

# Laporan Modul 8: Authentication & Authorization

---

**Mata Kuliah:** Workshop Web Lanjut

**Nama:** Ahmad Aulia Fahlevi

**NIM:** 2024573010077 **Kelas:** TI-2C

---

## Abstrak

Praktikum ini mengeksplorasi dua pilar keamanan aplikasi web modern di Laravel: Autentikasi (verifikasi identitas pengguna) dan Otorisasi (menentukan izin akses pengguna). Tahap pertama berhasil mengimplementasikan sistem autentikasi dasar menggunakan Laravel Breeze untuk fitur registrasi, login, dan pengamanan rute dengan middleware auth. Tahap kedua berfokus pada Otorisasi Berbasis Peran (RBAC) dengan mendefinisikan peran (admin, manager, user). Hal ini dicapai dengan menambahkan kolom role pada tabel pengguna dan membuat Middleware Kustom (RoleMiddleware) untuk membatasi akses ke route spesifik. Hasil praktikum menunjukkan bahwa Laravel menyediakan tool yang powerful dan terstruktur untuk membangun sistem keamanan yang berlapis, dari validasi identitas hingga kontrol akses yang granular.

## 1. Dasar Teori

Dasar teori praktikum ini mencakup konsep inti keamanan web dan implementasinya dalam framework Laravel.

- **Autentikasi (Authentication)**
  - **Definisi:** Proses memverifikasi identitas pengguna, memastikan mereka adalah siapa yang mereka klaim.
    - **Implementasi Laravel:** Menggunakan starter kit Laravel Breeze untuk menyediakan scaffolding rute, controller, dan view siap pakai untuk login, register, dan password reset. Laravel secara otomatis mengamankan kata sandi menggunakan bcrypt dan mengelola sesi melalui cookies aman.
    - **Middleware auth:** Digunakan untuk melindungi rute, memastikan hanya pengguna yang telah terautentikasi (sudah login) yang dapat mengakses rute atau grup rute tertentu.
- **Otorisasi (Authorization)**
  - **Definisi:** Digunakan untuk melindungi rute, memastikan hanya pengguna yang telah terautentikasi (sudah login) yang dapat mengakses rute atau grup rute tertentu.
  - **Sistem Laravel:** Otorisasi dapat diimplementasikan menggunakan Gates (closure sederhana) dan Policies (kelas untuk logika otorisasi berbasis Model).
  - **Role-Based Access Control (RBAC):** Merupakan pendekatan yang lebih maju, yang dilakukan dalam praktikum ini dengan:
    - Menambahkan field role ke tabel users.

- Membuat Middleware Kustom untuk memeriksa nilai field role pengguna sebelum mengizinkan akses ke rute.
  - **Komponen Pendukung**
    - **Migrations dan Seeder:** Digunakan untuk memodifikasi struktur database (menambahkan kolom role) dan mengisi data pengguna awal dengan peran yang sudah ditentukan.
    - **Middleware Kustom:** Merupakan mekanisme yang mencegah permintaan HTTP. Middleware khusus ini digunakan untuk memverifikasi apakah `$request->user()->role` cocok dengan peran yang diizinkan pada route.
- 

## 2. Langkah-Langkah Praktikum

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

### 2.1 Praktikum 1 – Autentikasi dan Otorisasi dengan Laravel 12 Breeze

- Menginstall Laravel Breeze

```
Administrator@WIN-C32CTMDNOHK MINGW64 /e/Kampus/Semester 3/Workshop Web Lanjut/web-lanjut-2024573010077/project/auth-lab (main)
$ composer require laravel/breeze --dev
```

- Kemudian jalankan perintah "php artisan breeze:install" untuk menginstall Breeze, selama proses instalasi, kita akan diminta beberapa opsi:
  - Pilih frontend framework: Pilih blade
  - Dark mode: Pilih yes (opsional)

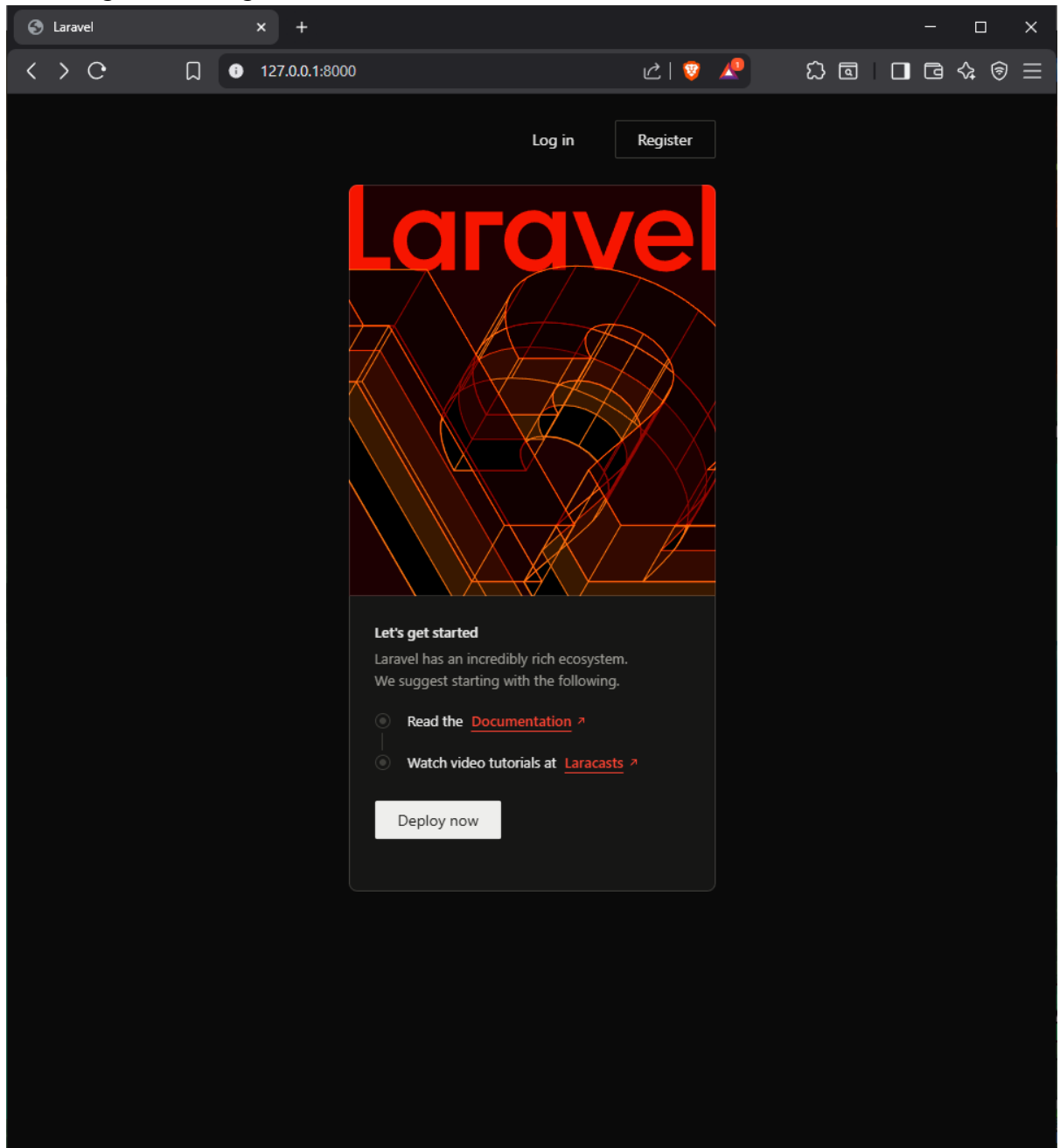
- Lanjutkan dengan perintah

```
Administrator@WIN-C32CTMDNOHK MINGW64 /e/Kampus/Semester 3/Workshop Web Lanjut/web-lanjut-2024573010077/project/auth-lab (main)
$ npm install
```

- Lalu lanjutkan dengan perintah

```
Administrator@WIN-C32CTMDNOHK MINGW64 /e/Kampus/Semester 3/Workshop Web Lanjut/web-lanjut-2024573010077/project/auth-lab (main)
$ php artisan migrate
```

- Akses Register dan Login via Web Interface



- Membuat Rute Profile yang Dilindungi di routes/web.php

```

1  <?php
2
3  use App\Http\Controllers\ProfileController;
4  use Illuminate\Support\Facades\Route;
5  use Illuminate\Support\Facades\Auth;
6
7  Route::get(uri: '/', function () {
8      return view( view: 'welcome');
9  });
10
11 Route::get(uri: '/dashboard', function () {
12     return view( view: 'dashboard');
13 })->middleware(['auth', 'verified'])->name( name: 'dashboard');
14
15 Route::middleware( middleware: 'auth')->group(function () {
16     Route::get(uri: '/profile', [ProfileController::class, 'edit'])->name( name: 'profile.edit');
17     Route::patch(uri: '/profile', [ProfileController::class, 'update'])->name( name: 'profile.update');
18     Route::delete(uri: '/profile', [ProfileController::class, 'destroy'])->name( name: 'profile.destroy');
19
20     Route::get(uri: '/myprofile', function () {
21         return Auth::user();
22     })->name( name: 'myprofile');
23 });
24
25 require __DIR__.'/auth.php';

```

- Menjalankan aplikasi dan Menunjukkan hasil dibrowser.

```

{
  "id": 1,
  "name": "lepi",
  "email": "lepi@gmail.com",
  "email_verified_at": null,
  "created_at": "2025-11-11T07:09:00.000000Z",
  "updated_at": "2025-11-11T07:09:00.000000Z"
}

```

## 2.2 Praktikum 2 – Membatasi Akses Berdasarkan Peran di Laravel 12

- Menginstall Laravel Breeze

```

Administrator@WIN-C32CTMDNOHK MINGW64 /e/Kampus/Semester 3/Workshop Web Lanjut/web-lanjut-2024573010077/project/role-lab (main)
$ composer require laravel/breeze --dev

```

- Kemudian jalankan perintah "php artisan breeze:install" untuk menginstall Breeze, selama proses instalasi, kita akan diminta beberapa opsi:
  - Pilih frontend framework: Pilih blade
  - Dark mode: Pilih yes (opsional)

- Lanjutkan dengan perintah

```

Administrator@WIN-C32CTMDNOHK MINGW64 /e/Kampus/Semester 3/Workshop Web Lanjut/web-lanjut-2024573010077/project/role-lab (main)
$ npm install

```

- Lalu lanjutkan dengan perintah

```

Administrator@WIN-C32CTMDNOHK MINGW64 /e/Kampus/Semester 3/Workshop Web Lanjut/web-lanjut-2024573010077/project/role-lab (main)
$ php artisan migrate

```

- Menambahkan Field Role ke Tabel Users.

- Membuat Migration setelah di edit kemudian kita jalankan dengan perintah "php artisan migrate"

```
Administrator@WIN-C32CTMDNOHK MINGW64 /e/Kampus/Semester 3/Workshop Web Lanjut/web-lanjut-2024573010077/project/role-lab (main)
$ php artisan make:migration add_role_to_users_table --table=users
```

```
1  <?php
2
3  use Illuminate\Database\Migrations\Migration;
4  use Illuminate\Database\Schema\Blueprint;
5  use Illuminate\Support\Facades\Schema;
6
7  return new class extends Migration
8  {
9      public function up(): void
10     {
11         Schema::table('users', function (Blueprint $table) {
12             $table->string('role')->default('user');
13         });
14     }
15
16     public function down(): void
17     {
18         Schema::table('users', function (Blueprint $table) {
19             $table->dropColumn('role');
20         });
21     }
22 };
```

- Membuat Seeding Pengguna dengan Peran Berbeda, setelah selesai kita jalankan "php artisan db:seed"

```

1      <?php
2
3      namespace Database\Seeders;
4
5      use App\Models\User;
6      use Illuminate\Database\Console\Seeds\WithoutModelEvents;
7      use Illuminate\Database\Seeder;
8      use Illuminate\Support\Facades\Hash;
9
10     class DatabaseSeeder extends Seeder
11     {
12         use WithoutModelEvents;
13
14         public function run(): void
15         {
16             // User::factory(10)->create();
17
18             // User::factory()->create([
19             //     'name' => 'Test User',
20             //     'email' => 'test@example.com',
21             // ]);
22
23             User::create([
24                 'name' => 'Lepi',
25                 'email' => 'lepi@gmail.com',
26                 'password' => Hash::make('majumapan'),
27                 'role' => 'admin'
28             ]);
29
30             User::create([
31                 'name' => 'Imam',
32                 'email' => 'imam@gmail.com',
33                 'password' => Hash::make('imam123'),
34                 'role' => 'manager'
35             ]);
36
37             User::create([
38                 'name' => 'Ilham',
39                 'email' => 'ilham@gmail.com',
40                 'password' => Hash::make('ilham123'),
41                 'role' => 'user'
42             ]);
43         }
44     }

```

- Membuat Role Middleware kemudian kita daftarkan di bootstrap/app.php.

```

Administrator@WIN-C32CTMDNOHK MINGW64 /e/Kampus/Semester 3/Workshop Web Lanjut/web-lanjut-2024573010077/project/role-lab (main)
$ php artisan make:middleware RoleMiddleware

```

```
1 <?php
2
3 namespace App\Http\Middleware;
4
5 use Closure;
6 use Illuminate\Http\Request;
7 use Symfony\Component\HttpFoundation\Response;
8
9 2 usages
10 class RoleMiddleware
11 {
12     public function handle(Request $request, Closure $next, string $role): Response
13     {
14         if ($request->user() && $request->user()->role === $role) {
15             return $next($request);
16         }
17
18         abort('code: 403, message: 'Unauthorized');
19     }
20 }
```

```
1 <?php
2
3 use App\Http\Middleware\RoleMiddleware;
4 use Illuminate\Foundation\Application;
5 use Illuminate\Foundation\Configuration\Exceptions;
6 use Illuminate\Foundation\Configuration\Middleware;
7
8 return Application::configure(basePath: dirname(__DIR__))
9     ->withRouting(
10         web: __DIR__.'/../routes/web.php',
11         commands: __DIR__.'/../routes/console.php',
12         health: '/up',
13     )
14     ->withMiddleware(function (Middleware $middleware) {
15         //
16         $middleware->alias([
17             'role' => RoleMiddleware::class,
18         ]);
19     })
20     ->withExceptions(function (Exceptions $exceptions) {
21         //
22     })->create();
```

- Membuat View untuk Setiap Role.

- Membuat Views admin.blade.php.

```

1  <x-app-layout>
2      <x-slot name="header">
3          <h2 class="font-semibold text-xl text-gray-800 dark:text-gray-200 leading-tight">
4              {{ __('Admin Dashboard') }}
5          </h2>
6      </x-slot>
7
8      <div class="py-12">
9          <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
10             <div class="bg-white dark:bg-gray-800 overflow-hidden shadow-sm sm:rounded-lg">
11                 <div class="p-6 text-gray-900 dark:text-gray-100">
12                     {{ __('Welcome, Admin! You have full access.') }}
13                 </div>
14             </div>
15         </div>
16     </div>
17 </x-app-layout>

```

- Membuat Views manager.blade.php.

```

1  <x-app-layout>
2      <x-slot name="header">
3          <h2 class="font-semibold text-xl text-gray-800 dark:text-gray-200 leading-tight">
4              {{ __('Manager Dashboard') }}
5          </h2>
6      </x-slot>
7
8      <div class="py-12">
9          <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
10             <div class="bg-white dark:bg-gray-800 overflow-hidden shadow-sm sm:rounded-lg">
11                 <div class="p-6 text-gray-900 dark:text-gray-100">
12                     {{ __('Welcome, Manager! You can manager and monitor resources.') }}
13                 </div>
14             </div>
15         </div>
16     </div>
17 </x-app-layout>

```

- Membuat Views user.blade.php.

```

1  <x-app-layout>
2      <x-slot name="header">
3          <h2 class="font-semibold text-xl text-gray-800 dark:text-gray-200 leading-tight">
4              {{ __('User Dashboard') }}
5          </h2>
6      </x-slot>
7
8      <div class="py-12">
9          <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
10             <div class="bg-white dark:bg-gray-800 overflow-hidden shadow-sm sm:rounded-lg">
11                 <div class="p-6 text-gray-900 dark:text-gray-100">
12                     {{ __('Welcome, User! You have limited access.') }}
13                 </div>
14             </div>
15         </div>
16     </div>
17 </x-app-layout>

```



- Membuat Views all.blade.php.

```

1      <x-app-layout>
2          <x-slot name="header">
3              <h2 class="font-semibold text-xl text-gray-800 dark:text-gray-200 leading-tight">
4                  {{ __('General Dashboard') }}
5              </h2>
6          </x-slot>
7
8          <div class="py-12">
9              <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
10                 <div class="bg-white dark:bg-gray-800 overflow-hidden shadow-sm sm:rounded-lg">
11                     <div class="p-6 text-gray-900 dark:text-gray-100">
12                         {{ __("Welcome! This view is accessible by all authenticated roles.") }}
13                     </div>
14                 </div>
15             </div>
16         </div>
17     </x-app-layout>

```

- Mendefinisikan Route untuk View Berbasis Role.

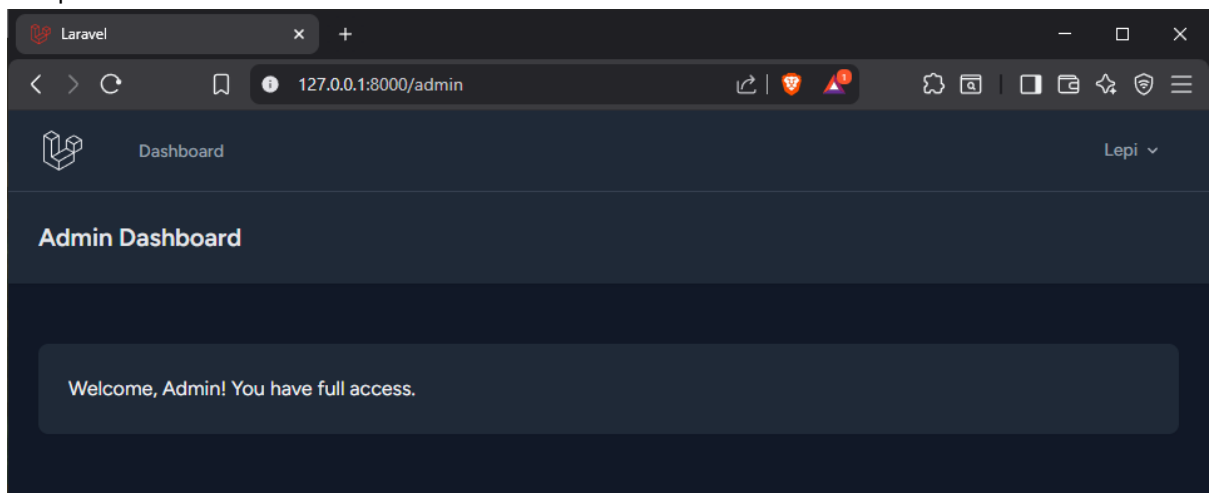
```

1      <?php
2
3      use App\Http\Controllers\ProfileController;
4      use Illuminate\Support\Facades\Route;
5
6      Route::get( uri: '/', function () {
7          return view( view: 'welcome');
8      });
9
10     Route::get( uri: '/dashboard', function () {
11         return view( view: 'dashboard');
12     })->middleware(['auth', 'verified'])->name( name: 'dashboard');
13
14     Route::middleware( middleware: 'auth')->group(function () {
15         Route::get( uri: '/profile', [ProfileController::class, 'edit'])->name( name: 'profile.edit');
16         Route::patch( uri: '/profile', [ProfileController::class, 'update'])->name( name: 'profile.update');
17         Route::delete( uri: '/profile', [ProfileController::class, 'destroy'])->name( name: 'profile.destroy');
18     });
19
20     Route::middleware(['auth'])->group(function () {
21         Route::get( uri: '/all', function () {
22             return view( view: 'all');
23         });
24
25         Route::get( uri: '/admin', function () {
26             return view( view: 'admin');
27         })->middleware( middleware: 'role:admin');
28
29         Route::get( uri: '/manager', function () {
30             return view( view: 'manager');
31         })->middleware( middleware: 'role:manager');
32
33         Route::get( uri: '/user', function () {
34             return view( view: 'user');
35         })->middleware( middleware: 'role:user');
36     });
37
38     require __DIR__.'/auth.php';

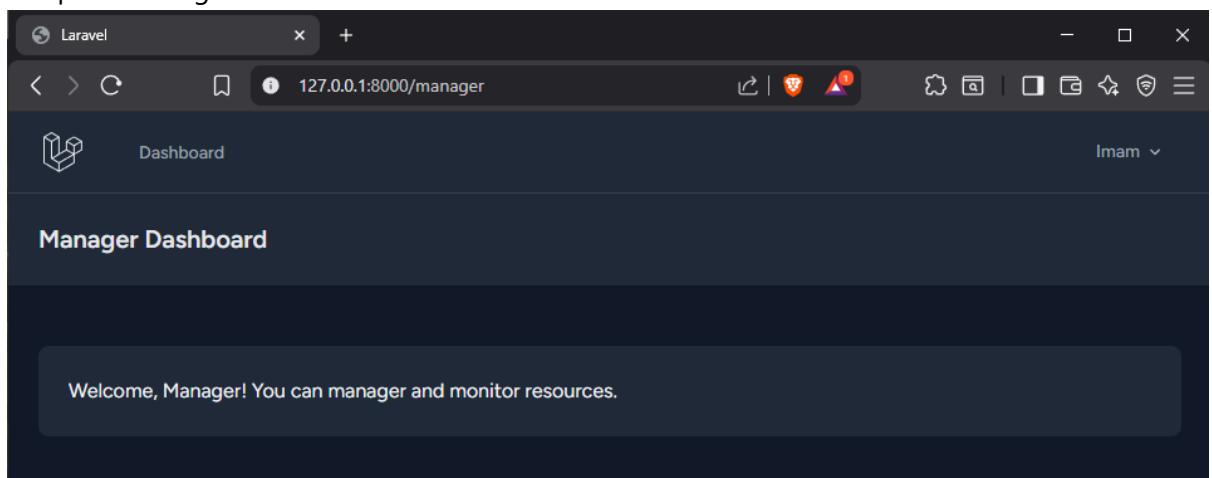
```

- Menjalankan aplikasi dan Menunjukkan hasil dibrowser.

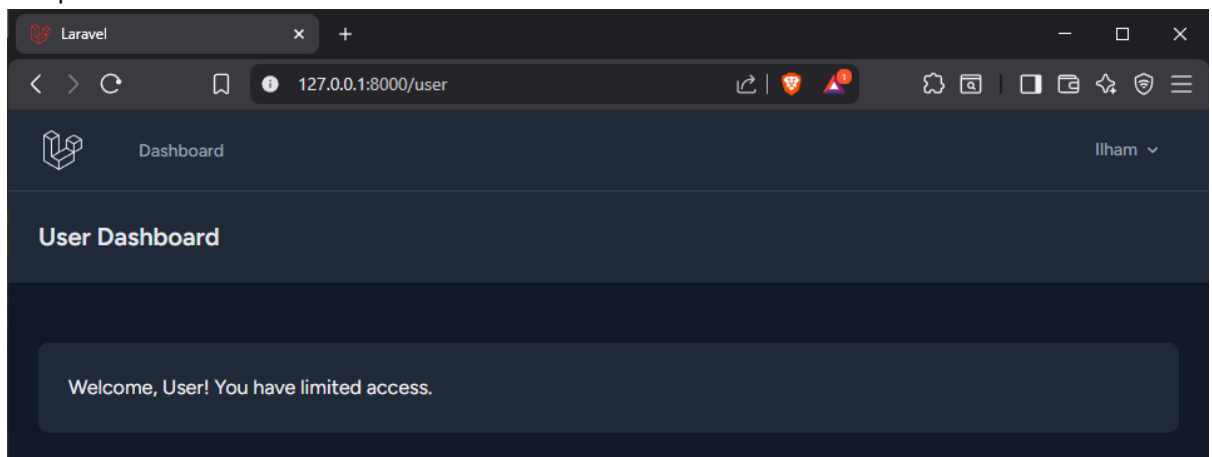
- Tampilan admin



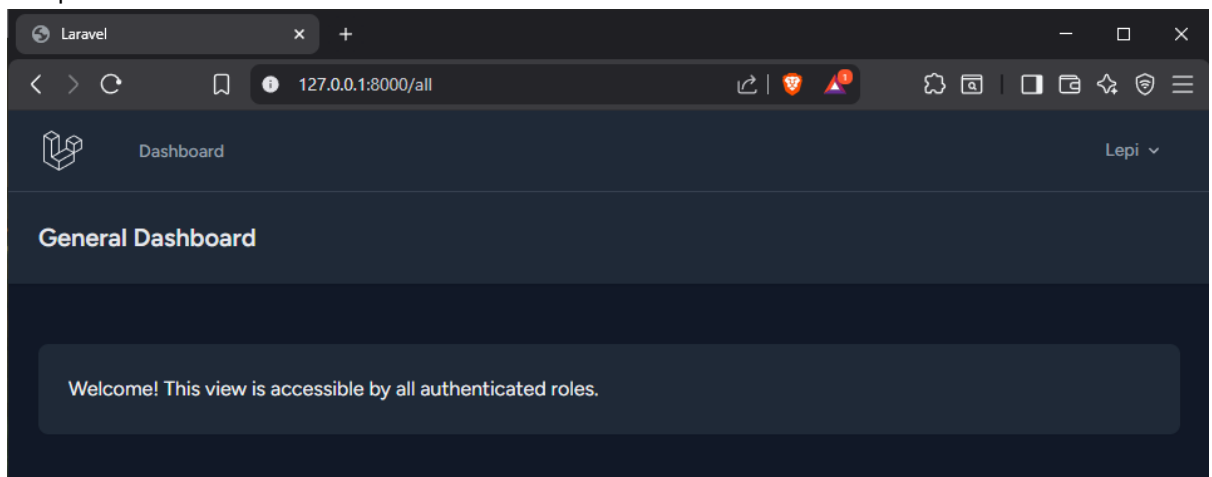
- Tampilan manager



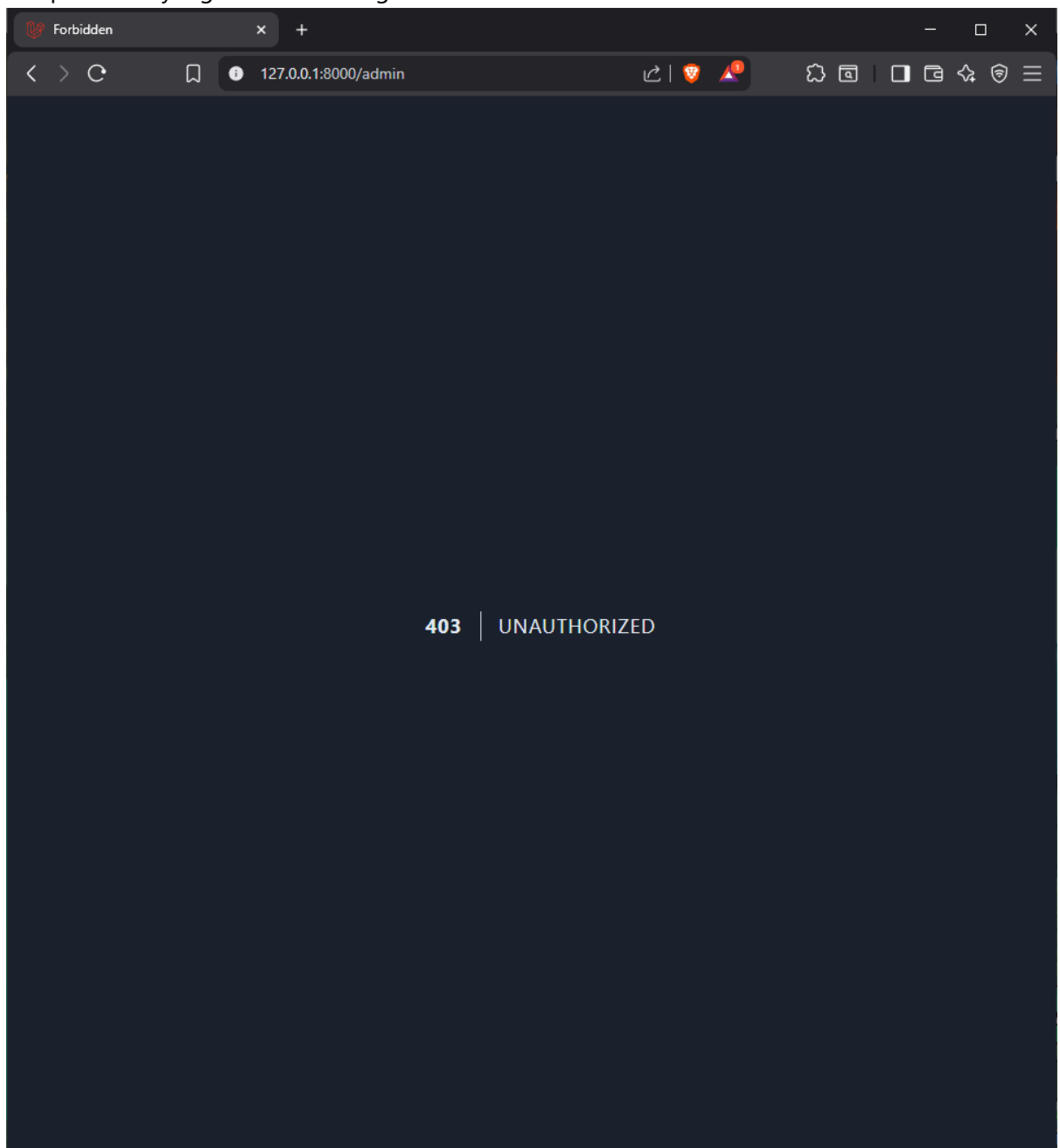
- Tampilan user



- Tampilan all



- Tampilan user yang mencoba mengakses situs admin



---

### 3. Hasil dan Pembahasan

Jelaskan apa hasil dari praktikum yang dilakukan.

- Apa Hasil dari Praktikum yang dilakukan?

Hasil dari praktikum ini adalah dua sistem keamanan yang terpisah namun komplementer, yang berjalan di Laravel:

- **Sistem Autentikasi Dasar Fungsional (Praktikum 1):** Berhasil menginstal dan mengkonfigurasi Laravel Breeze, menghasilkan alur kerja login dan register yang lengkap. Pengujian rute /myprofile yang dilindungi middleware auth membuktikan bahwa sistem secara efektif membatasi akses hanya untuk pengguna yang telah login, dan mengembalikan data pengguna yang terautentikasi (dalam format JSON).
- **Sistem Otorisasi RBAC Fungsional (Praktikum 2):** Berhasil membuat dan menerapkan kontrol akses berbasis peran. Hal ini ditunjukkan dengan keberhasilan pengguna admin, manager, dan user dalam mengakses rute umum (/all) dan rute khusus peran mereka (misalnya /admin hanya dapat diakses oleh Admin). Akses yang salah (misalnya Manager mencoba mengakses /admin) akan menghasilkan respons "403 Unauthorized" yang dikelola oleh RoleMiddleware kustom.

- Bagaimana Validasi Input Bekerja di Laravel?

Meskipun praktikum ini tidak secara eksplisit fokus pada validasi formulir, Laravel Breeze menyediakan validasi input saat proses registrasi:

- **Validasi Otomatis:** Ketika pengguna mendaftar (seperti di Praktikum 1), Laravel Breeze (melalui Controller yang dihasilkannya) secara otomatis menerapkan aturan validasi (required, email, unique, min:8 untuk password) pada input.
- **Keamanan Kata Sandi:** Input kata sandi dienkripsi menggunakan fungsi hashing bcrypt sebelum disimpan ke database, menjamin keamanan data sensitif meskipun database diretas.
- **Middleware & Keamanan:** Setelah login, keamanan lebih lanjut diperkuat oleh middleware auth (yang memeriksa Autentikasi) dan middleware role (yang memeriksa Otorisasi). Ini merupakan lapisan pengamanan yang terjadi setelah input awal divalidasi.

- Apa peran Masing-Masing Komponen (Route, Controller, View) dalam Program yang Dibuat?

Ketiga komponen ini berperan dalam menegakkan kebijakan keamanan dan menampilkan interface yang sesuai:

- **Route (Gerbang Keamanan):**

- **Peran:** Mendefinisikan jalur akses dan menentukan kebijakan keamanan yang harus diterapkan.
- **Aksi:** Menggunakan middleware chain. Contoh:
  - Praktikum 1: ->middleware('auth') untuk semua yang login.
  - Praktikum 2: ->middleware('role:admin') untuk membatasi akses hanya untuk peran admin. Route adalah penegak kebijakan keamanan pertama.

- **Controller (Pengelola Sesi dan Akses):**

- **Peran:** Mengelola proses login/logout (Autentikasi) dan menyiapkan data yang dilindungi (Otorisasi).

- **Aksi:** Controller Breeze menangani penyimpanan data pengguna baru. Di Praktikum 1, Controller digunakan untuk mendapatkan data pengguna terautentikasi (Auth::user()) untuk rute /myprofile.
  - **View (Antarmuka Pengguna):**
    - **Peran:** Bertanggung jawab penuh untuk merender struktur HTML dan menampilkan data yang dikirimkan oleh Controller. Menampilkan antarmuka yang berbeda berdasarkan status autentikasi dan otorisasi pengguna.
    - **Aksi:**
      - Menampilkan antarmuka yang berbeda berdasarkan status autentikasi dan otorisasi pengguna.
      - Menampilkan Admin Dashboard (admin.blade.php) atau Manager Dashboard (manager.blade.php) hanya ketika pengguna telah lolos pemeriksaan akses dari middleware yang sesuai.
- 

## 4. Kesimpulan

Praktikum ini berhasil mengimplementasikan sistem keamanan berlapis di Laravel, dimulai dari fondasi Autentikasi menggunakan Laravel Breeze, yang menyediakan mekanisme login, register, dan enkripsi kata sandi yang kuat. Kesuksesan ini dilanjutkan dengan implementasi sistem Otorisasi Berbasis Peran (RBAC) yang advanced. Melalui penambahan kolom role, seeding pengguna, dan pembuatan Middleware Kustom (RoleMiddleware), aplikasi secara efektif dapat membatasi akses pengguna (admin, manager, user) ke view dan rute yang hanya ditujukan bagi mereka, sementara rute umum tetap terbuka bagi semua yang terautentikasi. Secara keseluruhan, praktikum ini membuktikan bahwa arsitektur Laravel (dengan middleware dan kemampuan kustomisasi) sangat ideal untuk mengembangkan aplikasi dengan kebutuhan keamanan yang terstruktur, fleksibel, dan scalable.

---

## 5. Referensi

- Sumber dari :
    - Laraval 12 Training Kit: A Practical Guide to Modern Web Development. Link: <https://lnkd.in/gm6ms5cf>
    - BELAJAR LARAVEL Tutorial Framework Laravel Untuk Pemula by SANDHIKA GALIH. Link: <https://www.youtube.com/@sandhikagalihWPU>
    - Website Full Stack Open. Link: <https://fullstackopen.com/en/>
-