

LAPORAN PRAKTIKUM JOBSHEET 7

**MATA KULIAH PEMROGRAMAN WEB LANJUT
AUTENTICAATION DAN *AUTHORIZATION* DI LARAVEL**

Dosen Pengampu : Dimas Wahyu Wibowo, S.T., M.T.



**Di Susun Oleh :
Louise Nazarossa (2341760117)
SIB-2A/18**

**PROGRAM STUDI D4 SISTEM INFORMASI BISNIS
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG
2025**

PRAKTIKUM 1: Implementasi Authentication

- a.) Kita buka project laravel **PWL_POS** kita, dan kita modifikasi konfigurasi aplikasi kita di **config/auth.php**

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

```

- b.) Selanjutnya kita modifikasi sedikit pada **UserModel.php** untuk bisa melakukan proses otentikasi

```
6  use Illuminate\Database\Eloquent\Factories\HasFactory;
7  use Illuminate\Database\Eloquent\Model;
8  use Illuminate\Database\Eloquent\Relations\BelongsTo;
9  use Illuminate\Foundation\Auth\User as Authenticatable; //implementasi class authenticatable
10
11  class UserModel extends Authenticatable
12  {
13      use HasFactory;
14
15      protected $table = 'm_user'; //Mendefinisikan nama tabel yang digunakan model
16      protected $primaryKey = 'user_id'; //Mendefinisikan primary key dari tabel yang digunakan
17      /**
18       * The attribute that are mass assignable.
19       *
20       * @var array
21       */
22      protected $fillable = ['username', 'password', 'nama', 'level_id', 'create_at', 'update_at'];
23      protected $hidden = ['password']; //jangan di tampilkan saat select
24      protected $casts = ['password' => 'hashed']; //casting password agar otomatis di hash
25
26      public function level(): BelongsTo
27      {
28          return $this->belongsTo(LevelModel::class, 'level_id', 'level_id');
29      }
30  }

```

- c.) Selanjutnya kita buat **AuthController.php** untuk memproses login yang akan kita lakukan
- d.) Setelah kita membuat **AuthController.php**, kita buat view untuk menampilkan halaman login. View kita buat di **auth/login.blade.php**, tampilan login bisa kita ambil dari contoh login di template **AdminLTE** seperti berikut (pada contoh login ini, kita gunakan page **login-V2** di **AdminLTE**)

```
1  <?php
2
3  namespace App\Http\Controllers;
4  use Illuminate\Http\Request;
5  use Illuminate\Support\Facades\Auth;
6
7  class AuthController extends Controller
8  {
9      public function login()
10     {
11         if(Auth::check()){ // jika sudah login, maka redirect ke home
12             return redirect('/');
13         }
14         return view('auth.login');
15     }
```

```
16     public function postlogin(Request $request)
17     {
18         if($request->ajax() || $request->wantsJson()){
19             $credentials = $request->only('username', 'password');
20
21             if (Auth::attempt($credentials)) {
22                 return response()->json([
23                     'status' => true,
24                     'message' => 'Login Berhasil',
25                     'redirect' => url('/')
26                 ]);
27             }
```

```

29         return response()->json([
30             'status' => false,
31             'message' => 'Login Gagal'
32         ]);
33     }
34     return redirect('login');
35 }
36 public function logout(Request $request)
37 {
38     Auth::logout();
39
40     $request->session()->invalidate();
41     $request->session()->regenerateToken();
42     return redirect('login');
43 }
44
45 }

```

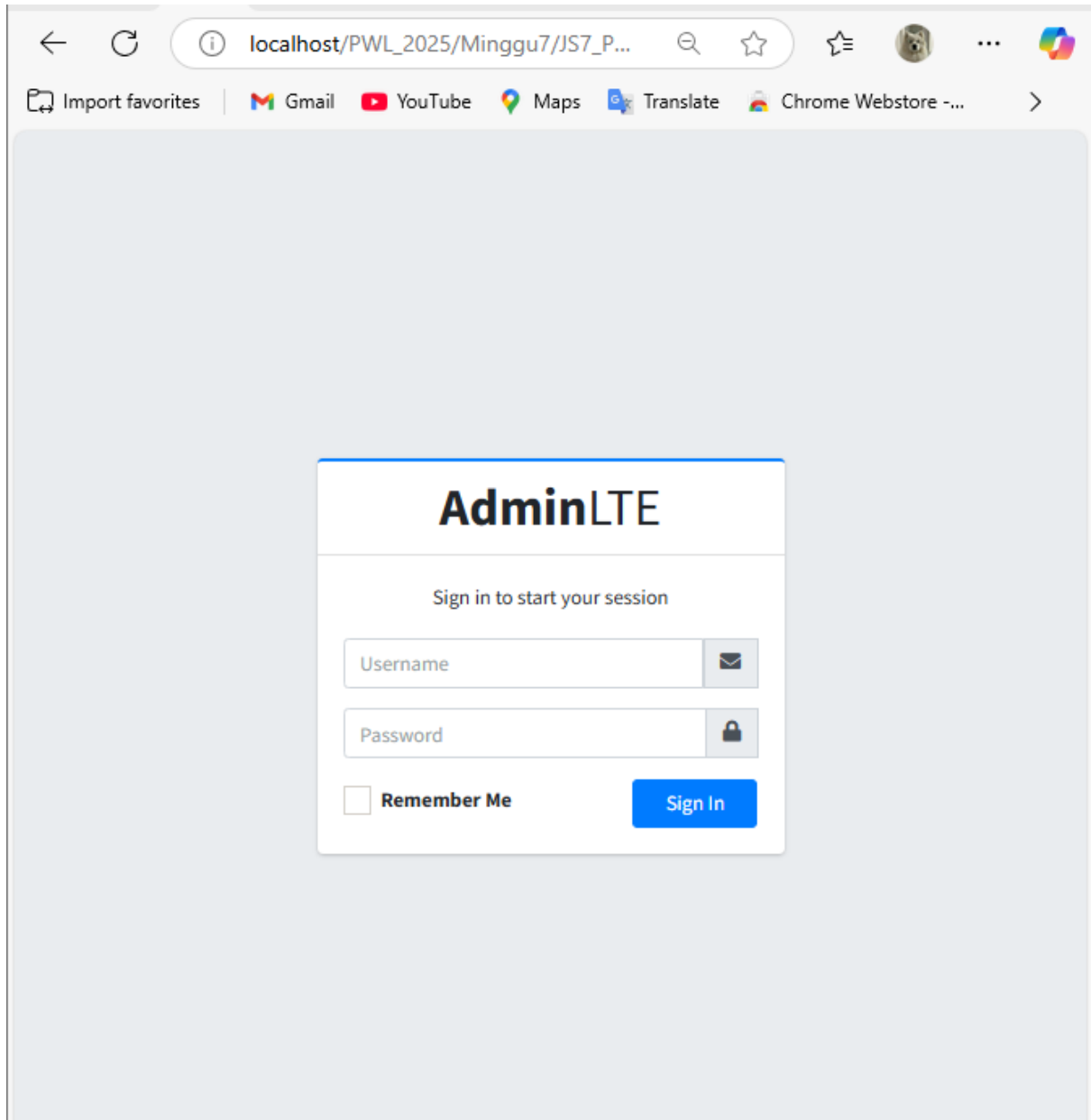
e.) Kemudian kita modifikasi `route/web.php` agar semua route masuk dalam auth

```

1  <?php
2
3  use Illuminate\Support\Facades\Route;
4  use App\Http\Controllers\LevelController;
5  use App\Http\Controllers\KategoriController;
6  use App\Http\Controllers\UserController;
7  use App\Http\Controllers>WelcomeController;
8  use App\Http\Controllers\SupplierController;
9  use App\Http\Controllers\BarangController;
10 use App\Http\Controllers\AuthController;
11
12 //Route::get('/', function () {
13     //return view('welcome');
14 //});
15 Route::pattern('id', '[0-9]+'); // artinya ketika ada parameter {id}, maka harus
16
17 Route::get('login', [AuthController::class, 'login'])->nama('login');
18 Route::post('login', [AuthController::class, 'postlogin']);
19 Route::get('logout', [AuthController::class, 'logout'])->middleware(['auth']);
20
21 Route::middleware(['auth'])->group(function(){
22

```

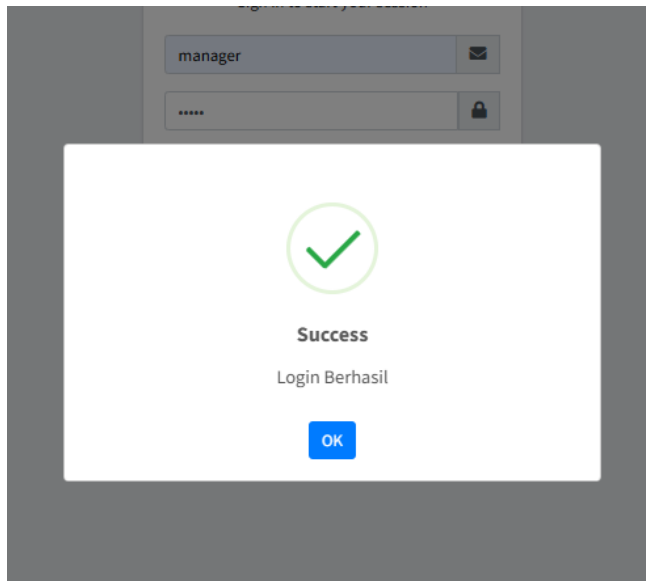
- f.) Ketika kita coba mengakses halaman `localhost/PWL_POS/public` maka akan tampil halaman awal untuk login ke aplikasi



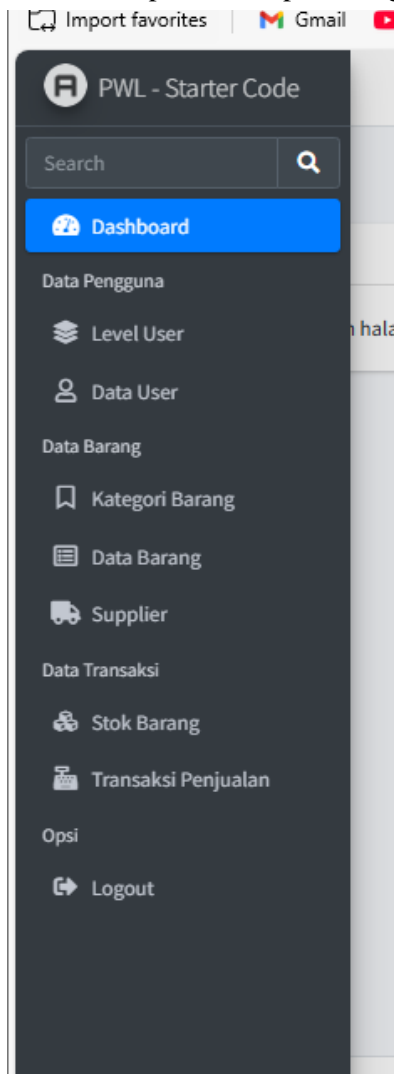
The screenshot shows a web browser window with the address bar displaying `localhost/PWL_2025/Minggu7/JS7_P...`. The browser's toolbar includes navigation buttons (back, forward, refresh), a search icon, and a star icon. Below the address bar, there are links for 'Import favorites', 'Gmail', 'YouTube', 'Maps', 'Translate', and 'Chrome Webstore -...'. The main content area of the browser displays a login form for 'AdminLTE'. The form is titled 'AdminLTE' and has a subtitle 'Sign in to start your session'. It contains two input fields: 'Username' and 'Password'. The 'Username' field has an envelope icon on the right, and the 'Password' field has a lock icon on the right. Below the 'Password' field, there is a checkbox labeled 'Remember Me' and a blue button labeled 'Sign In'.

Tugas 1 Implementasi Authentication

- a.) Silahkan implementasikan proses login pada project kalian masing-masing



- b.) Silahkan implementasi proses logout pada halaman web yang kalian buat



- c.) Amati dan jelaskan tiap tahapan yang kalian kerjakan, dan jabarkan dalam laporan
- d.) Submit kode untuk implementasi Authentication pada repository github kalian

Praktikum 2: Implementasi Authorization di Laravel dengan Middleware

- a.) Kita modifikasi `UserModel.php` dengan menambahkan kode berikut

```

27     public function getRoleName (){
28         return $this->level->level_nama;
29     }
30
31     public function hasRole ($role) : bool{
32         return $this->level->level_kode == $role;
33     }

```

- b.) Kemudian kita buat *middleware* dengan nama `AuthorizeUser.php`. Kita bisa buat *middleware* dengan mengetikkan perintah pada terminal/CMD
- c.) Kemudian kita edit *middleware* `AuthorizeUser.php` untuk bisa mengecek apakah pengguna yang mengakses memiliki Level/Role/Group yang sesuai

```

9     class AuthorizeUser
10    {
11        /**
12         * Handle an incoming request.
13         *
14         * @param  \Closure(\Illuminate\Http\Request): (\Symfony\Component\H
15         */
16        public function handle(Request $request, Closure $next, $role = ''):
17        {
18            $user = $request->user();
19
20            if($user->hasRole($role)){
21                return $next($request);
22            }
23            //jika tidak punya role, maka tampilkan error 403
24            abort(403, 'Forbidden. Kaamu tidak punya akses ke halaman ini');
25        }
26    }

```

- d.) Kita daftarkan ke `app/Http/Kernel.php` untuk *middleware* yang kita buat barusan

```

55    protected $middlewareAliases = [
56        'auth' => \App\Http\Middleware\Authenticate::class,
57        'authorize' => \App\Http\Middleware\AuthorizeUser::class,
58        'auth.basic' => \Illuminate\Auth\Middleware\AuthenticateWithB
59        'auth.session' => \Illuminate\Session\Middleware\Authenticate

```

- e.) Sekarang kita perhatikan tabel `m_level` yang menjadi tabel untuk menyimpan level/group/role dari user ada

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

		level_id	level_kode	level_nama	created_at	updated_at
<input type="checkbox"/>	Edit Copy Delete	1	ADM	Administrator	NULL	NULL
<input type="checkbox"/>	Edit Copy Delete	2	MNG	Manager	NULL	NULL
<input type="checkbox"/>	Edit Copy Delete	3	STF	Staff/Kasir	NULL	NULL
<input type="checkbox"/>	Edit Copy Delete	4	CUS	Customer	2025-03-07 09:54:41	NULL

☐ Check all | With selected: Edit Copy Delete Export

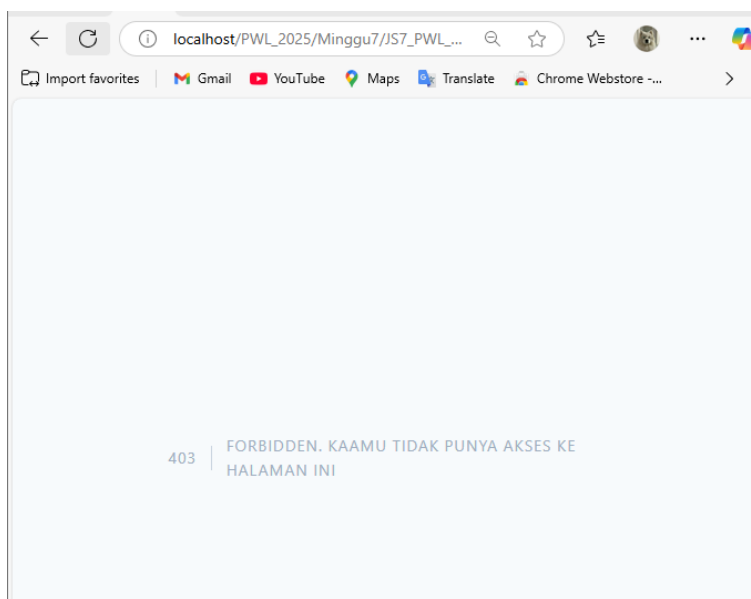
- f.) Untuk mencoba *authorization* yang telah kita buat, maka perlu kita modifikasi `route/web.php` untuk menentukan route mana saja yang akan diberi hak akses sesuai dengan level user

```

20
21 Route::middleware(['auth'])->group(function(){
22     Route::get('/', function () {
23         return view('welcome');
24     });
25     Route::get('/', [WelcomeController::class, 'index']);
26
27     Route::middleware(['authorize:ADM'])->group(function(){
28
29         Route::group(['prefix' => 'user'], function () {
30             Route::get('/', [UserController::class, 'index']); // menampilkan
31             Route::post('/list', [UserController::class, 'list']); // menampilkan
32             Route::get('/create', [UserController::class, 'create']); // menampilkan
33             Route::post('/update', [UserController::class, 'update']); // menampilkan

```

- g.) Pada kode yang ditandai merah, terdapat `authorize:ADM`. Kode `ADM` adalah nilai dari `level_kode` pada tabel `m_level`. Yang artinya, user yang bisa mengakses route untuk manage data level, adalah user yang memiliki level sebagai Administrator.
- h.) Untuk membuktikannya, sekarang kita coba login menggunakan akun selain level administrator, dan kita akses route menu level tersebut



Tugas 2 Implementasi Authorization

a.) Apa yang kalian pahami pada praktikum 2 ini?

Pada praktikum 2 ini, saya memahami konsep dan implementasi **authorization** dalam sistem Laravel. Authorization merupakan proses lanjutan setelah authentication berhasil, di mana sistem menentukan **hak akses** pengguna berdasarkan **peran atau level** yang dimilikinya. Melalui authorization, kita dapat membatasi fitur atau data apa saja yang bisa diakses oleh masing-masing pengguna sesuai tanggung jawabnya.

Contohnya, pengguna dengan peran **mahasiswa** hanya bisa mengakses fitur seperti melihat materi, mengerjakan tugas, dan mengikuti ujian. Sementara itu, pengguna dengan peran **dosen** memiliki hak untuk membuat materi, memberikan tugas, dan menilai ujian. Dalam Laravel, authorization bisa diimplementasikan menggunakan **middleware**, **gate**, atau **policy**, yang semuanya mempermudah pengelolaan kontrol akses terhadap fitur-fitur aplikasi.

Secara praktis, saya belajar bagaimana:

- Menentukan role pengguna (seperti admin, user biasa, dll).
- Mengatur akses ke route atau fitur tertentu sesuai role-nya.
- Mencegah akses tidak sah ke data atau fungsi yang tidak seharusnya digunakan oleh pengguna tertentu.

b.) Amati dan jelaskan tiap tahapan yang kalian kerjakan, dan jabarkan dalam laporan

c.) Submit kode untuk impementasi Authorization pada repository github kalian.

Praktikum 3: Implementasi Multi Level Authorization di Laravel dengan Middleware

a.) Kita modifikasi **UserModel.php** untuk mendapatkan level_kode dari user yang sudah login. Jadi kita buat fungsi dengan nama **getRole()**

```
35     public function getRole(){
36         return $this->level->level_kode;
37     }
```

b.) Selanjutnya, Kita modifikasi middleware **AuthorizeUser.php** dengan kode berikut

```
16     public function handle(Request $request, Closure $next,
17     {
18         $user_role = $request->user()->getRole();
19         if (in_array($user_role, $roles)) {
20             return $next($request);
21         }
22
23         abort(403, 'Forbidden, Kamu tidak punya akses ke halaman ini');
24     }
```

c.) Setelah itu tinggal kita perbaiki `route/web.php` sesuaikan dengan role/level yang diinginkan.

Contoh

```
25 Route::get('/', [WelcomeController::class, 'index']);
26
27 Route::middleware(['authorize:ADM,MNG'])->group(function () {
28
29     Route::group(['prefix' => 'user'], function () {
30         Route::get('/', [UserController::class, 'index']);
31         Route::post('/list', [UserController::class, 'list']);
32         Route::get('/create', [UserController::class, 'create']);
33         Route::post('/', [UserController::class, 'store']);
34         Route::get('/create_ajax', [UserController::class, 'create_ajax']);
35         Route::post('/ajax', [UserController::class, 'store_ajax']);
36         Route::get('/{id}', [UserController::class, 'show']);
37         Route::get('/{id}/edit', [UserController::class, 'edit']);
38         Route::put('/{id}', [UserController::class, 'update']);
39         Route::get('/{id}/edit_ajax', [UserController::class, 'edit_ajax']);
40         Route::put('/{id}/update_ajax', [UserController::class, 'update_ajax']);
41         Route::get('/{id}/delete_ajax', [UserController::class, 'delete_ajax']);
42         Route::delete('/{id}/delete_ajax', [UserController::class, 'delete_ajax']);
43         Route::delete('/{id}', [UserController::class, 'destroy']);
44     });
45 });
```

d.) Sekarang kita sudah bisa memberikan hak akses menu/route ke beberapa level user