

Menambahkan Sistem Autentikasi



Laravel Fortify

Instalasi Laravel Fortify

Pada framework Laravel 8 sudah terdapat sistem autentikasi seperti Laravel Fortify yang memudahkan developer. Laravel Fortify adalah sebuah paket backend autentikasi tanpa frontend untuk Laravel. Fortify mendaftarkan route dan controller yang diperlukan untuk mengimplementasikan semua fitur autentikasi dari Laravel, termasuk login, registrasi, reset kata sandi, verifikasi email, dan banyak lagi. Namun pada tutorial ini hanya fungsi login, ganti kata sandi, perbarui akun, dan logout yang akan digunakan. Fortify tidak menyediakan view untuk fitur tersebut. Sehingga kita perlu membuat view dari fitur itu sendiri.

Untuk memulai, silahkan install fortify menggunakan perintah composer di bawah ini. Perintah ini pada dasarnya akan menambahkan laravel/fortify ke dalam composer.json dan mengunduh dependency ke dalam folder vendor.

```
composer require laravel/fortify
```

Kemudian publish konfigurasi fortify menggunakan perintah Artisan di bawah ini. Perintah ini akan menambahkan file konfigurasi, controller, dan migration. Namun fortify hanya mewajibkan kita untuk mengimplementasikan view.

```
php artisan vendor:publish --provider="Laravel\Fortify\FortifyServiceProvider"
```

Selanjutnya lakukan migrasi basis data karena file migrasi baru telah ditambahkan. Gunakan perintah Artisan di bawah ini pada command prompt.

```
php artisan migrate
```

Buka file config\fortify.php dan edit features dengan melakukan comment untuk mengunci beberapa fitur yang tidak diperlukan.

Nama file	config\fortify.php
Deskripsi	Konfigurasi fortify
<pre>... 'features' => [// Features::registration(), // Features::resetPasswords(), // Features::emailVerification(), Features::updateProfileInformation(), Features::updatePasswords(),</pre>	

```

        // Features::twoFactorAuthentication([
        //     'confirmPassword' => true,
        // ]),
    ],
    ...

```

Buka file config/app.php dan sisipkan kelas FortifyServiceProvider ke dalam daftar application service providers.

Nama file	config/app.php
Deskripsi	Konfigurasi service provider
<pre> ... App\Providers\FortifyServiceProvider::class, ... </pre>	
<pre> /* * Application Service Providers... */ App\Providers\AppServiceProvider::class, App\Providers\AuthServiceProvider::class, // App\Providers\BroadcastServiceProvider::class, App\Providers\EventServiceProvider::class, App\Providers\FortifyServiceProvider::class, App\Providers\RouteServiceProvider::class, </pre>	

Buka file app/Providers/RouteServiceProvider.php dan hapus kode sebelah kiri. Kemudian ganti dengan kode yang ada pada sebelah kanan.

Nama file	app/Providers/RouteServiceProvider.php
Deskripsi	Service provider untuk router
<pre> ... public const HOME = '/home'; ... </pre>	<pre> ... public const HOME = '/dasbor'; ... </pre>

Untuk menggunakan router pertama kalinya, buka RouteServiceProvider dan uncomment kode `protected $namespace = 'App\Http\Controllers';` yang berada di baris ke-29.

Nama file	app/Providers/RouteServiceProvider.php
Deskripsi	Service provider untuk router
<pre> ... protected \$namespace = 'App\Http\Controllers'; ... </pre>	

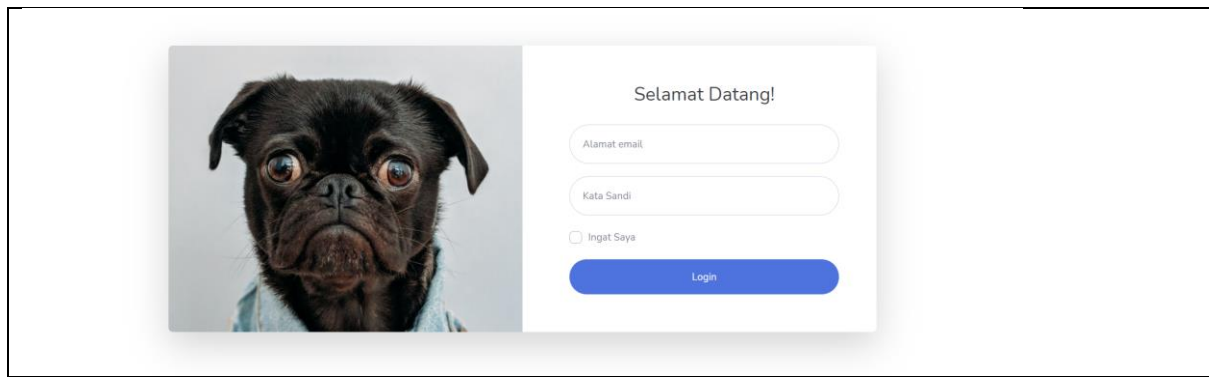
Menambahkan Fitur Login

Buka file app/Providers/FortifyServiceProvider.php dan tambahkan satu baris kode `Fortify::loginView('auth.login');` ke dalam method boot().

Nama file	app/Providers/FortifyServiceProvider.php
Deskripsi	Service provider untuk Laravel Fortify
<pre> ... public function boot() { ... Fortify::loginView('auth.login'); } ... </pre>	

Kemudian untuk mengimplementasikan fitur login, silahkan salin file `resources\views\simple-template.blade.php` ke file `resources\views\auth\login.blade.php` dan ketik kode di bawah ini.

Nama file	resources\views\auth\login.blade.php
Deskripsi	View untuk fitur login
	<pre> @extends('layouts.app') @section('title', 'Masuk') @section('simple') <div class="container"> <div class="row justify-content-center"> <div class="col-xl-10 col-lg-12 col-md-9"> <div class="card o-hidden border-0 shadow-lg my-5"> <div class="card-body p-0"> <div class="row"> <div class="col-lg-6 d-none d-lg-block bg-login-image"></div> <div class="col-lg-6"> <div class="p-5"> <div class="text-center"> <h1 class="h4 text-gray-900 mb-4">Selamat Datang!</h1> </div> <form action="{{ route('login') }}" method="POST" class="user"> @csrf <div class="form-group"> <input type="email" name="email" class="form-control form-control-user @error('email') is-invalid @enderror" placeholder="Alamat email"> @error('email') <div class="invalid-feedback">{{ \$message }}</div> @enderror </div> <div class="form-group"> <input type="password" name="password" class="form-control form-control-user @error('password') is-invalid @enderror" placeholder="Kata Sandi"> @error('password') <div class="invalid-feedback">{{ \$message }}</div> @enderror </div> <div class="form-group"> <div class="custom-control custom-checkbox small"> <input type="checkbox" class="custom-control-input" id="customCheck" name="remember" @if(old('remember') == 'on') checked @endif> <label class="custom-control-label" for="customCheck">Ingat Saya</label> </div> </div> <button type="submit" class="btn btn-primary btn-user btn-block">Login</button> </form> </div> </div> </div> </div> </div> </div> </div> </div> </section> </pre>



Pada tahap ini fitur login telah diimplementasikan. Semua proses backend autentikasi telah ditangani oleh Laravel Fortify.

Menambahkan Halaman Dasbor

Buka file `routes/web.php` dan hapus kode sebelah kiri. Kemudian ganti dengan kode yang ada pada sebelah kanan. Setelah itu, hapus file `resources/views/welcome.blade.php`.

Nama file	<code>routes/web.php</code>	
Deskripsi	Routes laravel	
	<pre>Route::get('/', function () { return view('welcome'); });</pre>	<pre>Route::redirect('/', '/dasbor');</pre>

Kita telah mengubah halaman root ke halaman dasbor. Nantinya apabila pengguna belum memiliki sesi di dalam sistem, maka sistem akan mengalihkan ke halaman login. Untuk menambahkan dasbor, Silahkan salin file `resources/views/template.blade.php` ke file `resources/views/dasbor.blade.php` dan ketik kode di bawah ini.

Nama file	<code>resources/views/dasbor.blade.php</code>	
Deskripsi	View untuk dasbor	
	<pre>@extends('layouts.app') @section('title', 'Dasbor') @section('content') <p>Selamat Datang!</p> @endsection</pre>	

Setelah view dibuat, buatlah DashboardController dengan menjalankan perintah Artisan make:controller pada command prompt.

```
php artisan make:controller DashboardController
```

File controller disimpan ke dalam file app\Http\Controllers\DashboardController.php. Buka file tersebut, terdapat kelas DashboardController. Kemudian tambahkan method index ke dalam kelas tersebut.

Nama file	app\Http\Controllers\DashboardController.php
Deskripsi	Controller untuk dasbor
<pre>... public function index(Request \$request) { return view('dasbor'); } ...</pre>	

Buka routes/web.app dan tambahkan route berikut.

Nama file	routes/web.app
Deskripsi	Routes laravel
<pre>... /* Routing untuk user terautentikasi */ Route::middleware('auth')->group(function() { Route::get('/dasbor', 'DashboardController@index')->name('dasbor'); /* Tambahkan route lain disini... */ }); ...</pre>	

Method group digunakan untuk menggabungkan beberapa route. Route::middleware digunakan untuk mendefinisikan middleware yang diaplikasikan pada route. Gabungan keduanya mendefinisikan bahwa route yang berada di dalam fungsi group harus melewati middleware auth dimana syarat ini akan terpenuhi apabila pengguna sudah login. Method name digunakan untuk mendefinisikan kode nama dari route itu sendiri. Pada saat ini di controller DashboardController hanya terdapat view dasbor dan manajemen data akan ditambahkan nantinya. **Catatan:** Setiap route yang ditambahkan selanjutnya harus diletakkan di dalam fungsi group.

Ubah template resources\views\layouts\sidebar.blade.php dan tambahkan href pada sidebar dasbor. Apabila route saat ini adalah dasbor maka kelas css active akan ditambahkan untuk menonjolkan pada sidebar. Fungsi route digunakan untuk generate url saat ini berdasarkan kode nama dari route.

Nama file	resources\views\layouts\sidebar.blade.php
Deskripsi	Sidebar
<pre>... <li class="nav-item @if(Route::is('dasbor')) active @endif"> <i class="fas fa-fw fa-tachometer-alt"></i></pre>	

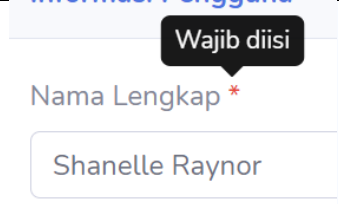
```
<span>Dasbor</span>
</a>
</li>
...
```

Menambahkan Komponen Kustom pada Blade

Tutorial ini merupakan fitur favorit dari template engine blade. Kita bisa membuat sebuah komponen yang dapat digunakan berulang-ulang. Jalankan perintah Artisan ini untuk membuat komponen Required. Komponen ini akan muncul sebagai simbol bintang berwarna merah (*) yang menandakan bahwa sebuah form wajib diisi.

```
php artisan make:component Required
```

Buka `resources\views\components\required.blade.php` dan ganti konten berkas tersebut dengan kode yang ada di bawah ini.

Nama file	resources\views\components\required.blade.php
Deskripsi	View untuk component Required
<pre>*</pre>	
	

Aktifkan tooltip yang disediakan bootstrap dengan menambahkan kode javascript di bawah ini ke dalam file `layouts\app.blade.php`. Letakkan kode script ini dibawah script sb-admin.

Nama file	resources\views\layouts\app.blade.php
Deskripsi	Layout scaffold
<pre>... <script src="{{ asset('js/sb-admin-2.min.js') }}"></script> <script> \$('[data-toggle="tooltip"]').tooltip(); </script> ...</pre>	

Menambahkan Profil, Edit Profil, dan Edit Kata Sandi.

Buat ProfileController dengan perintah Artisan. Kemudian buka ProfileController.php dan tambahkan method index.

```
php artisan make:controller ProfileController
```

Nama file	app\Http\Controllers/ProfileController.php
Deskripsi	Controller untuk halaman profil

```

...
public function index(Request $request) {
    return view('profil', [
        'user' => $request->user()
    ]);
}
...

```

Kemudian definisikan route dari profil dengan mengedit routes/web.php. Pastikan kamu meletakkan route di dalam fungsi group yang sudah dibuat sebelumnya.

Nama file	routes/web.php
Deskripsi	Routes laravel
<pre> ... /* Routing untuk user terautentikasi */ Route::middleware('auth')->group(function() { Route::get('/dasbor', 'DashboardController@index')->name('dasbor'); Route::get('/profil', 'ProfileController@index')->name('profil'); }); ... </pre>	

Buka topbar.blade.php pada folder layouts. Kemudian cari dropdown item untuk profil dan ganti kode tersebut menjadi seperti di bawah ini dengan menambahkan href pada tag anchor.

Nama file	resources/views/layouts/topbar.blade.php
Deskripsi	Topbar
<pre> ... <i class="fas fa-user fa-sm fa-fw mr-2 text-gray-400"></i> Profil ... </pre>	

Duplikat file template.blade.php ke profil.blade.php. Kemudian salin kode ini ke dalam file profil.blade.php.

Nama file	resources/views/profil.blade.php
Deskripsi	View untuk halaman profil
<pre> @extends('layouts.app') @section('title', 'Profil Saya') @section('content') <div class="card shadow mb-4"> <div class="card-header py-3 d-flex flex-row align-items-center justify-content-between"> <h6 class="m-0 font-weight-bold text-primary">Informasi Pengguna</h6> </div> <div class="card-body"> <form action="{{ route('user-profile-information.update') }}" method="post"> @method('put') @csrf </pre>	

```

        <div class="row">
            <div class="col-lg-6">
                <div class="form-group">
                    <label class="form-control-label" for="input-name">Nama
Lengkap <x-required/></label>
                    <input type="text" id="input-name" name="name" class="form-
control @error('name') is-invalid @enderror" placeholder="Nama Lengkap"
value="{{ old('name', $user->name) }}" required>
                    @error('name')
                <div class="invalid-feedback">{{ $message }}</div>
                @enderror
            </div>
        </div>
        <div class="col-lg-6">
            <div class="form-group">
                <label class="form-control-label" for="input-email">Email
address <x-required/></label>
                <div class="input-group input-group-merge">
                    <input type="email" id="input-email" name="email"
class="form-control @error('email') is-invalid @enderror"
placeholder="Alamat Email" value="{{ old('email', $user->email) }}"
required>
                    <div class="input-group-append bg-white">
                        <div class="input-group-text">
                            @if($user->hasVerifiedEmail())
                                <i class="fas fa-check-circle fa-fw text-primary"
data-toggle="tooltip" title="Terverifikasi"></i>
                            @else
                                <i class="fas fa-times-circle fa-fw text-danger"
data-toggle="tooltip" title="Tidak Terverifikasi"></i>
                            @endif
                        </div>
                    </div>
                </div>
                @error('email')
                <div class="invalid-feedback d-block">{{ $message }}</div>
                @enderror
            </div>
        </div>
        <div>
            <button type="submit" class="btn btn-primary"><i class="fa fa-
save fa-fw"></i> Simpan</button>
        </form>
    </div>
</div>

<div class="card shadow mb-4">
    <div class="card-header py-3 d-flex flex-row align-items-center
justify-content-between">
        <h6 class="m-0 font-weight-bold text-primary">Ganti Kata Sandi</h6>
    </div>
    <div class="card-body">
        <form action="{{ route('user-password.update') }}" method="post">
            @method('put')
            @csrf

            @if($user->password)
                <div class="form-group">
                    <label class="form-control-label" for="input-current-
password">Kata Sandi Saat Ini <x-required/></label>
                    <input type="password" id="input-current-password"

```



```

name="current_password" class="form-control @error('current_password')
is-invalid @enderror" placeholder="Masukkan kata sandi saat ini">
    @error('current_password')
    <div class="invalid-feedback">{{ $message }}</div>
    @enderror
</div>
@endif
<div class="form-group">
    <label class="form-control-label" for="input-password">Kata
Sandi Baru <x-required/></label>
    <input type="password" id="input-password" name="password"
class="form-control @error('password') is-invalid @enderror"
placeholder="Masukkan kata sandi baru">
    @error('password')
    <div class="invalid-feedback">{{ $message }}</div>
    @enderror
</div>
<div class="form-group">
    <label class="form-control-label" for="input-password-
confirmation">Konfirmasi Kata Sandi Baru <x-required/></label>
    <input type="password" id="input-password-confirmation"
name="password_confirmation" class="form-control
@error('password_confirmation') is-invalid @enderror" placeholder="Ketik
ulang kata sandi baru">
    @error('password_confirmation')
    <div class="invalid-feedback">{{ $message }}</div>
    @enderror
</div>
<button type="submit" class="btn btn-primary"><i class="fa fa-
save fa-fw"></i> Perbarui</button>
</form>
</div>
</div>
@endsection

```

Sama seperti login, backend untuk fitur update profil dan kata sandi ditangani oleh Laravel Fortify.

Menambahkan Logout

Untuk menambahkan tombol logout cukup mudah. Buka file `resources\views\logout-modal.blade.php` dan hapus kode sebelah kiri. Kemudian ganti dengan kode yang ada pada sebelah kanan.

Nama file	<code>resources\views\logout-modal.blade.php</code>	
Deskripsi	Logout modal	
<pre>... Keluar ...</pre>		<pre>... <form action="{{ route('logout') }}" method="POST"> @csrf <button type="submit" class="btn btn-primary">Keluar</button> </form> ...</pre>
		

Proses dari logout mulai dari penghancuran session dan cookie akan ditangani oleh Laravel Fortify. Jangan lupa untuk mencoba menjalankan proyek laravel dengan menggunakan perintah `Artisan serve`. Kemudian buka `http://localhost:8000`.

```
php artisan serve
```

Gunakan kredensial di bawah ini ketika mengakses halaman login. Kredensial ini dibuat otomatis ketika migration dan seeder dilakukan.

Email : admin@gmail.com
Password : password