

Praktikum – Auth dan Middleware

1. Sebelum melangkah lebih lanjut setting file pada config/auth.php ubah pada bagian model menjadi model yang telah kita buat sebelumnya untuk menyimpan username dan password

```
62     'providers' => [  
63         'users' => [  
64             'driver' => 'eloquent',  
65             'model' => App\Models\UserModel::class,  
66         ],  
67     ],
```

2. Membuat middleware dengan perintah pada cmd atau terminal

```
PS D:\COOLYEAH\Semester 4\PWL Praktikum\PWL_POS> php artisan make:middleware Cek_login  
  
INFO Middleware [D:\COOLYEAH\Semester 4\PWL Praktikum\PWL_POS\app\Http\Middleware\Cek_login.php] created successfully.
```

3. Kemudian tambahkan kode dibawah ini, letakkan di function handle()

```
public function handle(Request $request, Closure $next, $roles): Response  
{  
    if (!Auth::check()) {  
        return redirect('login');  
    }  
    $user = Auth::user();  
  
    if ($user->level_id == $roles) {  
        return $next($request);  
    }  
    return redirect('login')->with('error', 'Maaf anda tidak memiliki akses');  
}
```

- Kode ini merupakan middleware untuk mengecek hak akses pengguna sebelum melanjutkan ke halaman yang dituju. Pertama, fungsi ini mengecek apakah pengguna sudah login. Jika belum, pengguna akan diarahkan ke halaman login. Jika sudah login, fungsi ini akan mengecek apakah level pengguna sesuai dengan level yang dibutuhkan untuk mengakses halaman tersebut (parameter \$roles). Jika level cocok, fungsi akan melanjutkan ke halaman yang dituju. Sebaliknya, pengguna akan

diarahkan kembali ke halaman login dengan pesan bahwa mereka tidak memiliki akses.

4. Ketika sudah membuat middleware, langkah berikutnya harus registrasikan pada kernel.php. Agar middleware dapat dibaca oleh sistem. Buka folder App/Http/Kernel.php dan tambahkan code pada variabel \$middlewareAliases seperti dibawah

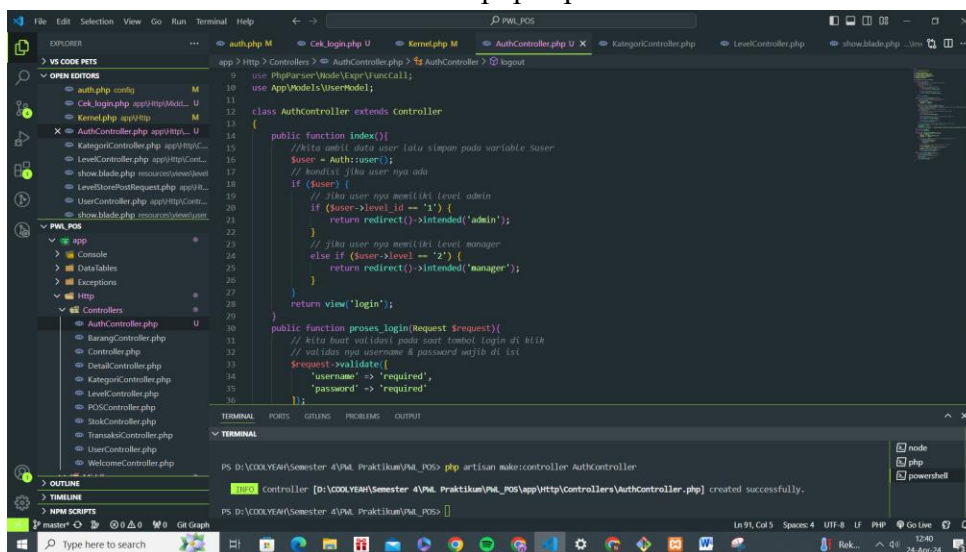
```
//cek login
'cek_login' => \App\Http\Middleware\Cek_login::class,
```

5. Membuat Controller Auth. Pada langkah ini kita akan membuat controller auth, yang dimana di dalamnya terdapat proses autentikasi dari request login. Jalankan perintah make:controller

```
PS D:\COOLYEAH\Semester 4\PWL Praktikum\PWL_POS> php artisan make:controller AuthController
```

```
INFO Controller [D:\COOLYEAH\Semester 4\PWL Praktikum\PWL_POS\app\Http\Controllers\AuthController.php] created successfully.
```

6. Kemudian buat code AuthController.php seperti dibawah ini



- Secara keseluruhan AuthController digunakan untuk menangani fungsi login dan register

7. Kemudian tambahkan code pada route pada file routes/web.php seperti kode program dibawah

```
Route::group(['middleware'=>['auth']], function(){
    Route::group(['middleware'=>['cek_login:1']], function(){
        Route::resource('admin', [AdminController::class]);
    });
    Route::group(['middleware'=>['cek_login:2']], function(){
        Route::resource('manager', [ManagerController::class]);
    });
});
```

8. Kemudian buat file controller dengan nama AdminController sebagai halaman yang boleh diakses oleh admin saja. Jalankan perintah dibawah

```
PS D:\COOLYEAH\Semester 4\PWL Praktikum\PWL_POS> php artisan make:controller AdminController
```

```
INFO Controller [D:\COOLYEAH\Semester 4\PWL Praktikum\PWL_POS\app\Http\Controllers\AdminController.php] created successfully.
```

9. Buka folder App/Http/Controllers dan isi file AdminController seperti dibawah

```
class AdminController extends Controller
{
    public function index(){
        return view('admin');
    }
}
```

- Mengarahkan ke halaman admin

10. . Kemudian buat file controller dengan nama ManagerController sebagai halaman yang boleh diakses oleh admin saja. Jalankan perintah dibawah

```
PS D:\COOLYEAH\Semester 4\PWL Praktikum\PWL_POS> php artisan make:controller ManagerController
```

```
INFO Controller [D:\COOLYEAH\Semester 4\PWL Praktikum\PWL_POS\app\Http\Controllers\ManagerController.php] created successfully.
```

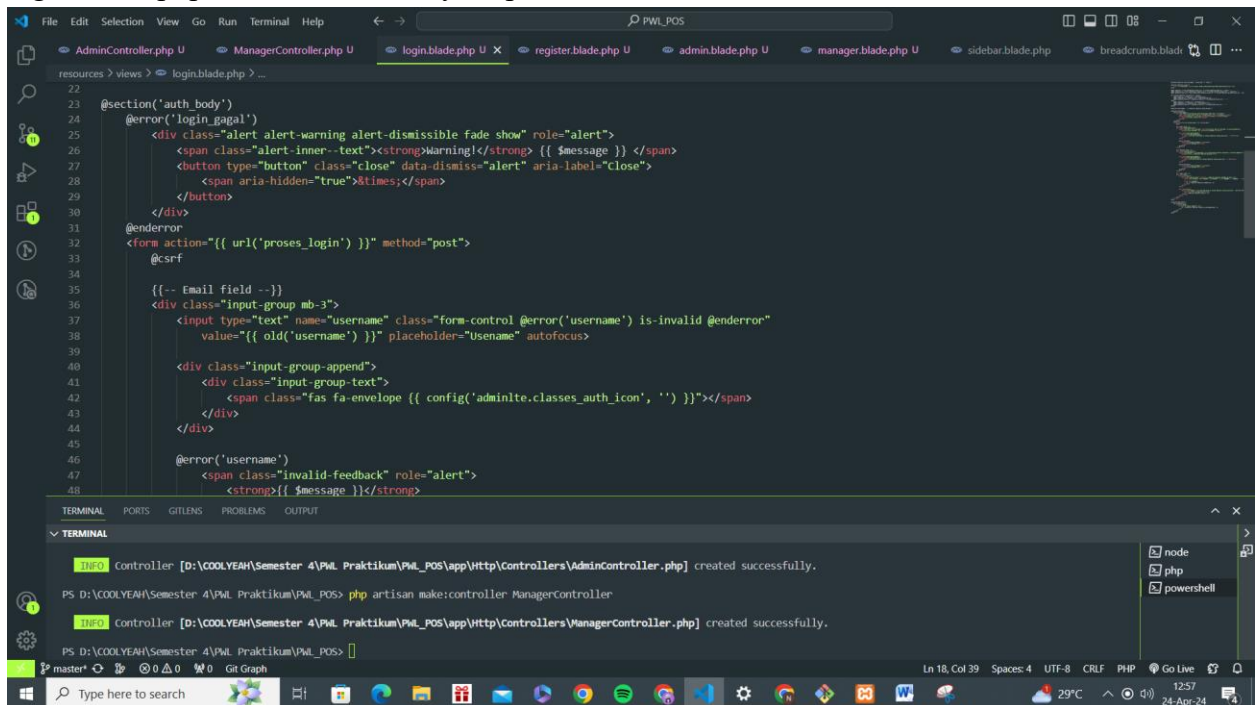
11. Buka folder App/Http/Controllers dan isi file ManagerController seperti dibawah

```
class ManagerController extends Controller
{
    public function index(){
        return view('manager');
    }
}
```

- Mengarahkan ke halaman manager

12. Pada langkah terakhir yaitu membuat layout sederhana

13. Buatlah halaman login dengan membuka folder resources/views dengan nama file login.blade.php. Lalu kita isi filenya seperti contoh dibawah



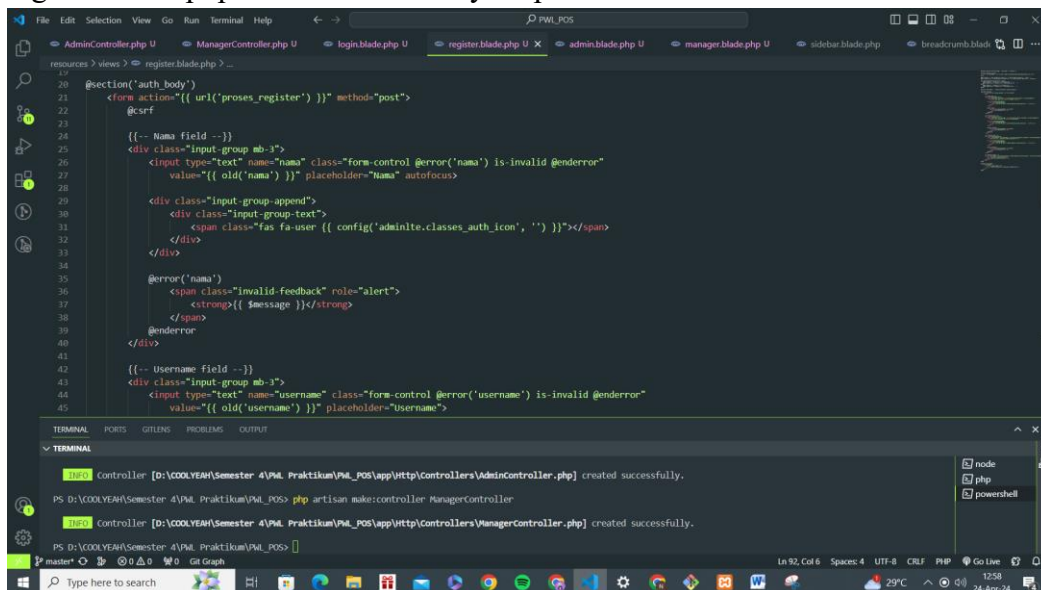
The screenshot shows a Visual Studio Code editor with the file `login.blade.php` open. The file contains Blade template code for a login form. The code includes an error handling section for login failure, a form action pointing to `proses_login`, and input fields for email and username with associated error messages. The terminal at the bottom shows the successful creation of the `AdminController.php` and `ManagerController.php` files.

```
resources > views > login.blade.php > ...
22
23 @section('auth_body')
24 @error('login_gagal')
25 <div class="alert alert-warning alert-dismissible fade show" role="alert">
26 <span class="alert-inner--text"><strong>Warning!</strong> {{ $message }} </span>
27 <button type="button" class="close" data-dismiss="alert" aria-label="Close">
28 <span aria-hidden="true">&times;</span>
29 </button>
30 </div>
31 @enderror
32 <form action="{{ url('proses_login') }}" method="post">
33 @csrf
34
35 {{-- Email field --}}
36 <div class="input-group mb-3">
37 <input type="text" name="username" class="form-control @error('username') is-invalid @enderror"
38 value="{{ old('username') }}" placeholder="Username" autofocus>
39
40 <div class="input-group-append">
41 <div class="input-group-text">
42 <span class="fas fa-envelope {{ config('adminlte.classes_auth_icon', '') }}"></span>
43 </div>
44 </div>
45
46 @error('username')
47 <span class="invalid-feedback" role="alert">
48 <strong>{{ $message }}</strong>
49 </span>
50 @enderror
51 </div>
52
53 {{-- Password field --}}
54 <div class="input-group mb-3">
55 <input type="password" name="password" class="form-control @error('password') is-invalid @enderror"
56 value="{{ old('password') }}" placeholder="Password" autofocus>
57
58 <div class="input-group-append">
59 <div class="input-group-text">
60 <span class="fas fa-lock {{ config('adminlte.classes_auth_icon', '') }}"></span>
61 </div>
62 </div>
63
64 @error('password')
65 <span class="invalid-feedback" role="alert">
66 <strong>{{ $message }}</strong>
67 </span>
68 @enderror
69 </div>
70
71 <div class="text-center">
72 <button type="submit" class="btn btn-primary">Login</button>
73 </div>
74 </form>
75
76 @endsection
```

TERMINAL

```
INFO controller [D:\COOLYEAH\Semester 4\PMI Praktikum\PMI_POS\app\Http\Controllers\AdminController.php] created successfully.
PS D:\COOLYEAH\Semester 4\PMI Praktikum\PMI_POS> php artisan make:controller ManagerController
INFO controller [D:\COOLYEAH\Semester 4\PMI Praktikum\PMI_POS\app\Http\Controllers\ManagerController.php] created successfully.
PS D:\COOLYEAH\Semester 4\PMI Praktikum\PMI_POS>
```

14. Buatlah halaman register dengan membuka folder resources/views dengan nama file register.blade.php. Lalu kita isi filenya seperti contoh dibawah



The screenshot shows a Visual Studio Code editor with the file `register.blade.php` open. The file contains Blade template code for a registration form. The code includes a form action pointing to `proses_register`, input fields for name and username with associated error messages, and a password field. The terminal at the bottom shows the successful creation of the `AdminController.php` and `ManagerController.php` files.

```
resources > views > register.blade.php > ...
20 @section('auth_body')
21 <form action="{{ url('proses_register') }}" method="post">
22 @csrf
23
24 {{-- Nama field --}}
25 <div class="input-group mb-3">
26 <input type="text" name="nama" class="form-control @error('nama') is-invalid @enderror"
27 value="{{ old('nama') }}" placeholder="Nama" autofocus>
28
29 <div class="input-group-append">
30 <div class="input-group-text">
31 <span class="fas fa-user {{ config('adminlte.classes_auth_icon', '') }}"></span>
32 </div>
33 </div>
34
35 @error('nama')
36 <span class="invalid-feedback" role="alert">
37 <strong>{{ $message }}</strong>
38 </span>
39 @enderror
40 </div>
41
42 {{-- Username field --}}
43 <div class="input-group mb-3">
44 <input type="text" name="username" class="form-control @error('username') is-invalid @enderror"
45 value="{{ old('username') }}" placeholder="Username">
46
47 <div class="input-group-append">
48 <div class="input-group-text">
49 <span class="fas fa-user {{ config('adminlte.classes_auth_icon', '') }}"></span>
50 </div>
51 </div>
52
53 @error('username')
54 <span class="invalid-feedback" role="alert">
55 <strong>{{ $message }}</strong>
56 </span>
57 @enderror
58 </div>
59
60 {{-- Password field --}}
61 <div class="input-group mb-3">
62 <input type="password" name="password" class="form-control @error('password') is-invalid @enderror"
63 value="{{ old('password') }}" placeholder="Password" autofocus>
64
65 <div class="input-group-append">
66 <div class="input-group-text">
67 <span class="fas fa-lock {{ config('adminlte.classes_auth_icon', '') }}"></span>
68 </div>
69 </div>
70
71 @error('password')
72 <span class="invalid-feedback" role="alert">
73 <strong>{{ $message }}</strong>
74 </span>
75 @enderror
76 </div>
77
78 <div class="text-center">
79 <button type="submit" class="btn btn-primary">Register</button>
80 </div>
81 </form>
82
83 @endsection
```

TERMINAL

```
INFO controller [D:\COOLYEAH\Semester 4\PMI Praktikum\PMI_POS\app\Http\Controllers\AdminController.php] created successfully.
PS D:\COOLYEAH\Semester 4\PMI Praktikum\PMI_POS> php artisan make:controller ManagerController
INFO controller [D:\COOLYEAH\Semester 4\PMI Praktikum\PMI_POS\app\Http\Controllers\ManagerController.php] created successfully.
PS D:\COOLYEAH\Semester 4\PMI Praktikum\PMI_POS>
```

15. Buatlah halaman register dengan membuka folder resources/views dengan nama file admin.blade.php. Lalu kita isi filenya seperti contoh dibawah

```
1 @extends('layouts.app')
2
3 {{-- Customize layout sections --}}
4
5 @section('subtitle', 'Admin')
6 @section('content_header_title', 'Home')
7 @section('content_header_subtitle', 'Admin')
8
9 @section('content')
10 <div class="container">
11     <div class="card">
12         <div class="card-header">Tampilan @if (Auth::user()->level_id == 1) Admin @else Manager @endif</div>
13         <div class="card-body">
14             <h1>Login Sebagai: @if (Auth::user()->level_id == 1) Admin @else Manager @endif</h1>
15             <a href="{{ route('logout') }}">Logout</a>
16         </div>
17     </div>
18 </div>
19 @endsection
20
21 @push('js')
22 @endpush
23
```

16. Buatlah halaman register dengan membuka folder resources/views dengan nama file manager.blade.php. Lalu kita isi filenya seperti contoh dibawah

```
@extends('layouts.app')

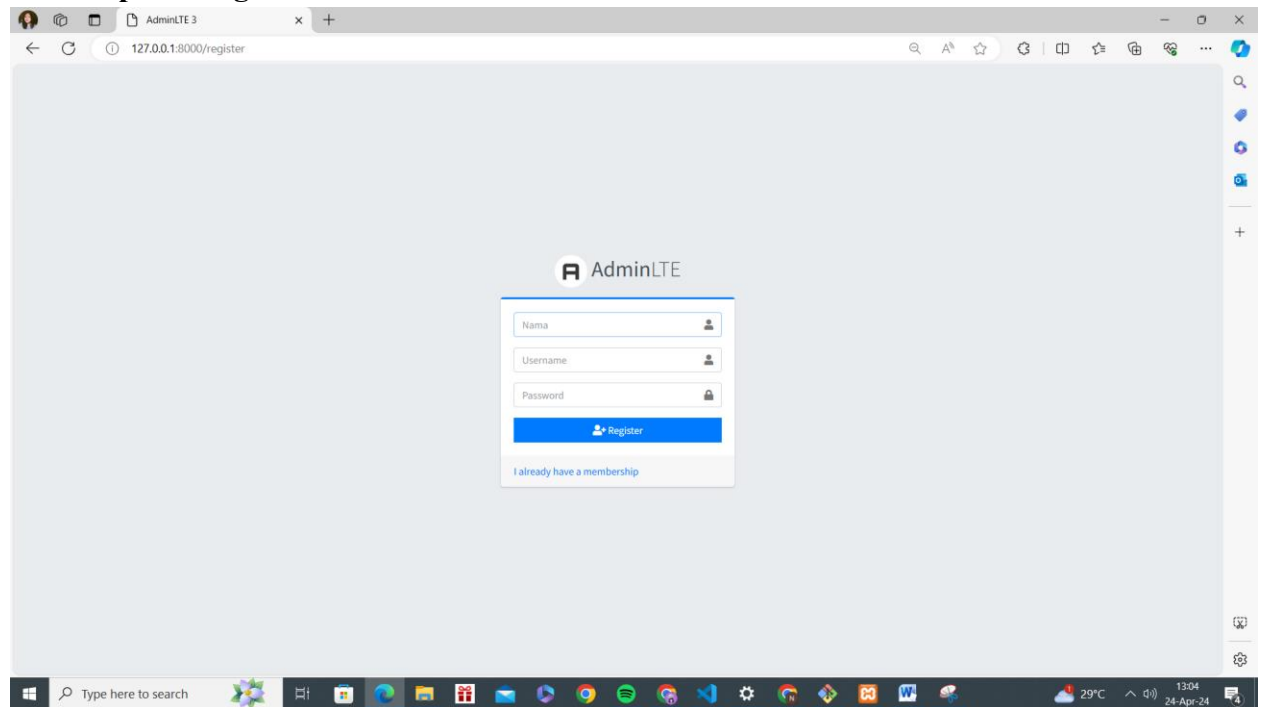
{{-- Customize layout sections --}}

@section('subtitle', 'Manager')
@section('content_header_title', 'Home')
@section('content_header_subtitle', 'Manager')

@section('content')
<div class="container">
    <div class="card">
        <div class="card-header">Tampilan @if (Auth::user()->level_id == 1) Admin @else Manager @endif</div>
        <div class="card-body">
            <h1>Login Sebagai: @if (Auth::user()->level_id == 1) Admin @else Manager @endif</h1>
            <a href="{{ route('logout') }}">Logout</a>
        </div>
    </div>
</div>
@endsection

@push('js')
@endpush
```

- Tampilan Register

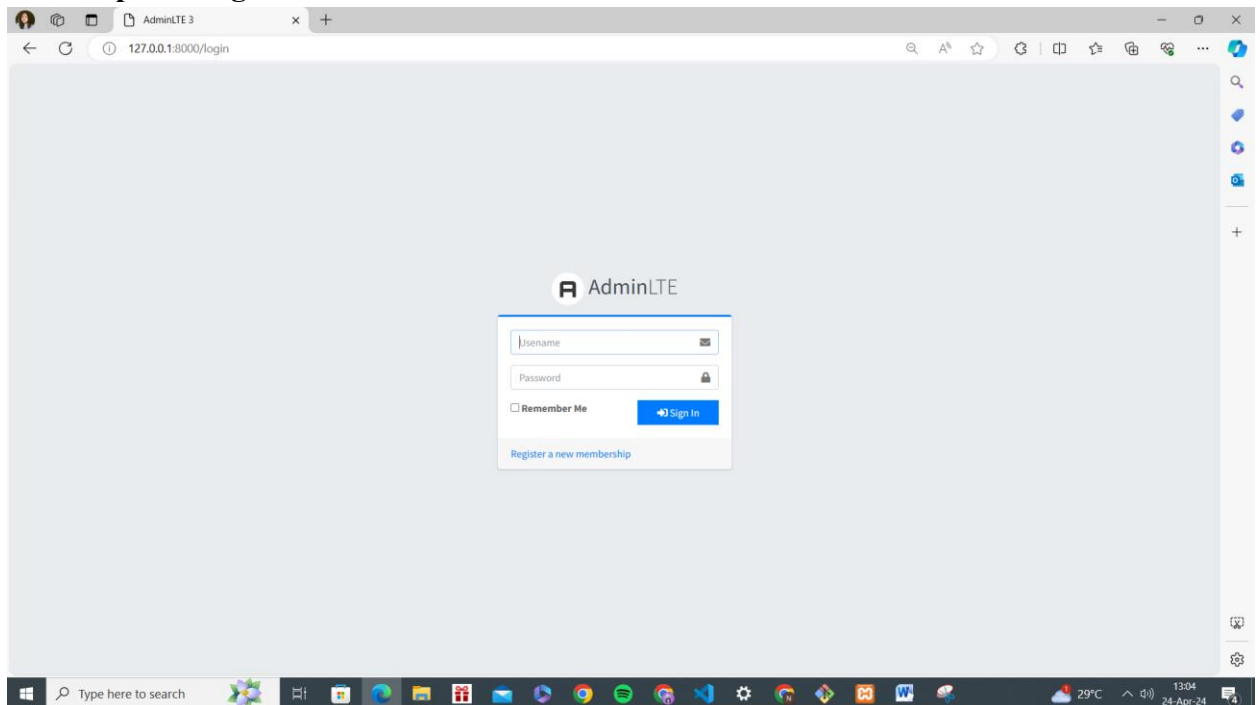


Hasil Register

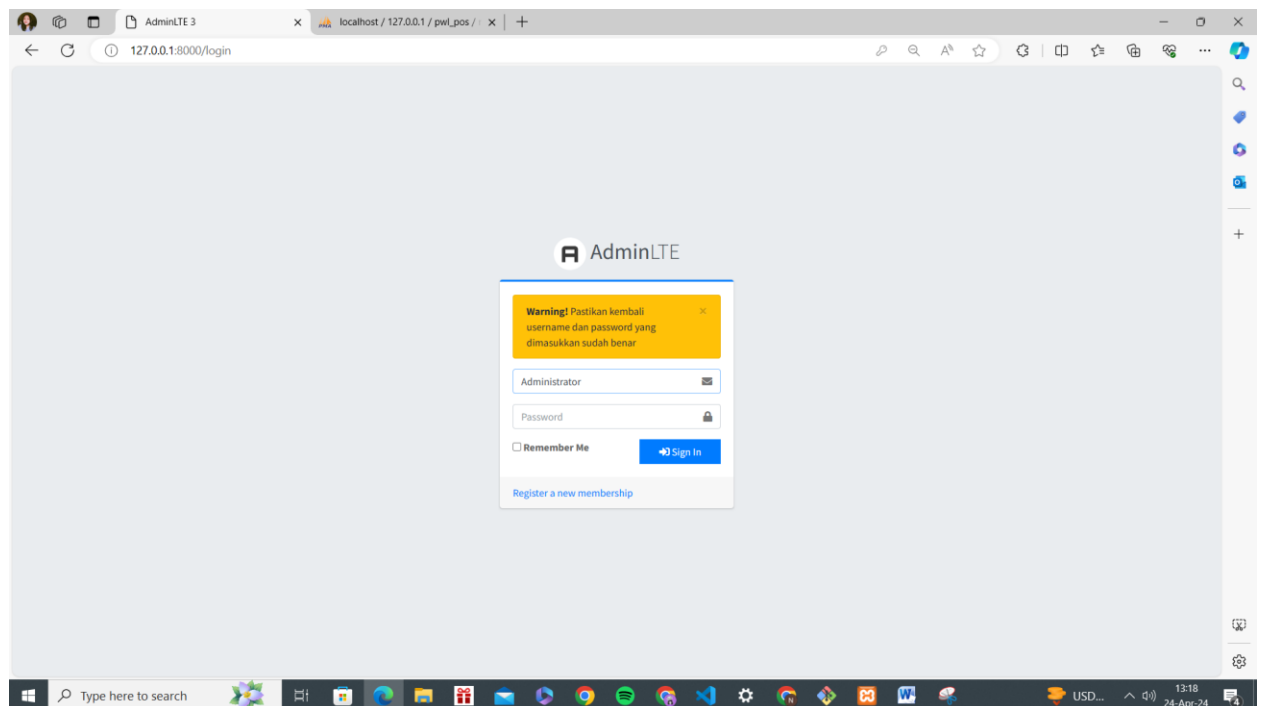
<input type="checkbox"/>	Edit Copy Delete	15	2 nadila	Nadila Amalia	\$2y\$12\$5IG7J6MGhQT4GE2pkLgPnO0bYtOR9Xklldlxt73kl...	2024-04-24 06:06:41	2024-04-24 06:06:41
--------------------------	--	----	----------	---------------	--	------------------------	------------------------

- Pada fitur register akan diminta untuk menginputkan nama, kemudian user dan password yang nantinya akan digunakan untuk login

- Tampilan Login



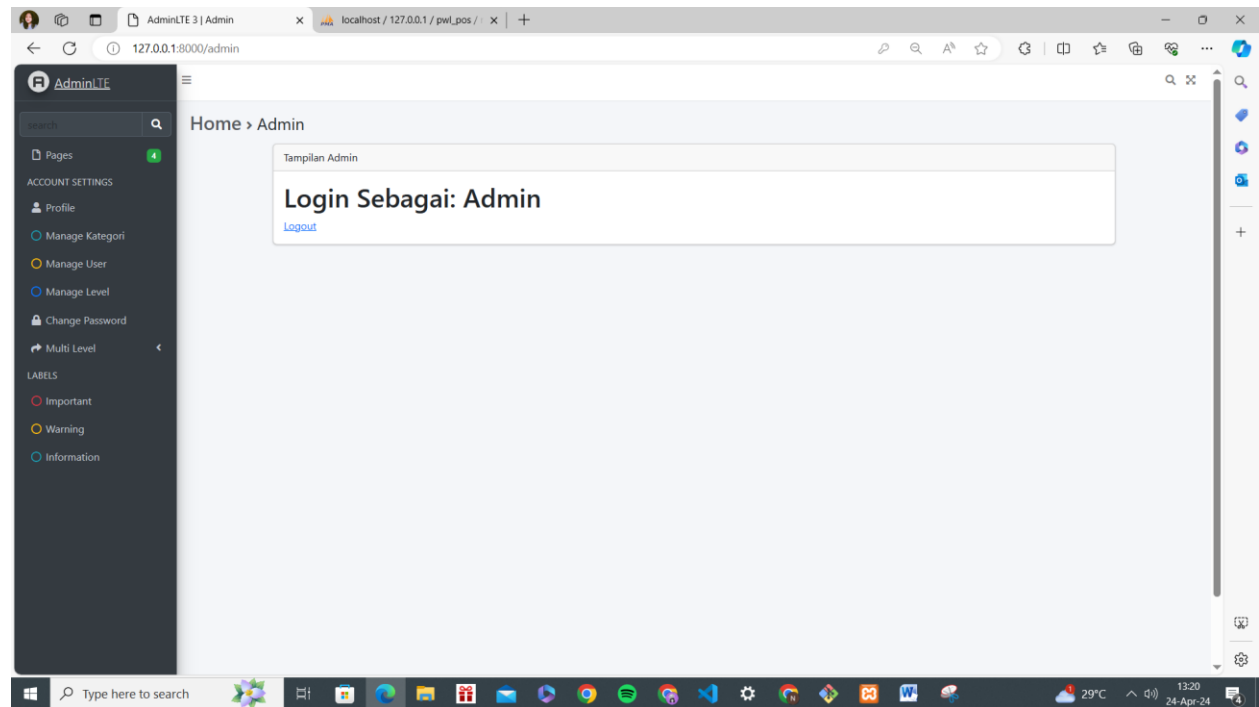
- Pada proses login akan diminta untuk memasukkan username dan password
Password / Username salah



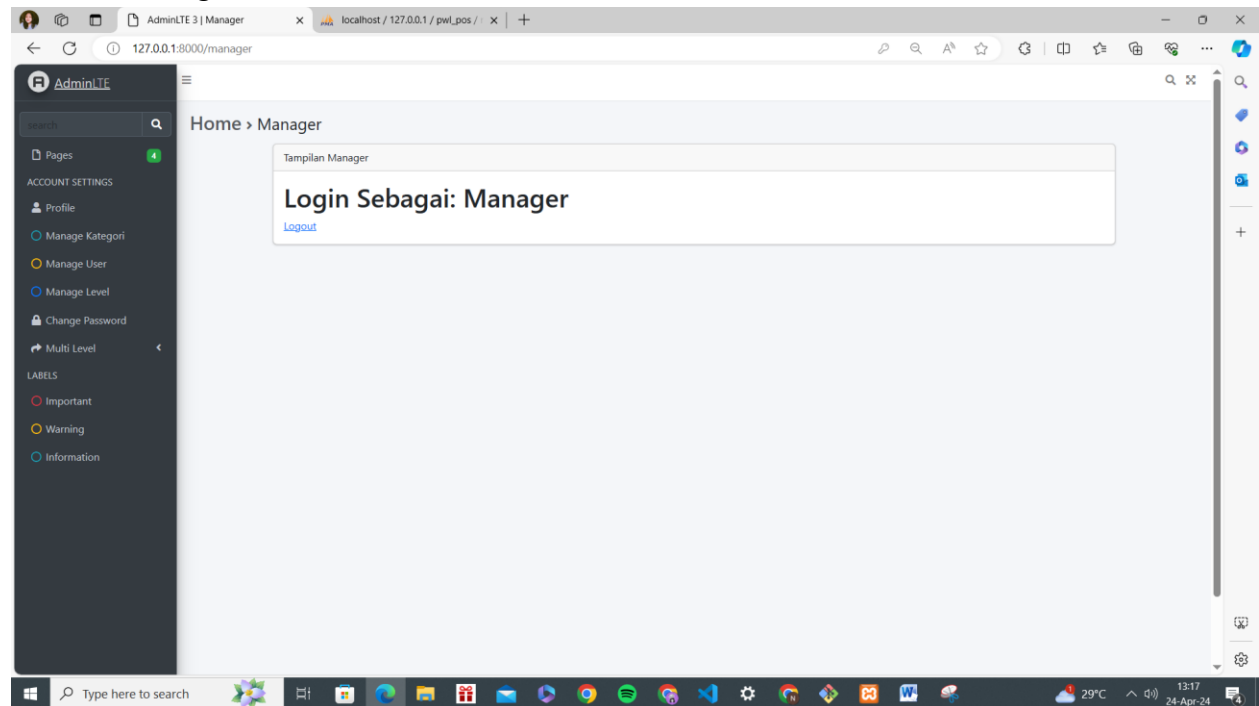
- Apabila username atau password salah, maka akan muncul warning dan tetap berada pada halaman login

Berhasil Login

- Apabila berhasil login, maka akan dicek roles nya apakah merupakan manager atau admin. Roles dengan ID = 1 adalah admin, maka saat login berhasil akan diarahkna ke halaman admin



- Roles dengan ID = 2 adalah manager, maka saat login berhasil akan diarahkan ke halaman manager



- Apabila user logout, maka akan diarahkan ke halaman login kembali