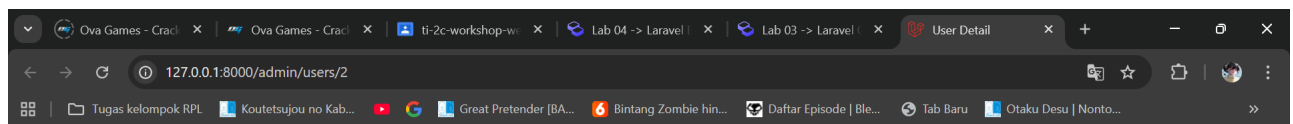
- Hasil dari Admin Users



User List

- Ria
- Lie
- jon

- Hasil dari Users Detail



Details for: User #2

3. Hasil dan Pembahasan

Praktikum 1 – Menangani Request dan Response View di Laravel 12

Pada praktikum pertama, dilakukan pembuatan proyek baru bernama lab-view untuk memahami cara controller menangani request dan menampilkan response view. Controller yang dibuat adalah DemoController, yang berfungsi untuk mengarahkan request ke beberapa tampilan seperti Hello.blade.php, Greet.blade.php, dan Search.blade.php. Setiap metode dalam controller mengembalikan tampilan yang

berbeda sesuai dengan rute yang telah ditentukan di `routes/web.php`.

Hasil pengujian menunjukkan bahwa setiap rute menampilkan halaman yang sesuai, yaitu:

- `/hello` menampilkan teks sapaan dari `Hello.blade.php`.
- `/greet` menampilkan halaman salam menggunakan variabel dari controller.
- `/search` menampilkan halaman pencarian sederhana. Hal ini menunjukkan bahwa controller berperan penting dalam mengatur data dan tampilan secara dinamis sesuai permintaan pengguna.

Praktikum 2 – Menggunakan Group Route

Pada percobaan kedua, dibuat proyek baru bernama `lab-group` untuk mempelajari pengelompokan rute menggunakan Route Group. Controller yang digunakan adalah `PageController`, yang berisi metode `home()`, `about()`, dan `contact()`. Ketiga metode tersebut mengarah ke tampilan view yang berbeda di folder `resources/views/pages/`.

Penggunaan route group memungkinkan pengelompokan beberapa rute dengan kesamaan tertentu, seperti prefix atau middleware. Dalam praktikum ini, semua rute dikelompokkan untuk mempermudah pengaturan dan menghindari penulisan rute berulang.

Hasil pengujian:

- `/pages/home` menampilkan halaman utama.
- `/pages/about` menampilkan informasi tentang halaman.
- `/pages/contact` menampilkan halaman kontak.

Dari hasil ini dapat disimpulkan bahwa route group sangat membantu dalam manajemen rute yang memiliki struktur serupa dan meningkatkan keteraturan kode.

Praktikum 3 – Pengelompokan Prefix dengan Namespace Rute di Laravel 12

\Praktikum ketiga berfokus pada penerapan prefix dan namespace untuk mengelompokkan rute serta memisahkan controller berdasarkan kategori. Proyek baru bernama `lab-prefix` dibuat dengan dua controller di dalam folder `Admin`, yaitu `DashboardController` dan `UserController`.

Melalui pengaturan prefix admin pada `routes/web.php`, semua rute yang berkaitan dengan admin dapat diakses melalui URL yang dimulai dengan `/admin`.

Selain itu, penggunaan namespace mempermudah pemanggilan controller tanpa perlu menuliskan path lengkap pada setiap rute.

Hasil pengujian menunjukkan:

- `/admin/dashboard` menampilkan tampilan dari `DashboardController`.
- `/admin/users` menampilkan daftar pengguna dari `UserController`.
- `/admin/users/{id}` menampilkan detail pengguna tertentu.

Dari hasil ini, dapat disimpulkan bahwa penggunaan prefix dan namespace menjadikan struktur proyek lebih terorganisir, terutama untuk aplikasi berskala besar yang memiliki banyak controller.

4. Kesimpulan

Dari hasil praktikum Modul 3 tentang Laravel Controller dapat disimpulkan bahwa controller memiliki peranan penting dalam arsitektur MVC pada framework Laravel. Controller berfungsi sebagai penghubung antara model dan view, serta bertugas menangani logika aplikasi, request, dan response dari pengguna. Melalui praktikum pertama, mahasiswa dapat memahami bagaimana controller bekerja dalam menampilkan tampilan (view) secara dinamis. Pada praktikum kedua, penerapan route group mempermudah pengelompokan beberapa rute yang memiliki kesamaan sehingga struktur kode menjadi lebih rapi dan efisien. Selanjutnya, pada praktikum ketiga, penggunaan prefix dan namespace membantu dalam pengorganisasian controller sesuai kategori, terutama untuk proyek berskala besar. Secara keseluruhan, penggunaan controller di Laravel memberikan kemudahan dalam pengembangan aplikasi web yang terstruktur, mudah dipelihara, serta sesuai dengan prinsip pemrograman berbasis MVC.

5. Referensi

- Modul 3 - Laravel Controller - <https://hackmd.io/@mohdrzu/H1sB73dnxg>
 - Laravel - <https://laravel.com/docs/12.x/controllers>
-