

Pertemuan 2

Modul 4: Setup Autentikasi Multi-Role

Tujuan Pembelajaran Modul 4 :

- Membuat controller dan mengimplementasikan fungsi login pengguna
- Memahami struktur login dan registrasi di Laravel menggunakan Blade
- Mengkonfigurasi middleware untuk membatasi akses berdasarkan role: Admin, Pasien, Dokter
- Mengatur redirect pengguna secara otomatis ke dashboard sesuai rolenya setelah login
- Mengetahui cara menggunakan template layout yang sudah disediakan
- Membedakan layout berdasarkan role yang sedang aktif
- Mencegah supaya user yang tidak sesuai role melihat route role lain
- Memahami konsep dari layouting dalam blade
- Mengetahui cara membuat halaman yang baik menggunakan blade.php

Setelah menyelesaikan modul ini, peserta diharapkan mampu:

1. **Mengimplementasikan routing resource di Laravel**
 - Menggunakan `Route::resource()` untuk memetakan operasi CRUD ke controller `PoliController`
 - Mengatur akses routing hanya untuk role `admin` dengan middleware.
 2. **Membangun tampilan form tambah, edit, dan daftar data Poli**
 - Membuat halaman `index`, `create`, dan `edit` untuk data Poli menggunakan komponen layout (`x-layouts.app`)
 - Menyediakan tombol aksi seperti **Tambah**, **Edit**, dan **Hapus** dalam tabel data
 3. **Melakukan validasi input data**
 - Validasi field `nama_poli` (required) dan `keterangan` (nullable) di sisi controller menggunakan `$request->validate()`.
 4. **Memberikan feedback kepada pengguna melalui notifikasi sukses/gagal**
 - Menampilkan notifikasi (alert) saat data berhasil ditambah, diubah, atau dihapus.
 - Notifikasi muncul di halaman index, dengan gaya visual berbeda untuk sukses (hijau) dan gagal (merah).
 - Alert otomatis menghilang setelah beberapa detik menggunakan JavaScript.
 5. **Mengorganisasi tampilan admin dengan struktur layout sidebar**
 - Navigasi sidebar menampilkan menu "Dashboard Admin" dan "Manajemen Poli"
-



Overview :

Dalam implementasi login berbasis role di Laravel, langkah pertama adalah membuat controller khusus autentikasi yang berfungsi menangani proses login, logout, dan validasi data pengguna. Proses login dilakukan dengan memverifikasi email atau username dan password, kemudian mengautentikasi pengguna menggunakan sistem bawaan Laravel. Untuk antarmuka, Laravel menyediakan Blade Template Engine yang memudahkan pembuatan tampilan form login dan registrasi, sekaligus menghubungkannya dengan route dan controller. Agar dapat membedakan jenis pengguna, tabel `users` perlu ditambahkan kolom `role` melalui migration.

Nilai role ini digunakan untuk mengatur level akses, misalnya `admin`, `dokter`, dan `pasien`. Selanjutnya, middleware dikonfigurasi untuk memastikan setiap pengguna hanya dapat mengakses halaman sesuai dengan perannya. Dengan begitu, admin hanya bisa masuk ke halaman manajemen data, dokter ke halaman konsultasi, dan pasien ke halaman layanan pasien. Terakhir, setelah proses login berhasil, pengguna secara otomatis diarahkan (redirect) ke dashboard sesuai dengan role masing-masing sehingga alur aplikasi menjadi lebih aman, terstruktur, dan sesuai kebutuhan pengguna.

4.1 Konsep Autentikasi dan Multi-Role

4.1.1 Apa itu Autentikasi?

Autentikasi adalah proses **verifikasi identitas pengguna** sebelum mengakses aplikasi. Laravel menyediakan sistem autentikasi bawaan untuk login, logout, dan registrasi, yang bisa dikustomisasi sesuai kebutuhan aplikasi.

4.1.2 Apa itu Multi-Role?

Multi-role adalah mekanisme autentikasi yang memungkinkan aplikasi membedakan jenis pengguna berdasarkan peran, seperti:

- **Admin** – akses penuh ke seluruh data dan manajemen sistem
- **Dokter** – akses ke data pasien, jadwal periksa, dan hasil pemeriksaan
- **Pasien** – akses ke pendaftaran, jadwal, dan histori pemeriksaan

Setiap role akan diarahkan ke halaman dashboard yang berbeda setelah login.

4.2 Struktur Login dan Registrasi di Laravel

Laravel memiliki fitur `Auth` yang menangani login dan registrasi secara otomatis. Kita bisa menggunakan `Auth::attempt()` untuk melakukan pengecekan kredensial pengguna.

Jika ingin lebih fleksibel (seperti mengatur redirect berdasarkan role), kita bisa membuat controller login sendiri.

4.3 Langkah-Langkah Setup Autentikasi Multi-Role

4.3.1 Pembuatan Controller & Fungsi Login/Register

Untuk menangani logika autentikasi multi-role, kita buat controller `AuthController`.

Buat Controller

[TERMINAL]

```
php artisan make:controller AuthController
```

Isi `AuthController.php`

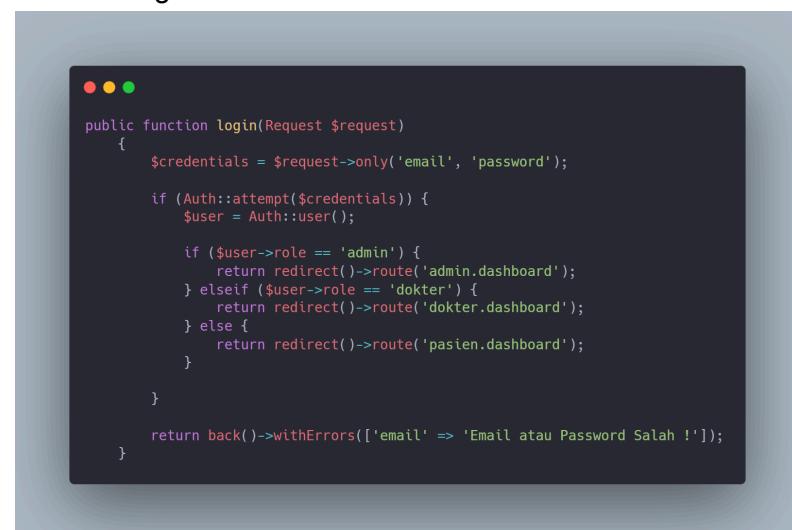
`app/Http/Controllers/AuthController.php`

Function `showLogin`



```
public function showLogin()
{
    return view('auth.login');
}
```

Function `login`



```
public function login(Request $request)
{
    $credentials = $request->only('email', 'password');

    if (Auth::attempt($credentials)) {
        $user = Auth::user();

        if ($user->role == 'admin') {
            return redirect()->route('admin.dashboard');
        } elseif ($user->role == 'dokter') {
            return redirect()->route('dokter.dashboard');
        } else {
            return redirect()->route('pasien.dashboard');
        }
    }

    return back()->withErrors(['email' => 'Email atau Password Salah !']);
}
```

- `$credentials = $request->only('email', 'password');`
`$request->only('email','password')`: Mengambil hanya input `email` dan `password` dari form login.
- if (Auth::attempt(\$credentials)) {

`$user = Auth::user();`
 - `Auth::attempt($credentials)`: Mengecek apakah email & password sesuai dengan data `users` di database.
 Jika benar → user dianggap **terautentikasi**.
- Auth::user(): Mengambil data user yang sedang login.
- return back()->withErrors(['email' => 'Email atau Password Salah !']);
 Jika `Auth::attempt` gagal, kembali ke halaman login dengan pesan error.

Function Register



```

public function register(Request $request)
{
    $request->validate([
        'nama' => ['required', 'string', 'max:255'],
        'alamat' => ['required', 'string', 'max:255'],
        'no_ktp' => ['required', 'string', 'max:30'],
        'no_hp' => ['required', 'string', 'max:20'],
        'email' => ['required', 'string', 'email', 'max:255', 'unique:users,email'],
        'password' => ['required', 'confirmed'],
    ]);

    User::create([
        'nama' => $request->nama,
        'alamat' => $request->alamat,
        'no_ktp' => $request->no_ktp,
        'no_hp' => $request->no_hp,
        'email' => $request->email,
        'password' => Hash::make($request->password),
        'role' => 'pasien',
    ]);

    return redirect()->route('login');
}

```

Validasi input:

- `required` → wajib diisi.
- `unique:users,email` → email harus unik di tabel `users`.
- `confirmed` → password harus sama dengan `password_confirmation`.

Menyimpan data user baru:

- Password **di-hash** agar aman.
- Role default → **pasien**.

`return redirect()->route('login');`

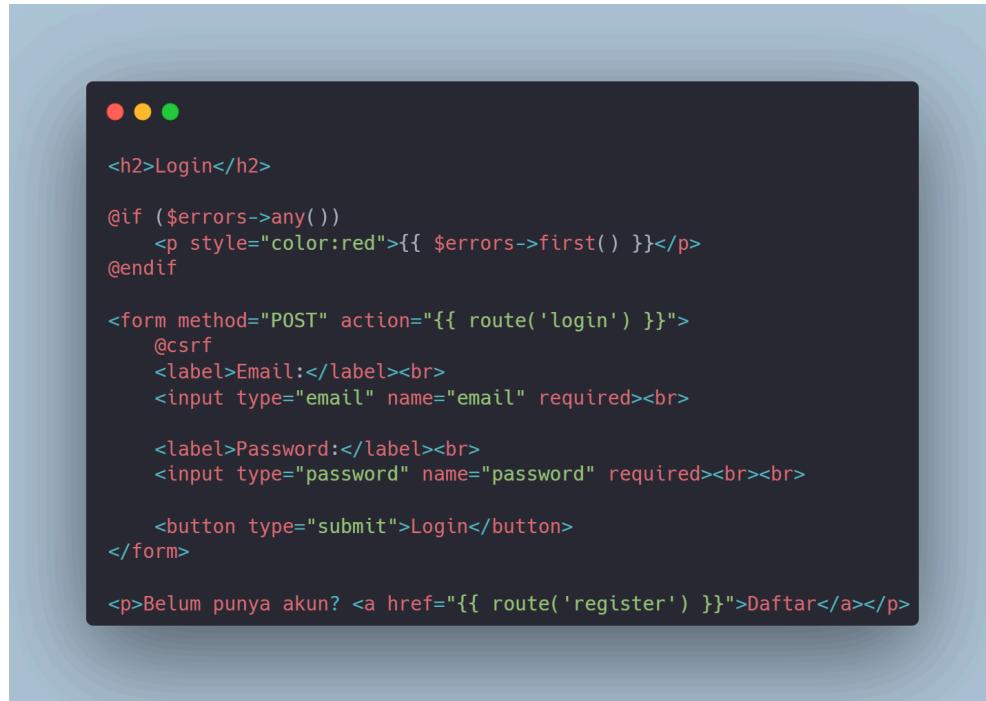
- Setelah register, diarahkan ke halaman login

4.3.2 Struktur Login & Register (Blade Template)

Membuat tampilan antarmuka untuk login dan registrasi menggunakan **Blade Template**.

Isi login.blade.php

resources/views/auth/login.blade.php



```
<h2>Login</h2>

@if ($errors->any())
    <p style="color:red">{{ $errors->first() }}</p>
@endif

<form method="POST" action="{{ route('login') }}">
    @csrf
    <label>Email:</label><br>
    <input type="email" name="email" required><br>

    <label>Password:</label><br>
    <input type="password" name="password" required><br><br>

    <button type="submit">Login</button>
</form>

<p>Belum punya akun? <a href="{{ route('register') }}">Daftar</a></p>
```

Isi register.blade.php

resources/views/auth/register.blade.php



```
<h2>Register</h2>

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

<form method="POST" action="{{ route('register') }}">
    @csrf
    <label>Nama Lengkap:</label><br>
    <input type="text" name="nama" required><br>

    <label>Email:</label><br>
    <input type="email" name="email" required><br>

    <label>Alamat:</label><br>
    <input type="text" name="alamat" required><br>

    <label>No HP:</label><br>
    <input type="text" name="no_hp" required><br>

    <label>No KTP:</label><br>
    <input type="text" name="no_ktp" required><br>

    <label>Password:</label><br>
    <input type="password" name="password" required><br>

    <label>Konfirmasi Password:</label><br>
    <input type="password" name="password_confirmation" required><br><br>

    <button type="submit">Daftar</button>
</form>

<p>Sudah punya akun? <a href="{{ route('login') }}">Login</a></p>
```

4.3.3 Setup Middleware Role

Untuk membatasi akses berdasarkan role pengguna, kita akan membuat middleware khusus. **Middleware** ini akan memverifikasi apakah pengguna memiliki role yang sesuai sebelum diberikan akses ke halaman tertentu (misalnya admin hanya boleh mengakses dashboard admin, dst).

`app/Http/Middleware/RoleMiddleware.php`

```
<?php

namespace App\Http\Middleware;

use Closure;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Symfony\Component\HttpFoundation\Response;

class RoleMiddleware
{

    public function handle(Request $request, Closure $next, $role): Response
    {
        $user = Auth::user();

        if ($user->role !== $role) {
            return response('Unauthorized', 403);
        }

        return $next($request);
    }
}
```

Untuk menambahkan middleware khusus ke dalam `bootstrap/app.php`, kita perlu mendaftarkan middleware tersebut agar bisa dipakai di route. Proses ini memungkinkan kita memberi **alias** sederhana (misalnya `role`) untuk class middleware yang sudah kita buat. Dengan alias ini, kita bisa memanggil middleware langsung di file route tanpa harus menulis path class yang panjang.

bootstrap/app.php

```
<?php

use App\Http\Middleware\RoleMiddleware;
use Illuminate\Foundation\Application;
use Illuminate\Foundation\Configuration\Exceptions;
use Illuminate\Foundation\Configuration\Middleware;

return Application::configure(basePath: dirname(__DIR__))
    ->withRouting(
        web: __DIR__.'/../routes/web.php',
        commands: __DIR__.'/../routes/console.php',
        health: '/up',
    )
    ->withMiddleware(function (Middleware $middleware): void {
        $middleware->alias([
            'role' => RoleMiddleware::class
        ]);
    })
    ->withExceptions(function (Exceptions $exceptions): void {
        //
    })->create();
```

4.3.4 Setting Routing

Setelah membuat controller, tampilan Blade, dan middleware, langkah berikutnya adalah mengatur route pada file `routes/web.php`. Ini mengatur route untuk proses login, register, logout, serta dashboard berdasarkan role pengguna yang telah dilindungi oleh middleware.

Isi `web.php`

`routes/web.php`

```
Route::get('/', function () {
    return view('welcome');
});
```

Menampilkan halaman `welcome.blade.php` saat pengguna mengakses root (/) aplikasi. Kemudian route nya diubah menjadi code seperti berikut

Route Autentikasi Umum !!

```
Route::get('/login', [AuthController::class, 'showLogin'])->name('login');
Route::post('/login', [AuthController::class, 'login']);
Route::get('/register', [AuthController::class, 'showRegister'])->name('register');
Route::post('/register', [AuthController::class, 'register']);
Route::post('/logout', [AuthController::class, 'logout']);
```

- Menampilkan dan memproses form login & register
- Menangani logout pengguna

Route Dashboard Berdasarkan Role

Setiap role diarahkan ke dashboard-nya masing-masing dan dilindungi oleh middleware **auth** dan **role**.

Admin

```
Route::middleware(['auth', 'role:admin'])->prefix('admin')->group(function () {
    Route::get('/dashboard', function () {
        return view('admin.dashboard');
    })->name('admin.dashboard');
});
```

Dokter

```
Route::middleware(['auth', 'role:dokter'])->prefix('dokter')->group(function () {
    Route::get('/dashboard', function () {
        return view('dokter.dashboard');
    })->name('dokter.dashboard');
});
```

Pasien

```
Route::middleware(['auth', 'role:pasien'])->prefix('pasien')->group(function () {
    Route::get('/dashboard', function () {
        return view('pasien.dashboard');
    })->name('pasien.dashboard');
});
```

4.3.5 Buat Dashboard

Setelah pengguna berhasil login, mereka akan diarahkan ke **halaman dashboard sesuai dengan role-nya**. Oleh karena itu, kita perlu membuat tampilan dashboard terpisah untuk masing-masing role: **admin**, **dokter**, dan **pasien**.

📝 Isi `admin/dashboard.blade.php`

```
resources/views/admin/dashboard.blade.php
```



```
<h1>Selamat Datang Admin</h1>
<form method="POST" action="/logout">@csrf <button>Logout</button></form>
```

📝 Isi

```
resources/views/dokter/dashboard.blade.php
```



```
<h1>Selamat Datang Dokter</h1>
<form method="POST" action="/logout">@csrf <button>Logout</button></form>
```

📝 Isi `pasien/dashboard.blade.php`

```
resources/views/pasien/dashboard.blade.php
```



```
<h1>Selamat Datang Pasien</h1>
<form method="POST" action="/logout">@csrf <button>Logout</button></form>
```

4.3.6 Buat User Seeder

Seeder digunakan untuk mengisi data awal pada database secara otomatis. Pada modul ini, kita akan membuat **UserSeeder** untuk mengisi data pengguna dengan berbagai role: **admin**, **dokter**, dan **pasien**.

🔧 Buat Seeder

```
[TERMINAL]
```

```
php artisan make:seeder UserSeeder
```

Isi UserSeeder.php

database/seeders/UserSeeder.php

```
● ● ●

class UserSeeder extends Seeder
{
    /**
     * Run the database seeds.
     */
    public function run(): void
    {
        $users = [
            [
                'nama' => 'Admin',
                'email' => 'admin@gmail.com',
                'password' => Hash::make('admin'),
                'role' => 'admin',
            ],
            [
                'nama' => 'Dokter',
                'email' => 'dokter@gmail.com',
                'password' => Hash::make('dokter'),
                'role' => 'dokter',
            ],
        ];
        foreach ($users as $user) {
            User::create($user);
        }
    }
}
```

Daftarkan di DatabaseSeeder.php

Tambahkan UserSeeder di file database/seeders/DatabaseSeeder.php

```
class DatabaseSeeder extends Seeder
{
    public function run(): void
    {
        $this->call([
            UserSeeder::class,
        ]);
    }
}
```

► Jalankan Seeder

Untuk mengisi data ke database, jalankan perintah:

[TERMINAL]

```
php artisan db:seed
```

Setelah berhasil, akan ada tiga user default di tabel `users` dengan role yang berbeda.



Modul 5: Memasang template Admin LTE



5.0 Overview

Modul ini memberikan panduan langkah demi langkah untuk mengintegrasikan template AdminLTE ke dalam sebuah proyek Laravel. Fokus utamanya adalah pada konsep layouting di dalam Blade, yaitu sebuah teknik untuk memecah struktur HTML menjadi komponen-komponen yang dapat digunakan kembali seperti sidebar, header, dan footer agar tampilan aplikasi web menjadi lebih terstruktur, konsisten, dan mudah dikembangkan dalam jangka panjang.

Beberapa poin penting dalam modul:

1. Konsep Layouting di Laravel Blade

- Menyusun struktur HTML agar bisa dipisah (header, sidebar, footer) sehingga tidak perlu menulis ulang.
- Prinsip DRY (*Don't Repeat Yourself*).

2. Struktur Layouting

- Menggunakan `app.blade.php` sebagai kerangka utama.
- Komponen lain seperti header, sidebar, dan footer dipanggil dari file terpisah.

3. Penggunaan CDN (Content Delivery Network)

- Memahami apa itu CDN dan mengapa bermanfaat dalam pengembangan.
- Menggunakan CDN agar tidak perlu meng-install library manual.

4. Pemasangan Template Admin

- Membuat komponen (header, sidebar, footer).
- Membuat layout utama (`app.blade.php`).
- Membedakan dashboard berdasarkan **role** (admin, dokter, pasien).
- Menjalankan aplikasi untuk mengecek hasil.

5. Template Login & Register

- Membuat layout berbeda untuk halaman login/register (`guest.blade.php`).

- Memastikan tampilan login & register berbeda dengan dashboard utama.
-

5.1: Apa itu layouting dalam laravel?

5.1.1: Definisi layouting

layouting adalah cara menyusun tampilan (template) agar struktur HTML bisa dipisah-pisah dan digunakan kembali secara efisien. Tujuannya Supaya kamu **tidak menulis ulang** bagian-bagian HTML yang sama di banyak file (seperti `<head>`, sidebar, navbar, footer).

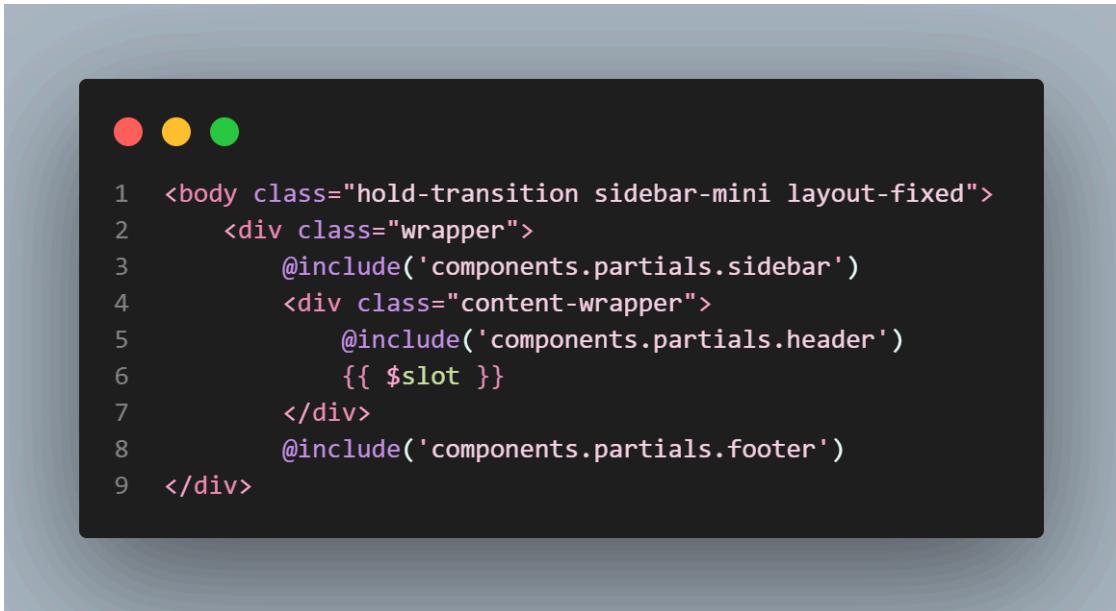
5.1.2: Manfaat layouting

Dengan membuat layouting dalam laravel blade, banyak manfaat yang bisa didapatkan daripada membuat satu persatu:

- DRY (Don't Repeat Yourself): Sidebar, header, footer cukup ditulis sekali.
- Konsistensi UI: Semua halaman mempunyai struktur yang sama.
- Fleksibilitas: Kamu bisa ubah sidebar atau footer di satu tempat, dan semua halaman ikut berubah.

5.2: Struktur layouting dalam laravel

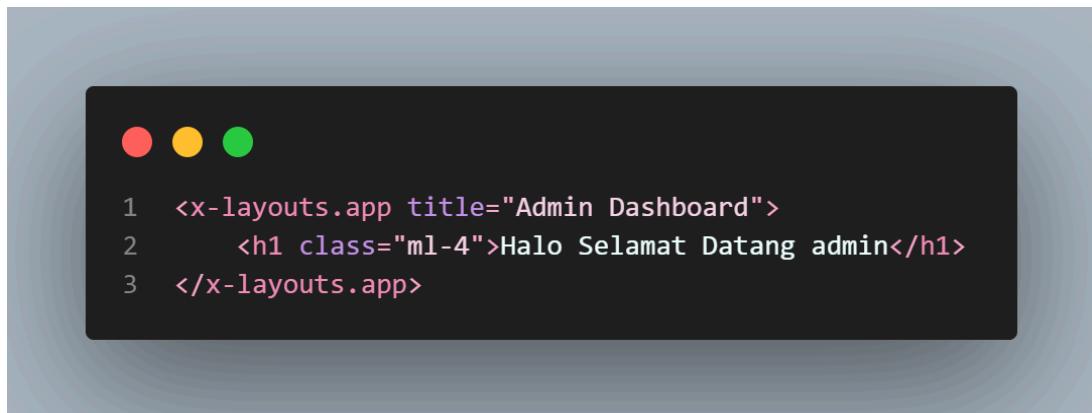
Membuat layouting dalam laravel blade. Bisa dilakukan dengan source code yang sederhana, perhatikan source code berikut:



```
1 <body class="hold-transition sidebar-mini layout-fixed">
2   <div class="wrapper">
3     @include('components.partials.sidebar')
4     <div class="content-wrapper">
5       @include('components.partials.header')
6       {{ $slot }}
7     </div>
8     @include('components.partials.footer')
9   </div>
```

Contoh diatas adalah salah satu cara untuk membuat template dalam app.blade.php. Kamu bisa menggunakan dalam blade php dengan cara menggunakannya didalam kontainer

<x-layouts.app></x-layouts.app> berikut adalah caranya untuk menggunakan template di page dashboard.blade.php



Dengan menggunakan source code diatas, anda sudah menggunakan template dari yang dibuat.

5.3: Konsep CDN (Content Delivery Network)

Dalam modul ini kita akan menggunakan CDN untuk memudahkan proses pengembangan, namun CDN itu apa sih? mengapa harus menggunakan CDN?

5.3.1: Definisi CDN

CDN (Content Delivery Network) adalah jaringan server yang tersebar di berbagai lokasi geografis, yang digunakan untuk menyimpan dan mengirimkan konten website seperti gambar, video, file JavaScript, CSS, dan lainnya secara cepat kepada pengguna.

5.3.2: Mengapa harus menggunakan CDN?

Daripada harus menginstall dari library yang diperlukan, menggunakan CDN akan mempercepat proses pengembangannya, karna tidak perlu menginstall library yang diperlukan. Library yang dibutuhkan akan langsung bekerja dengan menggunakan CDN pada *parent* dari struktur frontend.

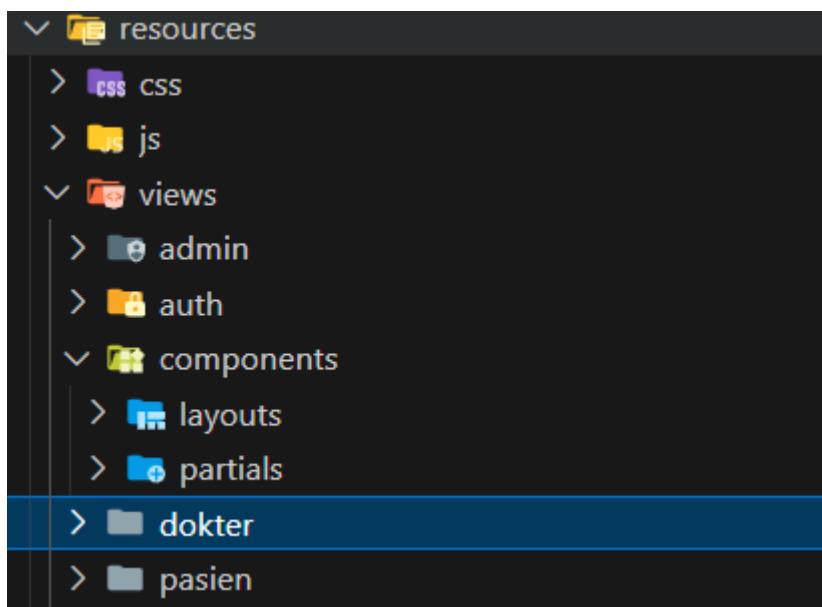
5.4: Memasang template admin

Dibawah ini adalah cara untuk memasang template pada masing masing halaman pada setiap role sehingga bisa memudahkan dalam pengembangan dalam jangka waktu panjang

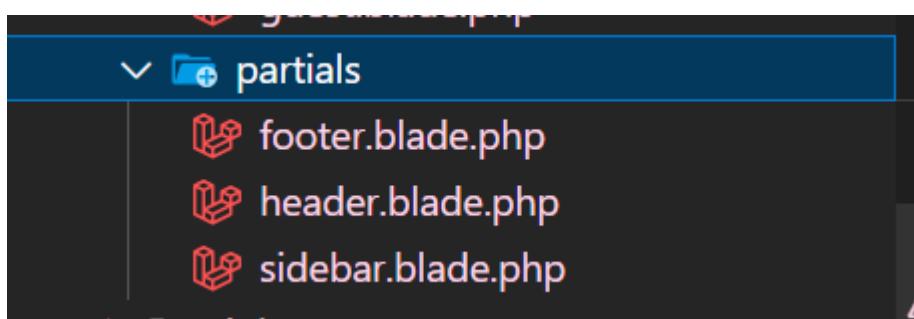
5.4.1: Membuat components yang diperlukan

Untuk bisa menggunakan memasang template silahkan untuk membuat komponen komponen yang diperlukan supaya bisa dipanggil di app.blade.php. Didalam direktori resources/views silahkan membuat direktori components, dalam components silahkan membuat direktori partials. Sehingga struktur 3

direktorinya menjadi seperti berikut:



Kemudian buat 3 file blade.php didalam direktori partials



- isi dari header.blade.php

```
1  <nav class="navbar navbar-expand navbar-white navbar-light">
2      <!-- Left navbar Links -->
3      <ul class="navbar-nav">
4          <li class="nav-item">
5              <a class="nav-link" data-widget="pushmenu" href="#" role="button">
6                  <i class="fas fa-bars"></i>
7              </a>
8          </li>
9      </ul>
10
11     <li class="nav-item d-none d-sm-inline-block">
12         <a href="#" class="nav-link">Home</a>
13     </li>
14     <li class="nav-item d-none d-sm-inline-block">
15         <a href="#" class="nav-link">Contact</a>
16     </li>
17 </ul>
18
19     <!-- Right navbar Links -->
20     <ul class="navbar-nav ml-auto">
21         <!-- Navbar Search -->
22         <li class="nav-item">
23             <a class="nav-link" data-widget="navbar-search" href="#" role="button">
24                 <i class="fas fa-search"></i>
25             </a>
26             <div class="navbar-search-block">
27                 <form class="form-inline">
28                     <div class="input-group input-group-sm">
29                         <input class="form-control form-control-navbar" type="search" placeholder="Search"
30                             aria-label="Search">
31                         <div class="input-group-append">
32                             <button class="btn btn-navbar" type="submit">
33                                 <i class="fas fa-search"></i>
34                             </button>
35                             <button class="btn btn-navbar" type="button" data-widget="navbar-search">
36                                 <i class="fas fa-times"></i>
37                             </button>
38                         </div>
39                     </div>
40                 </form>
41             </div>
42         </li>
43
44         <!-- Fullscreen Button -->
45         <li class="nav-item">
46             <a class="nav-link" data-widget="fullscreen" href="#" role="button">
47                 <i class="fas fa-expand-arrows-alt"></i>
48             </a>
49         </li>
50     </ul>
51 </nav>
52
```

- isi dari sidebar.blade.php

untuk lebih jelasnya sidebar

<https://drive.google.com/file/d/1vopkY2AMM0WTzEc0rnPblvUsNX5VFSsH/view?usp=sharing>

- isi dari footer.blade.php



```
1 <footer class="main-footer">
2     <strong>Copyright © 2024 <a href="#">Poliklinik 2024</a>.</strong>
3     All right reserved.
4 </footer>
5
```

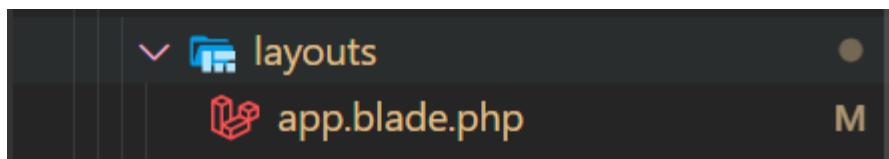
5.4.3: Membuat layout

Setelah 3 component sudah dibuat, kemudian dalam direktori components buat direktori layouts. Kemudian didalamnya buat file app.blade.php, dan isi dengan source code berikut

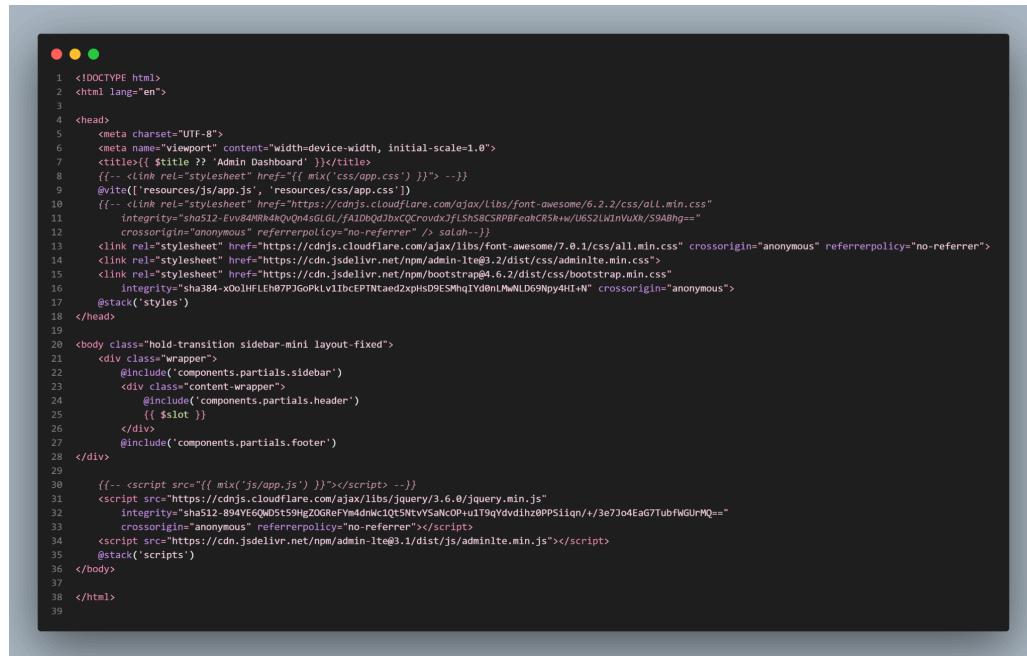
Pada layout utama app.blade.php, asset aplikasi dikelola menggunakan **Vite** melalui directive @vite. Dengan cara ini, file resources/css/app.css dan resources/js/app.js akan otomatis dikompilasi saat development dengan perintah npm run dev.

Selain itu, library pihak ketiga seperti **AdminLTE** dan **CDN**, sehingga kita bisa menggabungkan keunggulan Vite (untuk asset internal) dan CDN (untuk library eksternal).

Dengan kombinasi ini, pengembangan menjadi lebih cepat, konsisten, dan mudah dioptimasi untuk production dengan npm run build.



Isi dari app.blade.php



```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5     <meta charset="UTF-8">
6     <meta name="viewport" content="width=device-width, initial-scale=1.0">
7     <title>{{ $title ?? 'Admin Dashboard' }}</title>
8     {{-- <im rel="stylesheet" href="{{ mix('css/app.css') }}" --}}
9     @vite(['resources/js/app.js', 'resources/css/app.css'])
10    {{-- <im rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.2.2/css/all.min.css"
11        integrity="sha512-Ev84M4R4QjQn4sGLGL/FADbQdJbxQCrodxJfLSHSCSRPBFeabCR5k+w/U6S2LW1nVuXr/S9ABhj==" --}}
12    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/7.0.1/css/all.min.css" crossorigin="anonymous" referrerPolicy="no-referrer">
13    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/admin-lte@3.2/dist/css/adminlte.min.css">
14    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/css/bootstrap.min.css"
15        integrity="sha384-xOIHfEh07PJGopKlv1IbcEPtNtaed2xphSD9EMhqIYd0NLm2NLd69Npy4HI+N" crossorigin="anonymous">
16
17    @stack('styles')
18
19
20 <body class="hold-transition sidebar-mini layout-fixed">
21     <div class="wrapper">
22         @include('components.partials.sidebar')
23         <div class="content-wrapper">
24             @include('components.partials.header')
25             {{ $slot }}
26         </div>
27         @include('components.partials.footer')
28     </div>
29
30     {{-- <script src="{{ mix('js/app.js') }}" --}}</script>
31     <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.6.0/jquery.min.js"
32         integrity="sha512-894yE6QD5t59hgZ0GReYadmC1Qt5NcvSaNC0P+u1T9qYvdin2PPSi1qn+/3e7J04EaG7TubfWGURMQ==" --}}
33     <script src="https://cdn.jsdelivr.net/npm/admin-lte@3.1/dist/js/adminlte.min.js"></script>
34
35     @stack('scripts')
36
37
38 </html>
39
```

Link code :

https://drive.google.com/file/d/1d_5AU_bj28Vrl0AtH5oDcW1apcwQnPb0/view?usp=sharing

Disinilah akan menggunakan CDN, CDN seharusnya digunakan pada parent, sehingga komponen child akan menggunakan dari CDN nya juga tanpa perlu memasang CDN untuk setiap file.

5.4.4: Mengganti isi dari setiap dashboard antar role.

Ganti dari setiap isi dari dashboard.blade.php untuk setiap role pada views, supaya layout nya bisa terpanggil pada setiap halaman dashboard

Isi dari admin/dashboard.blade.php



```
1 <x-layouts.app title="Admin Dashboard">
2     <h1 class="ml-4">Halo Selamat Datang admin</h1>
3 </x-layouts.app>
4
```

Isi dari dokter/dashboard.blade.php



```
1 <x-layouts.app>
2     <h1 class="ml-4 mt-2">Selamat Datang Di Halaman Dokter</h1>
3 </x-layouts.app>
4
```

Isi dari pasien/dashboard.blade.php



```
1 <x-layouts.app title="Pasien">
2     <h1 class="ml-4">Halo Selamat Datang pasien</h1>
3 </x-layouts.app>
4
```

⌚ 5.4.5: Cek hasil tampilan

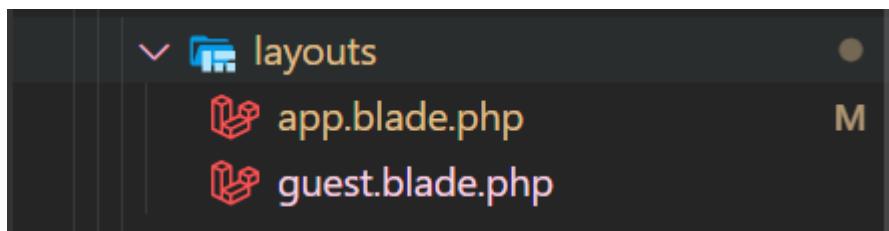
Setelah sudah silahkan ketik pada terminal “php artisan serve” untuk mengecek hasil dari setiap dashboard yang baru kamu buat

📄 5.5: Memasang template Login dan register

Berikutnya adalah memasang template untuk halaman login dan register. Mengapa harus menggunakan template yang berbeda? karena struktur layout antara halaman login/register.blade.php dan halaman dashboard.blade.php berbeda.

🛠 5.5.1: Membuat struktur layout

Didalam direktori layouts buat file guest.blade.php. Kemudian masukkan source code berikut:



isi file guest.blade.php

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1">
6   <title>{{ $title ?? 'Login' }}</title>
7
8 {{-- <Link rel="stylesheet" href="{{ mix('css/app.css') }}" --}}
9 @vite(['resources/js/app.js', 'resources/css/app.css'])
10 <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.1/css/all.min.css"
11   integrity="sha512-evv8wr4ngXGN5HgL/f7a1Qb0yQz0YrcdwrJThBS1CR7PB8cF51Ck+w/U6sJ4MztVlXv9Kvs9XAH9bg=="'
12   crossorigin="anonymous" referrerpolicy="no-referrer" />
13 <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/admin-lte@3.2/dist/css/adminlte.min.css">
14 <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/css/bootstrap.min.css"
15   integrity="sha384-aoKWFIE1NP7g0PKvlubICEPTNea2dzpi95cMHq1GnbVLnLD6Np941vN4--"
16   crossorigin="anonymous">
17
18   @stack('styles')
19 </head>
20
21 <body class="hold-transition sidebar-mini layout-fixed min-vh-100">
22   {{ $slot }}
23
24 {{-- <script src="{{ mix('js/app.js') }}" --}}
25 <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.6.0/jquery.min.js"
26   integrity="sha512-894YE6QWh5J0vNi6U06Bjr3D7tG6CkqbZo5smXKp4YfRvH+8abTE1Pi6jizoUczdlz7YT1Nx1B4ez7406+vEA=="
27   crossorigin="anonymous" referrerpolicy="no-referrer"></script>
28 <script src="https://cdn.jsdelivr.net/npm/admin-lte@3.1/dist/js/adminlte.min.js"></script>
29
30   @stack('scripts')
31 </body>
32 </html>
33
```

Link code :

<https://drive.google.com/file/d/1YmYzLg4RquXsI3o3kDUxjxHw-mmw6tu2/view?usp=sharing>

5.5.2: Mengganti login dan register

Ganti seluruh source code login dan register dengan source code berikut

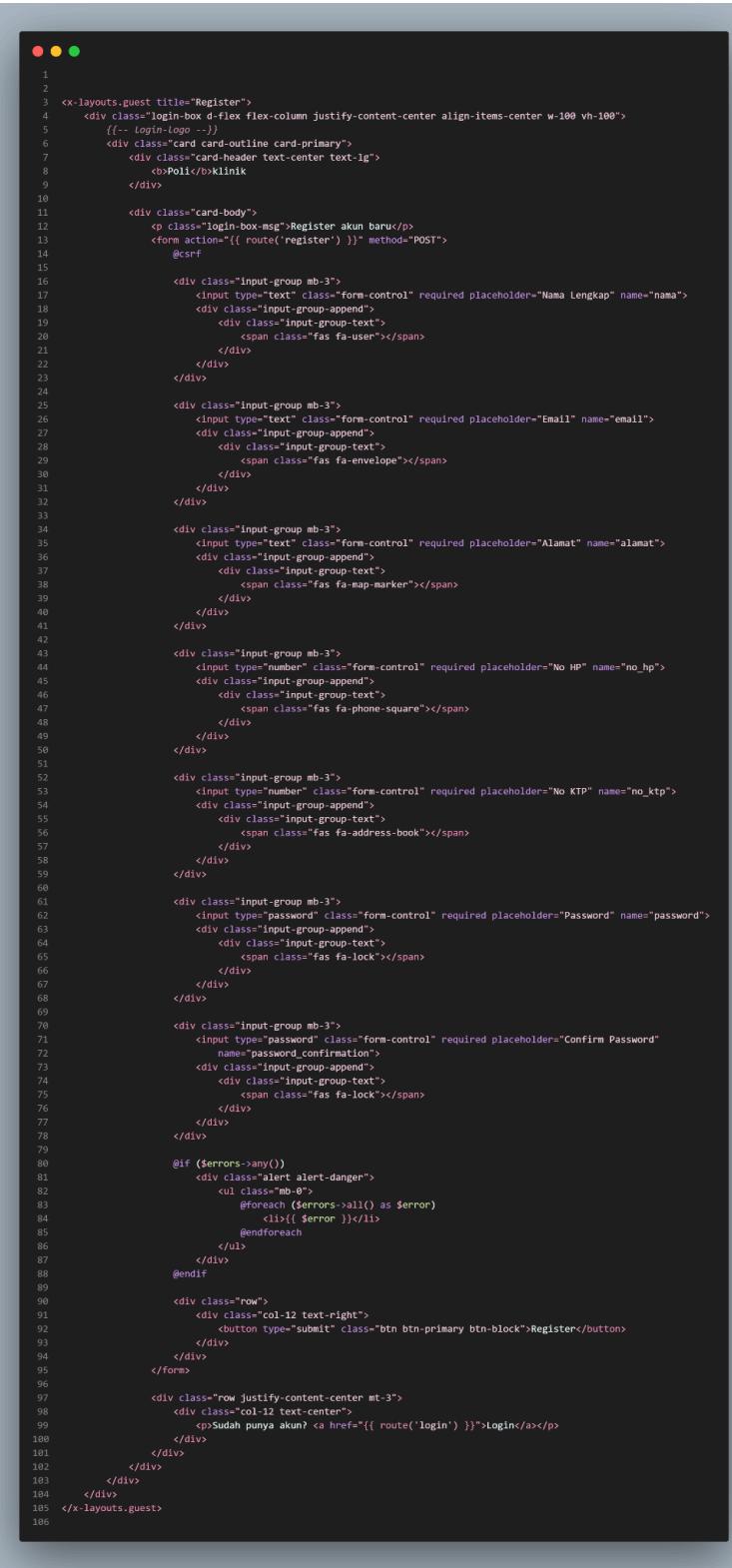
Isi dari login.blade.php

```
1  {{-- <h2>Login</h2> --}}
2  {{--
3      @if ($errors->any())
4          <p style="color:red">{{ $errors->first() }}</p>
5      @endif --}}
6
7  {{-- <form method="POST" action="{{ route('Login') }}" --}
8      @csrf
9      <Label>Email:</Label><br>
10     <input type="email" name="email" required><br>
11
12     <Label>Password:</Label><br>
13     <input type="password" name="password" required><br><br>
14
15     <button type="submit">Login</button>
16  </form> --}}
17
18 <x-layouts.guest title="Login">
19     <div class="login-box d-flex flex-column justify-content-center align-items-center w-100 vh-100">
20         <div class="card card-outline card-primary">
21             <div class="card-header text-center text-lg">
22                 <b>Poli</b>klinik
23             </div>
24             <div class="card-body">
25                 <p class="login-box-msg">Login ke akun anda</p>
26                 <form action="{{ route('login') }}" method="POST">
27                     @csrf
28                     <div class="input-group mb-3">
29                         <input type="text" class="form-control" placeholder="Email" name="email">
30                         <div class="input-group-append">
31                             <div class="input-group-text">
32                                 <span class="fas fa-envelope"></span>
33                             </div>
34                         </div>
35                     <div class="input-group mb-3">
36                         <input type="password" class="form-control" placeholder="Password" name="password">
37                         <div class="input-group-append">
38                             <div class="input-group-text">
39                                 <span class="fas fa-lock"></span>
40                             </div>
41                         </div>
42                     </div>
43                 </div>
44
45                 @if ($errors->any())
46                     <p class="alert alert-danger">{{ $errors->first() }}</p>
47                 @endif
48
49                 <div class="row">
50                     <div class="col-12">
51                         <button type="submit" class="btn btn-primary btn-block" name="submit">Login</button>
52                     </div>
53                 </div>
54             </form>
55
56             <div class="row mt-3 justify-content-center">
57                 <div class="col-12 text-center">
58                     <span>Belum punya akun? <a href="{{ route('register') }}>Register</a></span>
59                 </div>
60             </div>
61         </div>
62     </div>
63 </div>
64 </x-layouts.guest>
```

untuk lebih detailnya :

<https://drive.google.com/file/d/1TnlnsmOeUr665v7R2MovCWnLHInMpW0b/view?usp=sharing>

Isi dari register.blade.php



```
1
2
3 <x-layouts-guest title="Register">
4     <div class="login-box d-flex flex-column justify-content-center align-items-center w-100 vh-100">
5         {-- login-logo --}
6         <div class="card card-outline card-primary">
7             <div class="Card-header text-center text-lg">
8                 <>Poli</>klinik
9             </div>
10            <div class="card-body">
11                <p class="login-box-msg">Register akun baru!</p>
12                <form action="{{ route('register') }}" method="POST">
13                    @csrf
14
15                    <div class="input-group mb-3">
16                        <input type="text" class="form-control" required placeholder="Nama Lengkap" name="nama">
17                        <div class="input-group-append">
18                            <div class="input-group-text">
19                                <span class="fas fa-user"></span>
20                            </div>
21                        </div>
22                    </div>
23
24                    <div class="input-group mb-3">
25                        <input type="text" class="form-control" required placeholder="Email" name="email">
26                        <div class="input-group-append">
27                            <div class="input-group-text">
28                                <span class="fas fa-envelope"></span>
29                            </div>
30                        </div>
31                    </div>
32
33                    <div class="input-group mb-3">
34                        <input type="text" class="form-control" required placeholder="Alamat" name="alamat">
35                        <div class="input-group-append">
36                            <div class="input-group-text">
37                                <span class="fas fa-map-marker"></span>
38                            </div>
39                        </div>
40                    </div>
41
42                    <div class="input-group mb-3">
43                        <input type="number" class="form-control" required placeholder="No HP" name="no_hp">
44                        <div class="input-group-append">
45                            <div class="input-group-text">
46                                <span class="fas fa-phone-square"></span>
47                            </div>
48                        </div>
49                    </div>
50
51                    <div class="input-group mb-3">
52                        <input type="number" class="form-control" required placeholder="No KTP" name="no_ktp">
53                        <div class="input-group-append">
54                            <div class="input-group-text">
55                                <span class="fas fa-address-book"></span>
56                            </div>
57                        </div>
58                    </div>
59
60                    <div class="input-group mb-3">
61                        <input type="password" class="form-control" required placeholder="Password" name="password">
62                        <div class="input-group-append">
63                            <div class="input-group-text">
64                                <span class="fas fa-lock"></span>
65                            </div>
66                        </div>
67                    </div>
68
69                    <div class="input-group mb-3">
70                        <input type="password" class="form-control" required placeholder="Confirm Password" name="password_confirmation">
71                        <div class="input-group-append">
72                            <div class="input-group-text">
73                                <span class="fas fa-lock"></span>
74                            </div>
75                        </div>
76                    </div>
77
78                    @if ($errors->any())
79                        <div class="alert alert-danger">
80                            <ul class="mb-0">
81                                @foreach ($errors->all() as $error)
82                                    <li>{{ $error }}</li>
83                                @endforeach
84                            </ul>
85                        </div>
86                    @endif
87
88                    <div class="row">
89                        <div class="col-12 text-right">
90                            <button type="submit" class="btn btn-primary btn-block">Register</button>
91                        </div>
92                    </div>
93                </form>
94
95                <div class="row justify-content-center mt-3">
96                    <div class="col-12 text-center">
97                        <p>Sudah punya akun? <a href="{{ route('login') }}>Login</a></p>
98                    </div>
99                </div>
100            </div>
101        </div>
102    </div>
103 </x-layouts-guest>
```

untuk lebih detailnya :

<https://drive.google.com/file/d/198yidwhG6Gmh59bUBdZGP9keOERd5L2w/view?usp=sharing>

