

**Laporan Praktikum Pemrograman Web Lanjut**  
**Jobsheet 7: Authentication dan *Authorization* di Laravel**



Oleh :  
Danica Nasywa Putriniair (2341760122)  
Kelas SIB 2B / 05

**PROGRAM STUDI D-IV SISTEM INFORMASI BISNIS JURUSAN**  
**TEKNOLOGI INFORMASI**  
**POLITEKNIK NEGERI MALANG**

Jl. Soekarno Hatta No.9, Jatimulyo, Kec. Lowokwaru, Kota Malang, Jawa Timur  
65141

## Praktikum 1 - Implementasi Authentication

1. Kita buka project laravel PWL\_POS kita, dan kita modifikasi konfigurasi aplikasi kita di config/auth.php.

```
'providers' => [
    'users' => [
        'driver' => 'eloquent',
        'model' => App\Models\User::class,
    ],
],
```

Pada bagian ini kita sesuaikan dengan Model untuk tabel m\_user yang sudah kita buat.

```
'providers' => [
    'users' => [
        'driver' => 'eloquent',
        'model' => App\Models\UserModel::class,
    ],
],
```

2. Selanjutnya kita modifikasi sedikit pada UserModel.php untuk bisa melakukan proses autentikasi

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\Relations\BelongsTo;

class UserModel extends Model
{
    use HasFactory;

    protected $table = 'm_user'; // Mendefinisikan nama tabel yang digunakan
    protected $primaryKey = 'user_id'; // Mendefinisikan primary key dari tabel yang digunakan
    protected $fillable = ['level_id', 'username', 'nama', 'password'];

    protected $hidden = ['password'];

    protected $casts = ['password' => 'hashed'];

    public function level(): BelongsTo
    {
        return $this->belongsTo(LevelModel::class, 'level_id', 'level_id');
    }
}
```

3. Selanjutnya kita buat AuthController.php untuk memproses logi yang akan kita lakukan

```

app / Http / Controllers / AuthController.php / AuthController / logout
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6  use Illuminate\Support\Facades\Auth;
7
8  class AuthController extends Controller{
9      public function login() {
10         if (Auth::check()) {
11             return redirect('/');
12         }
13         return view('auth.login');
14     }
15
16     public function postlogin(Request $request) {
17         if ($request->ajax() || $request->wantsJson()) {
18             $credentials = $request->only('username', 'password');
19
20             if (Auth::attempt($credentials)) {
21                 return response()->json([
22                     'status' => true,
23                     'message' => 'Login Berhasil',
24                     'redirect' => url('/')
25                 ]);
26             }
27             return response()->json([
28                 'status' => false,
29                 'message' => 'Login Gagal',
30             ]);
31         }
32         return redirect('login');
33     }
34
35     public function logout(Request $request) {
36         Auth::logout();
37
38         $request->session()->invalidate();
39         $request->session()->regenerateToken();
40         return redirect('login');
41     }
42 }

```

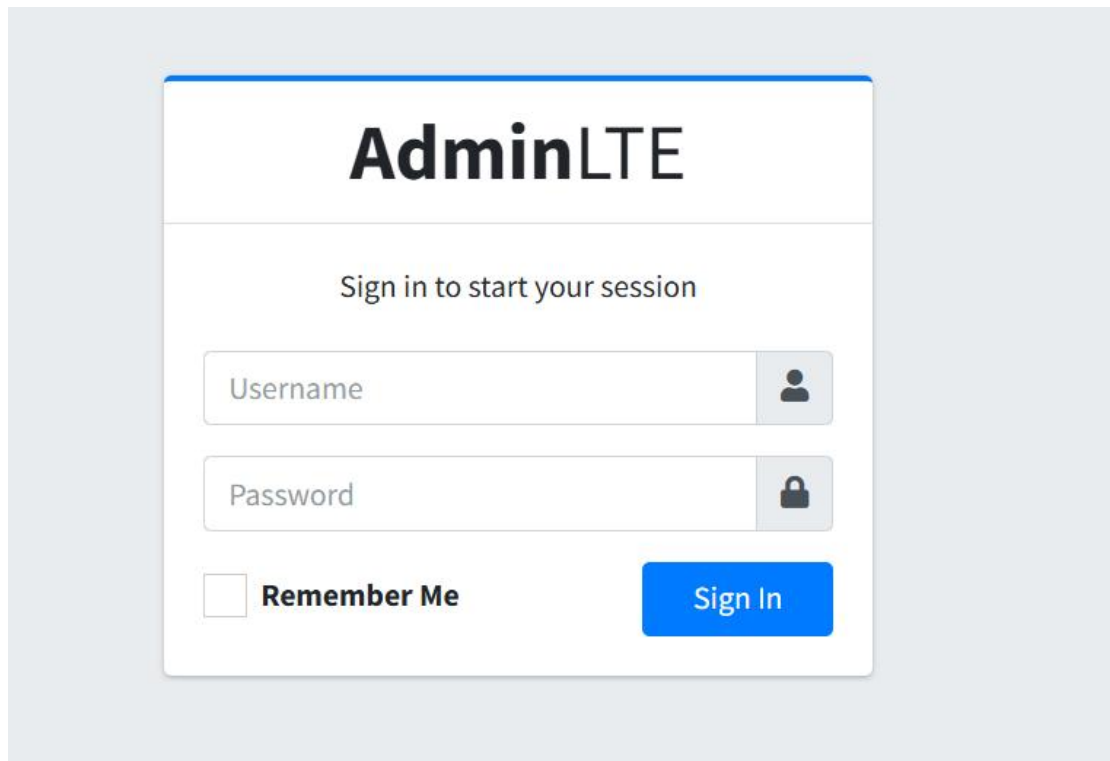
4. Setelah kita membuat AuthController.php, kita buat view untuk menampilkan halaman login. View kita buat di auth/login.blade.php , tampilan login bisa kita ambil dari contoh login di template AdminLTE seperti berikut (pada contoh login ini, kita gunakan page login-V2 di AdminLTE)

```
resources > views > auth > login.blade.php > html > body.hold-transition.login-page > script
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="UTF-8">
5 <meta name="viewport" content="width=device-width, initial-scale=1.0">
6 <title>Login Pengguna</title>
7
8 <!-- Google Font: Source Sans Pro -->
9 <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Source+Sans+Pro:300,400,400i,700&dis
10 <!-- Font Awesome -->
11 <link rel="stylesheet" href="{{ asset('plugins/fontawesome-free/css/all.min.css') }}">
12 <!-- icheck bootstrap -->
13 <link rel="stylesheet" href="{{ asset('plugins/icheck-bootstrap/icheck-bootstrap.min.css') }}">
14 <!-- SweetAlert2 -->
15 <link rel="stylesheet" href="{{ asset('plugins/sweetalert2-theme-bootstrap-4/bootstrap-4.min.css') }}">
16 <!-- Theme style --> <link rel="stylesheet" href="{{ asset('dist/css/adminlte.min.css') }}">
17 </head>
18 <body class="hold-transition login-page">
19 <div class="login-box">
20 <!-- /.login-logo -->
21 <div class="card card-outline card-primary">
22 <div class="card-header text-center">
23 <a href="{{ url('/') }}" class="h1"><b>Admin</b>LTE</a>
24 </div>
25 <div class="card-body">
26 <p class="login-box-msg">Sign in to start your session</p>
27 <form action="{{ url('login') }}" method="POST" id="form-login">
28 @csrf
29 <div class="input-group mb-3">
30 <input type="text" id="username" name="username" class="form-control" placeholder="Us
31 <div class="input-group-append">
32 <div class="input-group-text">
33 <span class="fas fa-envelope"></span>
34 </div>
35 </div>
36 <small id="error-username" class="error-text text-danger"></small>
37 </div>
38 <div class="input-group mb-3">
39 <input type="password" id="password" name="password" class="form-control" placeholder
40 <div class="input-group-append">
41 <div class="input-group-text">
42 <span class="fas fa-lock"></span>
43 </div>
44 </div>
45 <small id="error-password" class="error-text text-danger"></small>
46 </div> <div class="row">
47 <div class="col-8">
48 <div class="icheck-primary">
49 <input type="checkbox" id="remember">
50 <label for="remember">Remember Me</label>
51 </div>
52 </div>
53 <!-- /.col -->
54 <div class="col-4">
55 <button type="button" class="btn btn-block btn-primary">Sign In</button>
56 </div>
57 </div>
58 </div>
59 </div>
60 </body>
61 </html>
```

5. Kemudian kita modifikasi route/web.php agar semua route masuk dalam auth

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

6. Ketika kita coba mengakses halaman localhost/PWL\_POS/public maka akan tampil halaman awal untuk login ke aplikasi



### Tugas 1 - Implementasi Authentication

1. Silahkan implementasikan proses login pada project kalian masing - masing
2. Silahkan implementasi proses logout pada halaman web yang kalian buat

Menambahkan logout button pada header

```
@auth
<li class="nav-item dropdown">
  <a class="nav-link dropdown-toggle" data-toggle="dropdown" href="#" role="button">
    <i class="fas fa-user"></i> {{ Auth::user()->username }}
  </a>
  <div class="dropdown-menu dropdown-menu-right">
    <form action="{{ route('logout') }}" method="POST" class="d-inline">
      @csrf
      <button type="submit" class="dropdown-item">
        <i class="fas fa-sign-out-alt"></i> Logout
      </button>
    </form>
  </div>
</li>
@endauth
```

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

- Membuat Controller AuthController untuk menangani login dan logout
- Menambahkan route login/logout ke file web.php.
- Membuat halaman view login dengan form yang mendukung AJAX.
- Mengimplementasikan proses validasi dan autentikasi menggunakan Validator dan Auth::attempt.



- Membuat response AJAX dan redirect berdasarkan role/user state.
- Menambahkan fitur logout menggunakan Auth::logout dan mengamankan sesi.

4. Submit kode untuk implementasi Authentication pada repository github kalian.

- Submit Github Prak1

```
create mode 100644 Minggu 7/tests/Unit/ExampleTest.php
create mode 100644 Minggu 7/vite.config.js
PS E:\Aplik\laragon\www\Pemrograman_Web_Lanjut_2025\Minggu 7> git push
Enumerating objects: 2772, done.
Counting objects: 100% (2772/2772), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2062/2062), done.
Writing objects: 100% (2771/2771), 18.69 MiB | 1.61 MiB/s, done.
Total 2771 (delta 606), reused 2679 (delta 568), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (606/606), completed with 1 local object.
To https://github.com/snowvann/Pemrograman_Web_Lanjut_2025.git
6537581..cc113ed main -> main
```

- Submit Github Tugas1

```
it
PS E:\Aplik\laragon\www\Pemrograman_Web_Lanjut_2025\Minggu 7> git add .
PS E:\Aplik\laragon\www\Pemrograman_Web_Lanjut_2025\Minggu 7> git commit -m "JS 7 Tugas 1"
[main f863ca6] JS 7 Tugas 1
4 files changed, 58 insertions(+), 18 deletions(-)
PS E:\Aplik\laragon\www\Pemrograman_Web_Lanjut_2025\Minggu 7> git push
Enumerating objects: 29, done.
Counting objects: 100% (29/29), done.
Delta compression using up to 8 threads
Compressing objects: 100% (14/14), done.
Writing objects: 100% (15/15), 1.59 KiB | 812.00 KiB/s, done.
Total 15 (delta 13), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (13/13), completed with 13 local objects.
To https://github.com/snowvann/Pemrograman_Web_Lanjut_2025.git
cc113ed..f863ca6 main -> main
```

## Praktikum 2 - Implementasi *Authorization* di Laravel dengan Middleware

1. Kita modifikasi `UserModel.php` dengan menambahkan kode berikut

```
public $timestamps = false;

public function level(): BelongsTo{
    return $this->belongsTo(LevelModel::class, 'level_id', 'level_id');
}

public function getRoleName() : String {
    return $this->level->level_name;
}

public function hasRole($role) : bool {
    return $this->level->level_kode == $role;
}
```

2. Kemudian kita buat *middleware* dengan nama `AuthorizeUser.php`. Kita bisa buat *middleware* dengan mengetikkan perintah pada terminal/CMD

```
cc113ed..f863ca6 main -> main
PS E:\Aplik\laragon\www\Pemrograman_Web_Lanjut_2025\Minggu 7> php artisan make:middleware AuthorizeUser

INFO Middleware [E:\Aplik\laragon\www\Pemrograman_Web_Lanjut_2025\Minggu 7\app\Http\Middleware\AuthorizeUser.php] created successfully.
```

File *middleware* akan dibuat di `app/Http/Middleware/AuthorizeUser.php`

3. Kemudian kita edit *middleware* `AuthorizeUser.php` untuk bisa mengecek apakah pengguna yang mengakses memiliki Level/Role/Group yang sesuai

```
<?php

namespace App\Http\Middleware;

use Closure;
use Illuminate\Http\Request;
use Symfony\Component\HttpFoundation\Response;

class AuthorizeUser
{
    /**
     * Handle an incoming request.
     *
     * @param \Closure(\Illuminate\Http\Request): (\Symfony\Component\HttpFoundation\Response) $next
     */
    public function handle(Request $request, Closure $next, $role = ' '): Response
    {
        $user = $request->user();
        if ($user->hasRole($role)) {
            return $next($request);
        }
        abort(403, 'Forbidden, Kamu tidak punya akses ke halaman ini');
    }
}
```

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

```

protected $middlewareAliases = [
    'auth' => \App\Http\Middleware\Authenticate::class,
    'authorize' => \App\Http\Middleware\AuthorizeUser::class, // middleware yang kita buat
    'auth.basic' => \Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class,
    'auth.session' => \Illuminate\Session\Middleware\AuthenticateSession::class,
    'cache.headers' => \Illuminate\Http\Middleware\SetCacheHeaders::class,
    'can' => \Illuminate\Auth\Middleware\Authorize::class,
    'guest' => \App\Http\Middleware\RedirectIfAuthenticated::class,
    'password.confirm' => \Illuminate\Auth\Middleware\RequirePassword::class,
    'precognitive' => \Illuminate\Foundation\Http\Middleware\HandlePrecognitiveRequests::class,
    'signed' => \App\Http\Middleware\ValidateSignature::class,
    'throttle' => \Illuminate\Routing\Middleware\ThrottleRequests::class,
    'verified' => \Illuminate\Auth\Middleware\EnsureEmailIsVerified::class,
];

```

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

	level_id	level_kode	level_name	created_at	updated_at
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	ADM	Administrator	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	MNG	Manager	NULL	NULL
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	STF	Staff/Kasir	NULL	NULL

6. 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

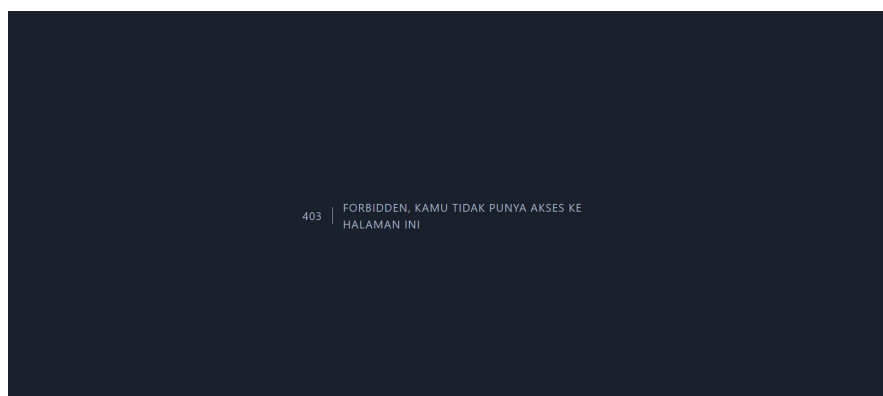
```

// LEVEL
Route::middleware('authorize:ADM')->group(function () {
    Route::get('/', [LevelController::class, 'index']);
    Route::post('/list', [LevelController::class, 'list']);
    Route::get('/create', [LevelController::class, 'create']);
});

```

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.

7. Untuk membuktikannya, sekarang kita coba login menggunakan akun selain level administrator, dan kita akses route menu level tersebut.





## Tugas 2 - Implementasi Authorizon

### 1. Apa yang kalian pahami pada praktikum 2 ini?

Memahami cara mengatur hak akses menggunakan middleware dan konsep role-based access control.

### 2. Amati dan jelaskan tiap tahapan yang kalian kerjakan, dan jabarkan dalam laporan

Untuk mengimplementasikan authorization, pertama kita modifikasi User.php untuk menambahkan relasi ke tabel level. Selanjutnya, kita buat middleware bernama AuthorizeUser.php menggunakan perintah artisan, lalu edit isinya agar dapat mengecek apakah user memiliki level yang sesuai. Middleware ini kemudian didaftarkan di app/Http/Kernel.php. Pastikan tabel m\_level berisi data level/kode role seperti "ADM" untuk administrator. Setelah itu, pada file routes/web.php, kita tetapkan middleware authorize:ADM pada route tertentu agar hanya bisa diakses oleh user dengan level administrator. Untuk mengujinya, login menggunakan akun dengan level selain administrator lalu akses route tersebut—maka sistem akan menolak akses sesuai pengaturan authorization yang telah dibuat.

### 3. Submit kode untuk impementasi Authorization pada repository github kalian.

```
PS E:\Applik\laragon\www\Pemrograman_Web_Lanjut_2025\Minggu 7> git add .
PS E:\Applik\laragon\www\Pemrograman_Web_Lanjut_2025\Minggu 7> git commit -m "Js 7 Tugas 2"
[main 4479acc] Js 7 Tugas 2
4 files changed, 34 insertions(+), 1 deletion(-)
create mode 100644 Minggu 7/app/Http/Middleware/AuthorizeUser.php
PS E:\Applik\laragon\www\Pemrograman_Web_Lanjut_2025\Minggu 7> git push
Enumerating objects: 22, done.
Counting objects: 100% (22/22), done.
Delta compression using up to 8 threads
Compressing objects: 100% (12/12), done.
Writing objects: 100% (12/12), 1.44 KiB | 211.00 KiB/s, done.
Total 12 (delta 8), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (8/8), completed with 8 local objects.
To https://github.com/snowvann/Pemrograman_Web_Lanjut_2025.git
f863ca6..4479acc main -> main
```

### Praktikum 3 - Implementasi Multi\_Level Authorizaton di Laravel dengan Middleware

1. Kita modifikasi UserModel.php untuk mendapatkan level\_kode dari user yang sudah login.

Jadi kita buat fungsi dengan nama getRole()

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

2. Selanjutnya, kita modifikasi middleware AuthorizerUser.php dengan kode berikut

```
namespace App\Http\Middleware;  
  
use Closure;  
use Illuminate\Http\Request;  
use Symfony\Component\HttpFoundation\Response;  
  
class AuthorizeUser  
{  
    /**  
     * Handle an incoming request.  
     *  
     * @param  \Closure(\Illuminate\Http\Request): (\Symfony\Component\HttpFoundation\Response)  $next  
     */  
    public function handle(Request $request, Closure $next, ... $roles): Response  
    {  
        $user_role = $request->user()->getRole(); // ambil data level_kode dari user yang log  
        if (in_array($user_role, $roles)) { //cek apakah level_kode user ada di dalam array ro  
            return $next($request); // jika ada, maka lanjutan request  
        }  
        // jika tidak punya role, maka tampilan error 403  
        abort(403, 'Forbidden, Kamu tidak punya akses ke halaman ini');  
    }  
}
```

3. Setelah itu tinggal kita perbaiki route/web.php sesuaikan dengan role/level yang diinginkan. Contoh

```
// BARANG  
Route::middleware('authorize:ADM,MNG')->group(function () {  
    Route::get('/', [BarangController::class, 'index']);  
    Route::post('/list', [BarangController::class, 'list']);  
    Route::get('/create', [BarangController::class, 'create']);  
    Route::post('/', [BarangController::class, 'store']);  
    Route::get('/{id}/edit', [BarangController::class, 'edit']);  
    Route::put('/{id}', [BarangController::class, 'update']);  
    Route::delete('/{id}', [BarangController::class, 'destroy']);  
    Route::get('/{id}', [BarangController::class, 'show']);  
  
    // AJAX  
    Route::get('/create_ajax', [BarangController::class, 'create_ajax'])->name('barang.create_ajax');  
    Route::post('/ajax', [BarangController::class, 'store_ajax'])->name('barang.store_ajax');  
    Route::get('/{id}/edit_ajax', [BarangController::class, 'edit_ajax'])->name('barang.edit_ajax');  
    Route::put('/{id}/update_ajax', [BarangController::class, 'update_ajax'])->name('barang.update_ajax');  
    Route::delete('/{id}/delete_ajax', [BarangController::class, 'delete_ajax'])->name('barang.delete_ajax');  
    Route::post('/check_unique/{id?}', [BarangController::class, 'checkUnique'])->name('barang.check_unique');  
});
```

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

### Tugas 3 - Implementasi Multi-Level Authorization

1. Silahkan implementasikan multi-level authorization pada project kalian masing-masing
2. Amati dan jelaskan tiap tahapan yang kalian kerjakan, dan jabarkan dalam laporan

Pada praktikum ini, pertama-tama kita memodifikasi UserModel.php dengan menambahkan fungsi getRole() untuk mengambil nilai level\_kode dari user yang sedang login. Selanjutnya, kita mengedit middleware AuthorizeUser.php agar dapat memeriksa apakah user memiliki level/role yang sesuai untuk mengakses suatu route. Setelah middleware selesai, kita sesuaikan route/web.php dengan menambahkan middleware authorize pada route tertentu sesuai dengan level user yang diinginkan, misalnya authorize:ADM untuk Administrator. Dengan langkah-langkah tersebut, kita sudah berhasil memberikan hak akses ke menu atau route tertentu sesuai dengan level user yang login.

3. Implementasikan multi-level authorization untuk semua Level/Jenis User dan Menu-menu yang sesuai dengan Level/Jenis User

```
// Global pattern untuk parameter 'id'
Route::pattern('id', '[0-9]+');

// ===== AUTH =====
Route::get('/login', [AuthController::class, 'login'])->name('login');
Route::post('/login', [AuthController::class, 'postlogin']);
Route::post('/logout', [AuthController::class, 'logout'])->name('logout');

// ===== PROTECTED ROUTES =====
Route::middleware(['auth'])->group(function () {
    Route::get('/', [WelcomeController::class, 'index']);

    // ===== USER (ADMIN ONLY) =====
    Route::middleware(['authorize:ADM'])->prefix('user')->group(function () {
        Route::get('/', [UserController::class, 'index']);
        Route::post('/list', [UserController::class, 'list']);
        Route::get('/create', [UserController::class, 'create']);
        Route::post('/', [UserController::class, 'store']);
        Route::get('/create_ajax', [UserController::class, 'create_ajax']);
        Route::post('/ajax', [UserController::class, 'store_ajax']);
        Route::get('/{id}', [UserController::class, 'show']);
        Route::get('/{id}/edit', [UserController::class, 'edit']);
        Route::put('/{id}', [UserController::class, 'update']);
        Route::get('/{id}/edit_ajax', [UserController::class, 'edit_ajax']);
        Route::put('/{id}/update_ajax', [UserController::class, 'update_ajax']);
        Route::get('/{id}/delete_ajax', [UserController::class, 'confirm_ajax']);
        Route::delete('/{id}/delete_ajax', [UserController::class, 'delete_ajax']);
        Route::delete('/{id}', [UserController::class, 'destroy']);
    });

    // ===== LEVEL (ADMIN ONLY) =====
    Route::middleware(['authorize:ADM'])->prefix('level')->group(function () {
        Route::get('/', [LevelController::class, 'index']);
        Route::post('/list', [LevelController::class, 'list']);
        Route::get('/create', [LevelController::class, 'create']);
        Route::post('/', [LevelController::class, 'store']);
        Route::get('/create_ajax', [LevelController::class, 'create_ajax']);
        Route::post('/ajax', [LevelController::class, 'store_ajax']);
        Route::get('/{id}', [LevelController::class, 'show']);
        Route::get('/{id}/edit', [LevelController::class, 'edit']);
        Route::put('/{id}', [LevelController::class, 'update']);
        Route::get('/{id}/edit_ajax', [LevelController::class, 'edit_ajax']);
        Route::put('/{id}/update_ajax', [LevelController::class, 'update_ajax']);
        Route::get('/{id}/delete_ajax', [LevelController::class, 'confirm_ajax']);
        Route::delete('/{id}/delete_ajax', [LevelController::class, 'delete_ajax']);
        Route::delete('/{id}', [LevelController::class, 'destroy']);
    });

    // ===== KATEGORI (ADMIN & MANAGER) =====
    Route::middleware(['authorize:ADM,MNG'])->prefix('kategori')->group(function () {
        Route::get('/', [KategoriController::class, 'index']);
        Route::post('/list', [KategoriController::class, 'list']);
        Route::get('/create', [KategoriController::class, 'create']);
        Route::post('/', [KategoriController::class, 'store']);
    });
});
```

4. Submit kode untuk implementasi Authorization pada repository github kalian.

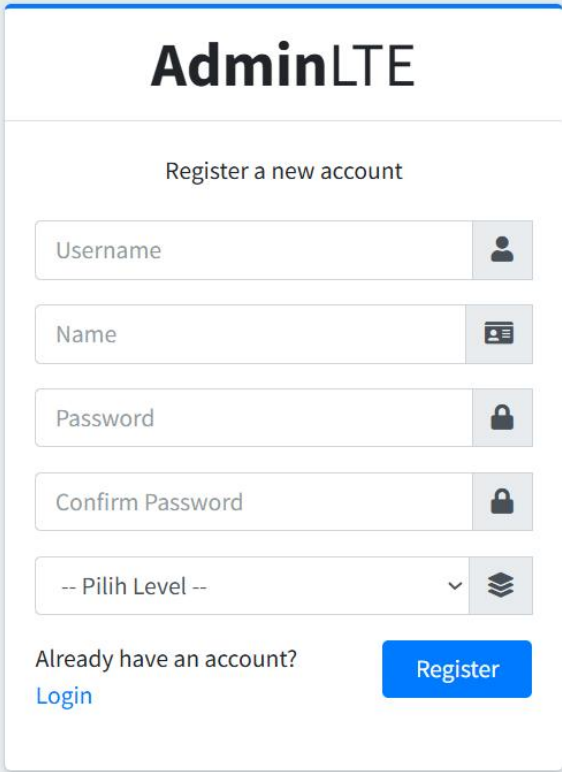
```

k\laragon\www\Pemrograman_Web_Lanjut_2025\Minggu 7> git add .
PS E:\Aplik\laragon\www\Pemrograman_Web_Lanjut_2025\Minggu 7> git commit -m "JS 7 Tugas 3"
[main 101933e] JS 7 Tugas 3
 4 files changed, 61 insertions(+), 62 deletions(-)
PS E:\Aplik\laragon\www\Pemrograman_Web_Lanjut_2025\Minggu 7> git push
Enumerating objects: 29, done.
Counting objects: 100% (29/29), done.
Delta compression using up to 8 threads
Compressing objects: 100% (15/15), done.
Writing objects: 100% (15/15), 1.90 KiB | 648.00 KiB/s, done.
Total 15 (delta 13), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (13/13), completed with 13 local objects.
To https://github.com/snowvann/Pemrograman_Web_Lanjut_2025.git
 4479acc..101933e main -> main

```


#### Tugas 4 - Implementasi Form Registrasi


1. Silahkan implementasikan form untuk registrasi user.
2. Screenshot hasil yang kalian kerjakan





**AdminLTE**


Register a new account

Username 

Name 

Password 

Confirm Password 

-- Pilih Level -- 

Already have an account? [Login](#) [Register](#)

3. Commit dan push hasil tugas kalian ke masing-masing repo github kalian



```
warning: in the working copy of Minggu 7/resources/views/auth/register.blade.php, CRLF will be replaced by LF.
PS E:\Aplik\laragon\www\Pemrograman_Web_Lanjut_2025\Minggu 7> git add .
PS E:\Aplik\laragon\www\Pemrograman_Web_Lanjut_2025\Minggu 7> git commit -m "JS 7 Tugas 4"
[main f7cf4e4] JS 7 Tugas 4
 4 files changed, 256 insertions(+), 2 deletions(-)
  create mode 100644 Minggu 7/resources/views/auth/register.blade.php
PS E:\Aplik\laragon\www\Pemrograman_Web_Lanjut_2025\Minggu 7> git push
Enumerating objects: 26, done.
Counting objects: 100% (26/26), done.
Delta compression using up to 8 threads
Compressing objects: 100% (14/14), done.
Writing objects: 100% (14/14), 3.58 KiB | 1.79 MiB/s, done.
Total 14 (delta 10), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (10/10), completed with 10 local objects.
To https://github.com/snowvann/Pemrograman_Web_Lanjut_2025.git
 101933e..f7cf4e4  main -> main
PS E:\Aplik\laragon\www\Pemrograman_Web_Lanjut_2025\Minggu 7>
```