

Laporan Modul 3: Laravel Controller

Mata Kuliah: Workshop Web Lanjut

Nama: Hamizan Putra Zulia **NIM:** 2024573010013 **Kelas:** TI-2C

Abstrak

Laporan praktikum ini membahas tentang bagaimana cara kerja dan penerapan Controller dalam framework Laravel 12. Controller berperan penting dalam konsep arsitektur MVC (Model-View-Controller) karena menjadi penghubung antara view (tampilan) dan model (data). Pada praktikum ini dilakukan tiga percobaan utama yaitu: membuat controller dasar untuk menangani request dan response, membuat group route dengan controller, serta mengelompokkan prefix dan namespace pada route. Tujuan dari praktikum ini adalah agar saya dapat memahami bagaimana controller mengatur alur logika, memanggil view, dan berinteraksi dengan route di Laravel. Dengan memahami controller, diharapkan proses pengembangan aplikasi menjadi lebih terstruktur, rapi, dan mudah di maintain.

1. Dasar Teori

- Apa itu Controller?

Controller adalah salah satu komponen inti dari MVC yang berfungsi sebagai penghubung antara request user (View) ke model yang nantinya akan di kembalikan lagi ke View dalam bentuk response. Controller ini akan banyak berisi logika – logika dalam menyusun suatu fungsi tertentu. Contohnya adalah aktivitas CRUD (Create, Read, Update, Delete) yang prosesnya berjalan di dalam Controller.

- Jenis-jenis Controller

1. Basic Controllers
2. Resource Controllers
3. Invokable Controllers

- Routing dan Controller pada Laravel

Routing adalah proses transmisi atau pengiriman data dari satu node ke node lainnya dalam suatu jaringan komputer atau sistem komunikasi. Ini adalah bagian penting dari bagaimana informasi ditransmisikan melalui jaringan yang terdiri dari berbagai perangkat seperti router, switch, dan gateway. Tujuan utama dari routing adalah untuk menentukan rute atau rute terbaik agar paket data mencapai tujuan akhirnya secara efisien dan andal.

Dalam routing, setiap node dalam jaringan memiliki tabel routing yang berisi informasi tentang cara mencapai tujuan tertentu. Tabel ini diperbarui secara dinamis oleh protokol perutean, yang dapat berupa protokol statis atau dinamis, seperti RIP (Routing Information Protocol), OSPF (Open Shortest Path First) atau BGP (Border Gateway Protocol). Proses perutean melibatkan keputusan tentang rute mana yang akan diambil paket data berdasarkan informasi dalam tabel perutean.

Keputusan ini dapat dipengaruhi oleh beberapa faktor, antara lain jarak, penggunaan jaringan, dan kondisi terkini dari setiap rute yang tersedia. Perutean memainkan peran penting dalam menjaga koneksi yang andal dan efisien dalam jaringan kompleks dengan banyak perangkat yang saling

terhubung. Ini juga merupakan bagian penting dari infrastruktur Internet global, yang memungkinkan komunikasi dan pertukaran data antara perangkat dan pengguna di seluruh dunia.

Controller adalah “Pengendali” yaitu konsep yang muncul dalam berbagai konteks, termasuk teknologi, sistem, dan manajemen. Secara umum, “pengendali” mengacu pada entitas yang memiliki kekuasaan atau kendali atas suatu proses atau sistem. Dalam konteks teknologi dan sistem komputer, “pengendali” dapat merujuk pada perangkat keras atau perangkat lunak yang bertanggung jawab untuk mengatur atau mengelola fungsi-fungsi tertentu dari suatu sistem. Contohnya termasuk pengontrol hard drive, yang melakukan operasi baca/tulis untuk data pada media penyimpanan, atau pengontrol jaringan, yang menangani komunikasi data melalui jaringan.

2. Langkah-Langkah Praktikum

2.1 Praktikum 1 - Menangani Request dan Response View di Laravel 12

- Langkah-langkah:

1. Buat dan Buka Proyek Laravel, dengan mengetik:

```
web-lanjut/projects main • ? > laravel new lab-view
```

Lalu masuk ke dalam folder projek nya

2. Buat sebuah Controller, dengan mengetik:

```
web-lanjut/projects main • ? X php artisan make:controller DemoController
```

Dan tambahkan kode-kode ini di DemoController:

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

3. Definiskan Route di routes/web.php:

```
1  <?php
2
3  use App\Http\Controllers\DemoController;
4  use Illuminate\Support\Facades\Route;
5
6  Route::get('/hello', [DemoController::class, 'hello']);
7  Route::get('/greet/{name}', [DemoController::class, 'greet']);
8  Route::get('/search', [DemoController::class, 'search']);
```

4. Buat tiga file view web sederhana, contoh:

- Di hello.blade.php

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

- Di greet.blade.php

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

- Di search.blade.php

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

5. Sudah selesai dan sudah dapat dijalankan menggunakan "php artisan serve"

2.2 Praktikum 2 - Menggunakan Group Route

- Langkah-langkah:

1. Buat dan buka proyek Laravel, dengan mengetik:

```
web-lanjut/projects main • ? > laravel new lab-group
```

Lalu masuk ke dalam folder proyek nya

2. Buat controller nya, dengan mengetik:

```
web-lanjut/projects main • ? X php artisan make:controller PageController
```

3. Masuk ke PageController lalu ubah seperti ini:

```
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
```

4. Masuk ke file routes/web.php untuk menambahkan route

```
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')->name('home');
8      Route::get('/about', 'about')->name('about');
9      Route::get('/contact', 'contact')->name('contact');
10 });
```

5. Di resource/views/ tambahkan tiga file dan isi seperti dibawah ini:

- home.blade.php

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

- about.blade.php

```

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

```

- contact.blade.php

```

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

```

6. Sudah selesai dan sudah dapat dijalankan menggunakan "php artisan serve"

2.3 Praktikum 3 - Pengelompokan Prefix dengan Namespace Rute di Laravel 12

- Langkah-langkah

1. Buat dan buka proyek Laravel, dengan mengetik:

```
web-lanjut/projects main • ? X laravel new lab-prefix
```

2. Buat controller dengan namespace menggunakan ketikan ini di git bash/terminal:

```
web-lanjut/projects main • ? X php artisan make:controller Admin/DashboardController
web-lanjut/projects main • ? X php artisan make:controller Admin/UserController
```

3. Mendefinisikan kelompok route dengan prefix dan namespace controller di routes/web.php

```

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

```

4. Menambahkan aksi ke Controller

- ke Admin/DashboardController

```
<?php

namespace App\Http\Controllers\Admin;

use App\Http\Controllers\Controller;
use Illuminate\Http\Request;

class DashboardController extends Controller
{
    public function index()
    {
        return view('admin.dashboard', ['message' => 'Welcome to Admin Dashboard']);
    }
}
```

- ke Admin/UserController

```
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. Buat View sederhana

Tambahkan file-file dan folder yg dibutuhkan, seperti:

1. dashboard.blade.php
2. users/index.blade.php
3. users/show.blade.php

- Tambahkan kode berikut di dashboard.blade.php

```
projects > lab-prefix > resources > views > dashboard.blade
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <title>Admin Dashboard</title>
5 </head>
6 <body>
7     <h1> {{ $message }} </h1>
8 </body>
9 </html>
```

- Tambahkan kode berikut di users/index.blade.php

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

- Tambahkan kode berikut di users/show.blade.php

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

3. Hasil dan Pembahasan

Dari ketiga praktikum yang dilakukan, saya jadi lebih memahami bagaimana Laravel mengatur logika aplikasi menggunakan controller.

Pada Praktikum 1, saya membuat DemoController untuk menangani request sederhana dan menampilkan view seperti hello.blade.php, greet.blade.php, dan search.blade.php. Dari sini saya belajar bahwa controller bertugas menerima request dari route dan mengarahkan hasilnya ke view yang sesuai. Intinya, controller adalah jembatan antara user dan tampilan web.

Di Praktikum 2, saya mempelajari cara membuat group route. Fitur ini berguna banget buat ngatur route yang banyak biar lebih rapi dan mudah dibaca. Dengan menggunakan Route::controller() dan Route::group(), saya bisa mengelompokkan beberapa route yang menggunakan controller yang sama, seperti PageController untuk halaman home, about, dan contact. Jadi kodenya lebih efisien dan tidak berulang-ulang.

Lalu di Praktikum 3, saya belajar menggunakan prefix dan namespace pada route. Di bagian ini, saya membuat controller yang dipisahkan berdasarkan folder, misalnya di App\Http\Controllers\Admin. Penggunaan prefix dan namespace ini membuat struktur project lebih terorganisir, apalagi kalau nanti aplikasi sudah besar dan punya banyak bagian seperti admin, user, dan lainnya. Misalnya, route admin/dashboard dan admin/users bisa langsung diarahkan ke controller berbeda dengan struktur folder yang teratur.

Secara keseluruhan, saya jadi lebih paham peran besar controller di Laravel. Semua logika aplikasi bisa ditaruh di controller supaya kode lebih bersih, gak berantakan di file route, dan lebih mudah dikelola kalau aplikasi terus berkembang.

4. Kesimpulan

Dari praktikum ini, saya bisa menyimpulkan bahwa controller adalah bagian penting dalam Laravel karena menjadi pengatur alur data dari route menuju view. Dengan controller, kode jadi lebih terstruktur dan sesuai dengan konsep MVC. Saya juga belajar cara menggunakan group route untuk merapikan banyak rute, serta bagaimana mengelompokkan prefix dan namespace agar controller bisa diatur sesuai fungsinya masing-masing. Secara pribadi, saya merasa konsep controller ini bikin pengembangan web lebih mudah dan fleksibel, terutama saat aplikasi mulai besar. Praktikum ini bikin saya makin paham gimana cara Laravel ngatur arsitektur aplikasinya dengan rapi dan efisien.

5. Referensi

- Apa itu Controller? - <https://informatika.ciputra.ac.id/2019/10/laravel-controller/>
 - Routing Dan Controller Pada Laravel - <https://course-net.com/blog/routing-dan-controller-pada-laravel/>
-