



Mata Kuliah : Pemrograman Web Lanjut (PWL)
Program Studi : D4 – Teknik Informatika / D4 – Sistem Informasi Bisnis
Semester : 4 (empat) / 5 (lima)
Pertemuan ke- : 7 (tujuh)
Nama : Aditya Yuhanda Putra

JOBSHEET 07

Authentication dan *Authorization* di Laravel

Laravel Authentication dipergunakan untuk memproteksi halaman atau fitur dari web yang hanya diakses oleh orang tertentu yang diberikan hak. Fitur seperti ini biasanya ditemui di sistem yang memiliki fitur administrator atau sistem yang memiliki pengguna yang boleh menambahkan datanya.

Laravel membuat penerapan otentikasi sangat sederhana dan telah menyediakan berbagai fitur yang dapat dimanfaatkan tanpa perlu melakukan penambahan instalasi modul tertentu. File konfigurasi otentikasi terletak di `config / auth.php`, yang berisi beberapa opsi yang terdokumentasi dengan baik untuk mengubah konfigurasi dari layanan otentikasi.

Pada intinya, fasilitas otentikasi Laravel terdiri dari “*guards*” dan “*providers*”. *Guards* menentukan bagaimana pengguna diautentikasi untuk setiap permintaan. Misalnya, Laravel mengirim dengan *guards* untuk sesi dengan menggunakan penyimpanan session dan cookie.

Middleware

Middleware adalah lapisan perantara antara permintaan *route HTTP* yang masuk dan *action* dari Controller yang akan dijalankan. **Middleware** memungkinkan kita untuk melakukan berbagai tugas baik itu sebelum ataupun sesudah tindakan dilakukan. Kita juga dapat menggunakan *tool CLI* untuk membuat sebuah **Middleware** dalam **Laravel**. Beberapa contoh penggunaan **Middleware** meliputi autentikasi, validasi, manipulasi permintaan, dan lainnya. Berikut di bawah ini adalah manfaat dari **Middleware** :

- **Keamanan** : dalam **Middleware** memungkinkan kita untuk memverifikasi apakah pengguna sudah diautentikasi sebelum mengakses halaman tertentu. Dengan demikian, kita dapat melindungi data sensitif dan mengontrol hak akses pengguna.
- **Pemfilteran Data** : **Middleware** dapat digunakan untuk memanipulasi data permintaan sebelum sebuah *action* dalam *controller* dilakukan. Misalnya, kita dapat memeriksa terlebih dahulu data yang dikirim oleh pengguna sebelum data tersebut diproses lebih



lanjut atau kita ingin memodifikasi data yang akan dikirim lalu kita dapat memeriksa ulang data yang akan dikirim oleh pengguna sebelum data tersebut diproses.

- **Logging dan Audit : Middleware** juga dapat digunakan untuk mencatat aktivitas pengguna atau melakukan audit terhadap permintaan yang masuk. Ini dapat membantu dalam pemantauan dan analisis aplikasi.

INFO

Kita akan menggunakan Laravel Auth secara manual seperti
<https://laravel.com/docs/10.x/authentication#authenticating-users>

Sesuai dengan **Studi Kasus PWL.pdf**.

Jadi project Laravel 10 kita masih sama dengan menggunakan repositori **PWL_POS**.

Project PWL_POS akan kita gunakan sampai pertemuan 12 nanti, sebagai project yang akan kita pelajari

A. Implementasi Manual Authentication di Laravel

Autentikasi adalah proses untuk memverifikasi identitas pengguna yang mencoba mengakses sistem. Dalam konteks aplikasi web, autentikasi memastikan bahwa pengguna yang mencoba login memiliki hak akses yang sesuai berdasarkan kredensial seperti email dan password. Proses autentikasi berbeda dengan **otorisasi**, yang merupakan langkah lanjutan untuk menentukan hak akses apa yang dimiliki pengguna setelah mereka berhasil diautentikasi.

Konsep Autentikasi di Laravel

Laravel menawarkan sistem autentikasi yang sangat fleksibel. Laravel menyediakan mekanisme autentikasi bawaan melalui layanan authentication scaffolding seperti *Laravel Jetstream* dan *Breeze*, yang dapat secara otomatis menghasilkan halaman dan logika autentikasi. Namun, terkadang pengembang memerlukan implementasi autentikasi yang lebih manual untuk memberikan kontrol penuh terhadap setiap aspek dari proses tersebut.

Beberapa komponen penting dalam sistem autentikasi Laravel meliputi:

- *Guard*: Komponen yang mengatur bagaimana pengguna diautentikasi untuk setiap permintaan. *Guard* default menggunakan sesi dan cookie.



- *Provider*: Mengatur bagaimana pengguna diambil dari database atau sumber data lainnya. *Provider* default mengambil data pengguna dari database dengan menggunakan Eloquent ORM.
- *Session*: Laravel menggunakan sesi untuk menyimpan status autentikasi pengguna. Sesi memungkinkan sistem untuk mengingat pengguna yang sudah login di antara permintaan HTTP yang berbeda.

Alur umum dari autentikasi meliputi:

1. *Login*: Pengguna mengirimkan kredensial (biasanya berupa email dan password).
2. *Verifikasi Kredensial*: Sistem memeriksa apakah kredensial yang diberikan sesuai dengan data di database.
3. *Pembuatan Sesi*: Jika kredensial benar, sistem akan membuat sesi untuk pengguna yang akan disimpan di server.
4. *Akses ke Halaman yang Dilindungi*: Pengguna yang terautentikasi dapat mengakses halaman-halaman yang dilindungi oleh *middleware* auth.
5. *Logout*: Pengguna bisa keluar dari sistem dan sesi mereka akan dihapus.

Middleware Autentikasi

Middleware auth di Laravel digunakan untuk melindungi rute atau halaman agar hanya dapat diakses oleh pengguna yang sudah terautentikasi. Jika pengguna mencoba mengakses rute yang memerlukan autentikasi tanpa login, mereka akan diarahkan ke halaman login.

- **Guard** bertanggung jawab untuk menangani proses autentikasi pengguna. Laravel secara default menggunakan *guard* berbasis sesi untuk autentikasi web, namun juga mendukung *guard* berbasis token (seperti API).
- **Provider** bertugas untuk mengambil pengguna dari database. Laravel menyediakan *provider* default yang menggunakan Eloquent, namun juga mendukung *provider* lain seperti Query Builder.

Implementasi di Laravel 10

Kita akan menerapkan penggunaan authentication di Laravel. Dalam penerapan ini, kita akan mencoba membuat otentikasi secara di Laravel, agar kita paham langkah-langkah dalam membuat Authentication



Praktikum 1 – Implementasi Authentication :

1. 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\User::class,  
66         ],  
    ],
```

Pada bagian ini kita sesuaikan dengan Model untuk tabel m_user yang sudah kita buat

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

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

```
<?php  
namespace App\Models;  
  
use Illuminate\Database\Eloquent\Model;  
use Illuminate\Database\Eloquent\Factories\HasFactory;  
use Illuminate\Database\Eloquent\Relations\BelongsTo;  
use Illuminate\Foundation\Auth\User as Authenticatable; // implementasi class Authenticatable  
  
class UserModel extends Authenticatable  
{  
    use HasFactory;  
  
    protected $table = 'm_user';  
    protected $primaryKey = 'user_id';  
    protected $fillable = ['username', 'password', 'nama', 'level_id', 'created_at', 'updated_at'];  
  
    protected $hidden    = ['password']; // jangan di tampilkan saat select  
  
    protected $casts     = ['password' => 'hashed']; // casting password agar otomatis di hash  
  
    /**  
     * Relasi ke tabel level  
     */  
    public function level(): BelongsTo  
    {  
        return $this->belongsTo(LevelModel::class, 'level_id', 'level_id');  
    }  
}
```



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

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;

class AuthController extends Controller
{
    public function login()
    {
        if(Auth::check()){ // jika sudah login, maka redirect ke halaman home
            return redirect('/');
        }
        return view('auth.login');
    }

    public function postlogin(Request $request)
    {
        if($request->ajax() || $request->wantsJson()){
            $credentials = $request->only('username', 'password');

            if (Auth::attempt($credentials)) {
                return response()->json([
                    'status' => true,
                    'message' => 'Login Berhasil',
                    'redirect' => url('/')
                ]);
            }

            return response()->json([
                'status' => false,
                'message' => 'Login Gagal'
            ]);
        }

        return redirect('login');
    }

    public function logout(Request $request)
    {
        Auth::logout();

        $request->session()->invalidate();
        $request->session()->regenerateToken();
        return redirect('login');
    }
}
```

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**)

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Login Pengguna</title>
```



```
<!-- Google Font: Source Sans Pro -->
<link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Source+Sans+Pro:300,400,400i,700&display=fallback">
<!-- Font Awesome -->
<link rel="stylesheet" href="{ asset('plugins/fontawesome-free/css/all.min.css') }">
<!-- icheck bootstrap -->
<link rel="stylesheet" href="{ asset('plugins/icheck-bootstrap/icheck-bootstrap.min.css')
}]">
<!-- SweetAlert2 -->
<link rel="stylesheet" href="{ asset('plugins/sweetalert2-theme-bootstrap-4/bootstrap-
4.min.css') }]">
<!-- Theme style -->
<link rel="stylesheet" href="{ asset('dist/css/adminlte.min.css') }]">
</head>
<body class="hold-transition login-page">
<div class="login-box">
  <!-- /.login-logo -->
  <div class="card card-outline card-primary">
    <div class="card-header text-center"><a href="{ url('/') }"
class="h1"><b>Admin</b>LTE</a></div>
    <div class="card-body">
      <p class="login-box-msg">Sign in to start your session</p>
      <form action="{ url('login') }" method="POST" id="form-login">
        @csrf
        <div class="input-group mb-3">
          <input type="text" id="username" name="username" class="form-control"
placeholder="Username">
          <div class="input-group-append">
            <div class="input-group-text">
              <span class="fas fa-envelope"></span>
            </div>
          </div>
          <small id="error-username" class="error-text text-danger"></small>
        </div>
        <div class="input-group mb-3">
          <input type="password" id="password" name="password" class="form-control"
placeholder="Password">
          <div class="input-group-append">
            <div class="input-group-text">
              <span class="fas fa-lock"></span>
            </div>
          </div>
          <small id="error-password" class="error-text text-danger"></small>
        </div>
        <div class="row">
          <div class="col-8">
            <div class="icheck-primary">
              <input type="checkbox" id="remember"><label for="remember">Remember Me</label>
            </div>
          <!-- /.col -->
          <div class="col-4">
            <button type="submit" class="btn btn-primary btn-block">Sign In</button>
          </div>
          <!-- /.col -->
        </div>
      </form>
    </div>
    <!-- /.card-body -->
  </div>
  <!-- /.card -->
</div>
<!-- /.login-box -->

<!-- jQuery -->
```



```
<script src="{{ asset('plugins/jquery/jquery.min.js') }}"></script>
<!-- Bootstrap 4 -->
<script src="{{ asset('plugins/bootstrap/js/bootstrap.bundle.min.js') }}"></script>
<!-- jquery-validation -->
<script src="{{ asset('plugins/jquery-validation/jquery.validate.min.js') }}"></script>
<script src="{{ asset('plugins/jquery-validation/additional-methods.min.js') }}"></script>
<!-- SweetAlert2 -->
<script src="{{ asset('plugins/sweetalert2/sweetalert2.min.js') }}"></script>
<!-- AdminLTE App -->
<script src="{{ asset('dist/js/adminlte.min.js') }}"></script>

<script>
$.ajaxSetup({
  headers: {
    'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('content')
  }
});

$(document).ready(function() {
  $("#form-login").validate({
    rules: {
      username: {required: true, minlength: 4, maxlength: 20},
      password: {required: true, minlength: 6, maxlength: 20}
    },
    submitHandler: function(form) { // ketika valid, maka bagian yg akan dijalankan
      $.ajax({
        url: form.action,
        type: form.method,
        data: $(form).serialize(),
        success: function(response) {
          if(response.status){ // jika sukses
            Swal.fire({
              icon: 'success',
              title: 'Berhasil',
              text: response.message,
            }).then(function() {
              window.location = response.redirect;
            });
          }else{ // jika error
            $('.error-text').text('');
            $.each(response.msgField, function(prefix, val) {
              $('#error-'+prefix).text(val[0]);
            });
            Swal.fire({
              icon: 'error',
              title: 'Tenjadi Kesalahan',
              text: response.message
            });
          }
        }
      });
    },
    errorElement: 'span',
    errorPlacement: function (error, element) {
      error.addClass('invalid-feedback');
      element.closest('.input-group').append(error);
    },
    highlight: function (element, errorClass, validClass) {
      $(element).addClass('is-invalid');
    },
    unhighlight: function (element, errorClass, validClass) {
      $(element).removeClass('is-invalid');
    }
  });
});
</script>
```




```
</body>  
</html>
```

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

```
<?php  
  
use App\Http\Controllers\UserController;  
use App\Http\Controllers\SupplierController;  
use App\Http\Controllers\BarangController;  
use App\Http\Controllers\AuthController;  
use App\Http\Controllers\KategoriController;  
use App\Http\Controllers\LevelController;  
use App\Http\Controllers>WelcomeController;  
use Illuminate\Support\Facades\Route;  
  
Route::pattern('id', '[0-9]+'); // artinya ketika ada parameter {id}, maka harus berupa angka  
  
Route::get('login', [AuthController::class, 'login'])->name('login');  
Route::post('login', [AuthController::class, 'postlogin']);  
Route::get('logout', [AuthController::class, 'logout'])->middleware('auth');  
  
Route::middleware(['auth'])->group(function(){ // artinya semua route di dalam group ini harus login dulu  
    // masukkan semua route yang perlu autentikasi di sini  
});
```

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
 - Implementasi proses login sudah mengikuti langkah praktikum 1 Jobsheet diatas.
2. Silahkan implementasi proses logout pada halaman web yang kalian buat
 - Baik saya akan buat.



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

➤ Mengupdate isi dari function logout pada **AuthController.php**.

```
public function logout(Request $request): Response {  
    Auth::logout();  
    $request->session()->invalidate();  
    $request->session()->regenerateToken();  
  
    return redirect(to: '/login');  
}
```

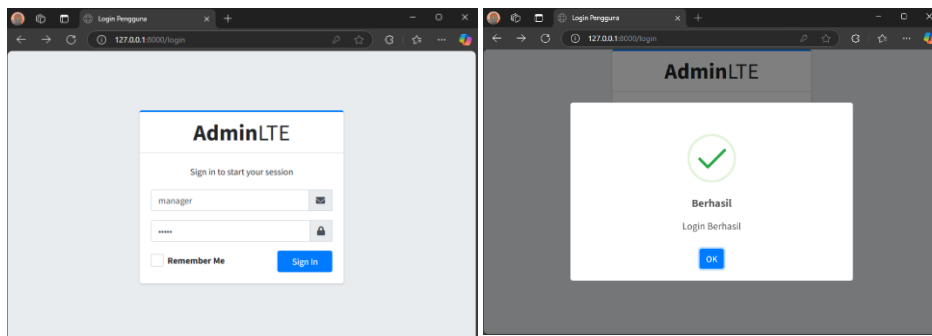
➤ Mengupdate Route logout di **web.php**.

```
Route::post(uri: 'login', action: [AuthController::class, 'postLogin']);  
Route::post(uri: 'logout', action: [AuthController::class, 'logout'])->name(name: 'logout')->middleware(middleware: 'auth');  
// Formulir login di bawah ini harus bisa diakses jika sudah login
```

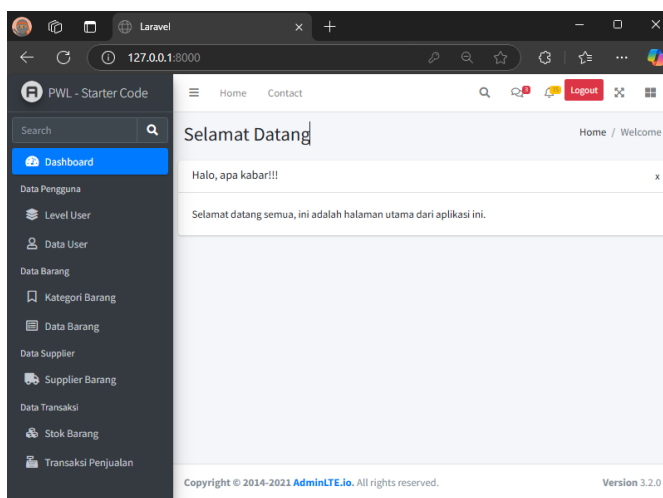
➤ Menambahkan tombol Logout pada tampilan di **header.blade.php**.

```
124 <li class="nav-item">  
125 <form method="POST" action="{{ route('logout') }}">  
126 @csrf  
127 <button type="submit" class="btn btn-danger btn-sm">Logout</button>  
128 </form>  
129 </li>
```

➤ Maka akan muncul tombol logout pada header, namun login dulu:



➤ Halaman tampilan ketika sudah login, dengan tombol Logout pada bagian header



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



B. Implementasi *Authorization* di Laravel

Authorization merupakan proses setelah *authentication* berhasil dilakukan (dalam kata lain, kita berhasil login ke sistem). *Authorization* berkenaan dengan hak akses pengguna dalam menggunakan sistem. *Authorization* memberikan/memastikan hak akses (ijin akses) kita, sesuai dengan aturan (role) yang ada di sistem. *Authorization* sangat penting untuk membatasi akses pengguna sesuai dengan peruntukannya.

Contoh ketika kita mengakses LMS dengan akun (*username* dan *password*) yang bertipe **Mahasiswa**. Saat berhasil melakukan *authentication*, maka hak akses kita juga akan diberikan selayaknya mahasiswa. Seperti melihat kursus (course), melihat materi, men-download file materi, mengerjakan/meng-upload tugas, mengikuti ujian, dll. Kita tidak akan diberikan hak akses oleh sistem untuk membuat materi, membuat soal ujian, membuat tugas, memberikan nilai tugas karena hak akses tersebut masuk ke ranah akun tipe **Dosen/Pengajar**.

Selain menyediakan layanan otentikasi bawaan, Laravel juga menyediakan cara sederhana untuk mengotorisasi tindakan pengguna terhadap sumber daya tertentu. Misalnya, meskipun pengguna diautentikasi, mereka mungkin tidak berwenang untuk memperbarui atau menghapus model Eloquent atau rekaman database tertentu yang dikelola oleh aplikasi Anda. Fitur otorisasi Laravel menyediakan cara yang mudah dan terorganisir untuk mengelola jenis pemeriksaan otorisasi ini.

Praktikum 2 – Implementasi *Authorization* di Laravel dengan Middleware

Kita akan menerapkan *authorization* pada project Laravel dengan menggunakan Middleware sebagai pengecekan akses. Langkah-langkah yang kita kerjakan sebagai berikut:

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

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

/**
 * Mendapatkan nama role
 */
public function getRoleName(): string
{
    return $this->level->level_nama;
}

/**
 * Cek apakah user memiliki role tertentu
 */
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

```
php artisan make:middleware AuthorizeUser
```

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

```
1  <?php
2  namespace App\Http\Middleware;
3
4  use Closure;
5  use Illuminate\Http\Request;
6  use Symfony\Component\HttpFoundation\Response;
7
8  class AuthorizeUser
9  {
10     /**
11      * Handle an incoming request.
12      *
13      * @param  \Closure(\Illuminate\Http\Request): (\Symfony\Component\HttpFoundation\Response)
14      */
15     public function handle(Request $request, Closure $next, $role = ''): Response
16     {
17         $user = $request->user(); // ambil data user yg login
18         // fungsi user() diambil dari UserModel.php
19         if($user->hasRole($role)){ // cek apakah user punya role yg diinginkan
20             return $next($request);
21         }
22         // jika tidak punya role, maka tampilkan error 403
23         abort(403, 'Forbidden. Kamu tidak punya akses ke halaman ini');
24     }
25 }
```

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 yg kita buat
    'auth.basic' => \Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class,
    'auth.session' => \Illuminate\Session\Middleware\AuthenticateSession::class,
```

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

level_id	level_kode	level_nama	created_at	updated_at	deleted_at
1	ADM	Administrator	NULL	NULL	NULL
2	MNG	Manager	NULL	NULL	NULL
3	STF	Staf	NULL	2024-08-16 01:49:20	NULL
4	KSR	Kasir	NULL	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

```
Route::middleware(['auth'])->group(function(){ // artinya semua route di dalam group ini harus login dulu
    Route::get('/', [WelcomeController::class, 'index']);
    // route Level

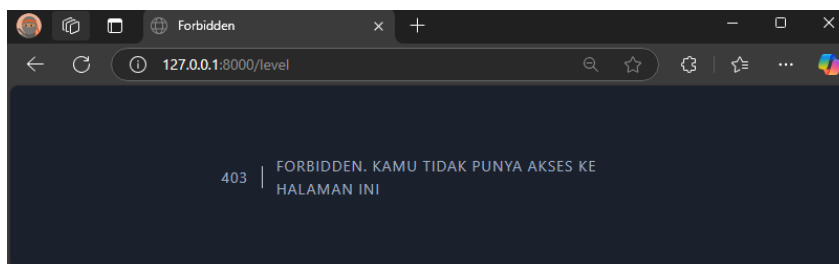
    // artinya semua route di dalam group ini harus punya role ADM (Administrator)
    Route::middleware(['authorize:ADM'])->group(function(){
        Route::get('/level', [LevelController::class, 'index']);
        Route::post('/level/list', [LevelController::class, 'list']); // untuk list json datatables
        Route::get('/level/create', [LevelController::class, 'create']);
        Route::post('/level', [LevelController::class, 'store']);
        Route::get('/level/{id}/edit', [LevelController::class, 'edit']); // untuk tampilkan form edit
        Route::put('/level/{id}', [LevelController::class, 'update']); // untuk proses update data
        Route::delete('/level/{id}', [LevelController::class, 'destroy']); // untuk proses hapus data
    });

    // route Kategori
```

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

Maka tampilan akan seperti ini kalau dari level manager :



Tugas 2 – Implementasi Authoriization :

1. Apa yang kalian pahami pada praktikum 2 ini?
 - Pada praktikum ini, kita menerapkan **authorization** di Laravel dengan menggunakan **Middleware** untuk mengecek apakah pengguna memiliki akses ke halaman tertentu berdasarkan **role**-nya. Middleware ini akan memfilter request dan memastikan hanya pengguna dengan level yang sesuai yang bisa mengakses route tertentu.
2. Amati dan jelaskan tiap tahapan yang kalian kerjakan, dan jabarkan dalam laporan
 - **Modifikasi UserModel.php**
 - Menambahkan method `getRoleName()` untuk mendapatkan nama role.
 - Menambahkan method `hasRole($role)` untuk mengecek apakah user memiliki role tertentu.



- Membuat Middleware AuthorizeUser.php
 - Menjalankan perintah:
php artisan make:middleware AuthorizeUser
 - Middleware ini mengecek apakah user yang sedang login memiliki role yang sesuai. Jika iya, request diteruskan. Jika tidak, ditampilkan error **403 (Forbidden)**.
- Mendaftarkan Middleware di Kernel.php
 - Middleware AuthorizeUser didaftarkan dengan alias 'authorize', sehingga bisa digunakan di routes dengan authorize:ROLE_KODE
- Menyesuaikan Data di Database (m_level)
 - Tabel m_level berisi daftar role seperti **Administrator (ADM)**, **Manager (MNG)**, **Staff (STF)**, dan **Pelanggan (CUS)**.
- Menggunakan Middleware di web.php
 - Semua route dalam group ini hanya bisa diakses oleh user dengan **level_kode = ADM (Administrator)**.

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

C. Multi-Level Authorization di Laravel

Bagaimana seandainya jika terdapat level/group/role satu dengan yang lain memiliki hak akses yang sama. Contoh sederhana, user level Admin dan Manager bisa sama-sama mengakses menu Barang pada aplikasi yang kita buat. Maka tidak mungkin kalau kita buat route untuk masing-masing level user. Hal ini akan memakan banyak waktu, dan proses yang lama.

```
// artinya semua route di dalam group ini harus punya role ADM (Administrator)
Route::middleware(['authorize:ADM'])->group(function(){
    Route::get('/barang',[BarangController::class,'index']);
    Route::post('/barang/list',[BarangController::class,'list']);
    Route::get('/barang/create_ajax',[BarangController::class,'create_ajax']); // ajax form create
    Route::post('/barang_ajax',[BarangController::class,'store_ajax']); // ajax store
    Route::get('/barang/{id}/edit_ajax',[BarangController::class,'edit_ajax']); // ajax form edit
    Route::put('/barang/{id}/update_ajax',[BarangController::class,'update_ajax']); // ajax update
    Route::get('/barang/{id}/delete_ajax',[BarangController::class,'confirm_ajax']); // ajax form confirm delete
    Route::delete('/barang/{id}/delete_ajax',[BarangController::class,'delete_ajax']); // ajax delete
});

// artinya semua route di dalam group ini harus punya role MNG (Manager)
Route::middleware(['authorize:MNG'])->group(function(){
    Route::get('/barang',[BarangController::class,'index']);
    Route::post('/barang/list',[BarangController::class,'list']);
    Route::get('/barang/create_ajax',[BarangController::class,'create_ajax']); // ajax form create
    Route::post('/barang_ajax',[BarangController::class,'store_ajax']); // ajax store
    Route::get('/barang/{id}/edit_ajax',[BarangController::class,'edit_ajax']); // ajax form edit
    Route::put('/barang/{id}/update_ajax',[BarangController::class,'update_ajax']); // ajax update
    Route::get('/barang/{id}/delete_ajax',[BarangController::class,'confirm_ajax']); // ajax form confirm delete
    Route::delete('/barang/{id}/delete_ajax',[BarangController::class,'delete_ajax']); // ajax delete
});
```

Hal ini jadi kendala ketika kita mau mengganti hak akses, maka kita akan mengganti sebagian besar route yang sudah kita tulis. Untuk itu, kita perlu mengelola middleware agar bisa mendukung penambahan hak akses secara dinamis.



Praktikum 3 – Implementasi Multi-Level Authorizaton di Laravel dengan Middleware

Kita akan menerapkan multi-level authorization pada project Laravel dengan menggunakan Middleware sebagai pengecekan akses. Langkah-langkah yang kita kerjakan sebagai berikut:

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

```
/**
 * Mendapatkan nama role
 */
public function getRoleName(): string
{
    return $this->level->level_nama;
}

/**
 * Cek apakah user memiliki role tertentu
 */
public function hasRole($role): bool
{
    return $this->level->level_kode == $role;
}

/**
 * Mendapatkan kode role
 */
public function getRole()
{
    return $this->level->level_kode;
}
```

2. Selanjutnya, Kita modifikasi middleware `AuthorizeUser.php` dengan kode berikut

```
1  <?php
2  namespace App\Http\Middleware;
3
4  use Closure;
5  use Illuminate\Http\Request;
6  use Symfony\Component\HttpFoundation\Response;
7
8  class AuthorizeUser
9  {
10
11      /**
12       * Handle an incoming request.
13       *
14       * @param  \Closure(\Illuminate\Http\Request): (\Symfony\Component\HttpFoundation\Response)
15       */
16      public function handle(Request $request, Closure $next, ... $roles): Response
17      {
18          $user_role = $request->user()->getRole(); // ambil data level_kode dari user yg login
19          if(in_array($user_role, $roles)){ // cek apakah level_kode user ada di dalam array roles
20              return $next($request); // jika ada, maka lanjutkan request
21          }
22          // jika tidak punya role, maka tampilkan error 403
23          abort(403, 'Forbidden. Kamu tidak punya akses ke halaman ini');
24      }
25  }
```

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

```
// artinya semua route di dalam group ini harus punya role ADM (Administrator) dan MNG (Manager)
Route::middleware(['authorize:ADM,MNG'])->group(function(){
    Route::get('/barang', [BarangController::class, 'index']);
    Route::post('/barang/list', [BarangController::class, 'list']);
    Route::get('/barang/create_ajax', [BarangController::class, 'create_ajax']); // ajax form create
    Route::post('/barang_ajax', [BarangController::class, 'store_ajax']); // ajax store
    Route::get('/barang/{id}/edit_ajax', [BarangController::class, 'edit_ajax']); // ajax form edit
    Route::put('/barang/{id}/update_ajax', [BarangController::class, 'update_ajax']); // ajax update
    Route::get('/barang/{id}/delete_ajax', [BarangController::class, 'confirm_ajax']); // ajax form confirm
    Route::delete('/barang/{id}/delete_ajax', [BarangController::class, 'delete_ajax']); // ajax delete
});
```



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

➤ Mengakses Data Barang dari Admin

The screenshot shows two browser windows. The left window is the 'AdminLTE' login page with fields for 'admin' and a password, and a 'Sign In' button. The right window shows the 'Data Barang' page for an Admin user. The page has a sidebar with a 'Data Barang' menu item highlighted. The main content area displays a table of goods with columns: ID, Kode Barang, Nama Barang, Nama Kategori, Harga Beli, Harga Jual, and Aksi. The table contains two rows of data.

ID	Kode Barang	Nama Barang	Nama Kategori	Harga Beli	Harga Jual	Aksi
1	BRG001	Biskuit	Makanan	3000	5000	Detail Edit Hapus
2	BRG002	Roti	Makanan	5000	7000	Detail Edit Hapus

➤ Mengakses Data Barang dari Manager

The screenshot shows two browser windows. The left window is the 'AdminLTE' login page with fields for 'manager' and a password, and a 'Sign In' button. The right window shows the 'Data Barang' page for a Manager user. The sidebar menu is different from the Admin user, and the 'Data Barang' menu item is highlighted. The main content area displays a table of goods with columns: ID, Kode Barang, Nama Barang, Nama Kategori, Harga Beli, Harga Jual, and Aksi. The table contains five rows of data.

ID	Kode Barang	Nama Barang	Nama Kategori	Harga Beli	Harga Jual	Aksi
1	BRG001	Biskuit	Makanan	3000	5000	Detail Edit Hapus
2	BRG002	Roti	Makanan	5000	7000	Detail Edit Hapus
3	BRG003	Teh Botol	Minuman	2000	3000	Detail Edit Hapus
4	BRG004	Kopi Instan	Minuman	3000	4000	Detail Edit Hapus
5	BRG005	Mouse Wireless	Elektronik	100000	150000	Detail Edit Hapus

➤ Mengakses Data Barang dari Staff

The screenshot shows two browser windows. The left window is the 'AdminLTE' login page with fields for 'staff' and a password, and a 'Sign In' button. The right window shows a 'Forbidden' page with the message: '403 | FORBIDDEN, KAMU TIDAK PUNYA AKSES KE HALAMAN INI'.

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

➤ Berikut Kode Program routes/web.php



```
routes > @ web.php -
1
2 </php
3
4 use App\Http\Controllers\UserController;
5 use App\Http\Controllers\SupplierController;
6 use App\Http\Controllers\BarangController;
7 use App\Http\Controllers\KategoriController;
8 use App\Http\Controllers\LevelController;
9 use App\Http\Controllers\WelcomeController;
10 use App\Http\Controllers\AuthController;
11 use Illuminate\Support\Facades\Route;
12
13 Route::pattern('id', pattern: '[0-9]+'); // Pastikan parameter (id) hanya berupa angka
14
15 // Auto otentikasi
16 Route::get('login', action: [AuthController::class, 'login'])->name('login');
17 Route::post('login', action: [AuthController::class, 'postlogin']);
18 Route::post('logout', action: [AuthController::class, 'logout'])->middleware('auth')->name('logout');
19
20 // Semua rute di bawah ini hanya bisa diakses jika sudah login
21 Route::middleware('auth')->group(callback: function () { void {
22     Route::get('/', action: [WelcomeController::class, 'index']);
23
24     Route::middleware('authorize:ADM')->group(callback: function () { void {
25         Route::group(attributes: ['prefix' => 'user'], routes: function () { void {
26             Route::get('/', action: [UserController::class, 'index']); // Menampilkan halaman awal user
27             Route::post('list', action: [UserController::class, 'list']); // Menampilkan data user dalam bentuk json untuk datatables
28             Route::get('create', action: [UserController::class, 'create']); // Menampilkan halaman form tambah user
29             Route::post('/', action: [UserController::class, 'store']); // Menyimpan data user baru
30
31             // Create menggunakan AJAX
32             Route::get('create_ajax', action: [UserController::class, 'create_ajax']); // Menampilkan halaman form tambah user Ajax
33             Route::post('ajax', action: [UserController::class, 'store_ajax']); // Menyimpan data user baru Ajax
34             Route::get('show', action: [UserController::class, 'show']); // Menampilkan detail user
35             Route::get('edit', action: [UserController::class, 'edit']); // Menampilkan halaman form edit user
36             Route::put('update', action: [UserController::class, 'update']); // Menyimpan perubahan data user
37
38             // Edit menggunakan AJAX
39             Route::get('edit_ajax', action: [UserController::class, 'edit_ajax']); // Menampilkan halaman form edit user Ajax
40             Route::put('update_ajax', action: [UserController::class, 'update_ajax']); // Menyimpan perubahan data user Ajax
41
42             // Delete menggunakan AJAX
43             Route::get('delete_ajax', action: [UserController::class, 'confirm_ajax']); // Untuk tampilan form confirm delete user Ajax
44             Route::delete('delete_ajax', action: [UserController::class, 'delete_ajax']); // Untuk hapus data user Ajax
45             Route::delete('destroy', action: [UserController::class, 'destroy']); // Menghapus data user
46         });
47     });
48
49 // artinya semua route di dalam group ini harus punya role ADM (Administrator)
50 Route::middleware('authorize:ADM')->group(callback: function () { void {
51     Route::group(attributes: ['prefix' => 'level'], routes: function () { void {
52         Route::get('/', action: [LevelController::class, 'index']); // Menampilkan halaman awal level
53         Route::post('list', action: [LevelController::class, 'list']); // Menampilkan data level dalam bentuk json untuk datatables
54         Route::get('create', action: [LevelController::class, 'create']); // Menampilkan halaman form tambah level
55         Route::post('/', action: [LevelController::class, 'store']); // Menyimpan data level baru
56
57         // Create menggunakan AJAX
58         Route::get('create_ajax', action: [LevelController::class, 'create_ajax']); // Menampilkan halaman form tambah level Ajax
59         Route::post('ajax', action: [LevelController::class, 'store_ajax']); // Menyimpan data level baru Ajax
60         Route::get('show', action: [LevelController::class, 'show']); // Menampilkan detail level
61         Route::get('edit', action: [LevelController::class, 'edit']); // Menampilkan halaman form edit level
62         Route::put('update', action: [LevelController::class, 'update']); // Menyimpan perubahan data level
63
64         // Edit menggunakan AJAX
65         Route::get('edit_ajax', action: [LevelController::class, 'edit_ajax']); // Menampilkan halaman form edit level Ajax
66         Route::put('update_ajax', action: [LevelController::class, 'update_ajax']); // Menyimpan perubahan data level Ajax
67
68         // Delete menggunakan AJAX
69         Route::get('delete_ajax', action: [LevelController::class, 'confirm_ajax']); // Untuk tampilan form confirm delete level Ajax
70         Route::delete('delete_ajax', action: [LevelController::class, 'delete_ajax']); // Untuk hapus data level Ajax
71         Route::delete('destroy', action: [LevelController::class, 'destroy']); // Menghapus data level
72     });
73 });
74
75 // artinya semua route di dalam group ini harus punya role ADM (Administrator) dan MNG (Manager)
76 Route::middleware('authorize:ADM,MNG')->group(callback: function () { void {
77     Route::group(attributes: ['prefix' => 'kategori'], routes: function () { void {
78         Route::get('/', action: [KategoriController::class, 'index']); // Menampilkan halaman awal kategori
79         Route::post('list', action: [KategoriController::class, 'list']); // Menampilkan data kategori dalam bentuk json untuk datatables
80         Route::get('create', action: [KategoriController::class, 'create']); // Menampilkan halaman form tambah kategori
81         Route::post('/', action: [KategoriController::class, 'store']); // Menyimpan data kategori baru
82
83         // Create menggunakan AJAX
84         Route::get('create_ajax', action: [KategoriController::class, 'create_ajax']); // Menampilkan halaman form tambah kategori Ajax
85         Route::post('ajax', action: [KategoriController::class, 'store_ajax']); // Menyimpan data kategori baru Ajax
86         Route::get('show', action: [KategoriController::class, 'show']); // Menampilkan detail kategori
87         Route::get('edit', action: [KategoriController::class, 'edit']); // Menampilkan halaman form edit kategori
88         Route::put('update', action: [KategoriController::class, 'update']); // Menyimpan perubahan data kategori
89
90         // Edit menggunakan AJAX
91         Route::get('edit_ajax', action: [KategoriController::class, 'edit_ajax']); // Menampilkan halaman form edit kategori Ajax
92         Route::put('update_ajax', action: [KategoriController::class, 'update_ajax']); // Menyimpan perubahan data kategori Ajax
93
94         // Delete menggunakan AJAX
95         Route::get('delete_ajax', action: [KategoriController::class, 'confirm_ajax']); // Menampilkan form confirm delete kategori Ajax
96         Route::delete('delete_ajax', action: [KategoriController::class, 'delete_ajax']); // Untuk hapus data kategori Ajax
97         Route::delete('destroy', action: [KategoriController::class, 'destroy']); // Menghapus data kategori
98     });
99 });
100
101 // Staff hanya bisa melihat data barang
102 Route::middleware('authorize:ADM,MNG,STF')->group(callback: function () { void {
103     Route::group(attributes: ['prefix' => 'barang'], routes: function () { void {
104         Route::get('/', action: [BarangController::class, 'index']); // Menampilkan daftar barang
105         Route::post('list', action: [BarangController::class, 'list']); // Menampilkan data barang dalam bentuk JSON untuk datatables
106         Route::get('show', action: [BarangController::class, 'show']); // Menampilkan detail barang
107     });
108 });
109
110 // ADM & MNG bisa menambah, mengedit, dan menghapus barang
111 Route::middleware('authorize:ADM,MNG')->group(callback: function () { void {
112     Route::group(attributes: ['prefix' => 'barang'], routes: function () { void {
113         Route::get('create', action: [BarangController::class, 'create']); // Form tambah barang
114         Route::post('/', action: [BarangController::class, 'store']); // Simpan barang baru
115
116         // Create menggunakan AJAX
117         Route::get('create_ajax', action: [BarangController::class, 'create_ajax']); // Form tambah barang AJAX
118         Route::post('ajax', action: [BarangController::class, 'store_ajax']); // Simpan barang baru AJAX
119         Route::get('edit', action: [BarangController::class, 'edit']); // Form edit barang
120         Route::put('update', action: [BarangController::class, 'update']); // Simpan perubahan barang
121
122         // Edit menggunakan AJAX
123         Route::get('edit_ajax', action: [BarangController::class, 'edit_ajax']); // Form edit barang AJAX
124         Route::put('update_ajax', action: [BarangController::class, 'update_ajax']); // Simpan perubahan barang AJAX
125
126         // Delete menggunakan AJAX
127         Route::get('delete_ajax', action: [BarangController::class, 'confirm_ajax']); // Form konfirmasi hapus barang AJAX
128         Route::delete('delete_ajax', action: [BarangController::class, 'delete_ajax']); // Hapus barang AJAX
129         Route::delete('destroy', action: [BarangController::class, 'destroy']); // Hapus barang
130     });
131 });
132
133 // artinya semua route di dalam group ini harus punya role ADM (Administrator) dan MNG (Manager)
134 Route::middleware('authorize:ADM,MNG')->group(callback: function () { void {
135     Route::group(attributes: ['prefix' => 'supplier'], routes: function () { void {
136         Route::get('/', action: [SupplierController::class, 'index']);
137         Route::post('list', action: [SupplierController::class, 'list']); // Menampilkan data supplier dalam bentuk JSON untuk datatables
138         Route::get('create', action: [SupplierController::class, 'create']); // Menampilkan halaman form tambah supplier
139         Route::post('/', action: [SupplierController::class, 'store']); // Menyimpan data supplier baru
140
141         // Create menggunakan AJAX
142         Route::get('create_ajax', action: [SupplierController::class, 'create_ajax']); // Menampilkan halaman form tambah supplier Ajax
143         Route::post('ajax', action: [SupplierController::class, 'store_ajax']); // Menyimpan data supplier baru Ajax
144         Route::get('show', action: [SupplierController::class, 'show']); // Menampilkan detail supplier
145         Route::get('edit', action: [SupplierController::class, 'edit']); // Menampilkan halaman form edit supplier
146         Route::put('update', action: [SupplierController::class, 'update']); // Menyimpan perubahan data supplier
147
148         // Edit menggunakan AJAX
149         Route::get('edit_ajax', action: [SupplierController::class, 'edit_ajax']); // Menampilkan halaman form edit supplier Ajax
150         Route::put('update_ajax', action: [SupplierController::class, 'update_ajax']); // Menyimpan perubahan data supplier Ajax
151
152         // Delete menggunakan AJAX
153         Route::get('delete_ajax', action: [SupplierController::class, 'confirm_ajax']); // Menampilkan form confirm delete supplier Ajax
154         Route::delete('delete_ajax', action: [SupplierController::class, 'delete_ajax']); // Hapus data supplier Ajax
155         Route::delete('destroy', action: [SupplierController::class, 'destroy']); // Menghapus data supplier
156     });
157 });
158
159 //
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163

```



➤ Berikut Hak Akses tiap level nya :

Menu	Level Admin	Level Manager	Staff
Data User			
Data Level			
Data Kategori			
Data Barang			
Data Supplier			

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

Tugas 4 – Implementasi Form Registrasi :

1. Silahkan implementasikan form untuk registrasi user.

➤ AuthController.php :

```
42 public function register(): Factory|View
43 {
44     return view('auth.register');
45 }
46
```



```
47 public function postRegister(Request $request): Redirector|RedirectRe
48 {
49     $validatedData = $request->validate(rules: [
50         'username' => 'required|string|unique:m_user,username',
51         'nama' => 'required|string|max:255',
52         'password' => 'required|string|min:6|confirmed',
53         'level_id' => 'required|exists:m_level,level_id',
54     ]);
55
56     $user = UserModel::create(attributes: [
57         'username' => $validatedData['username'],
58         'nama' => $validatedData['nama'],
59         'password' => Hash::make(value: $validatedData['password']),
60         'level_id' => $validatedData['level_id'],
61         'created_at' => now(),
62         'updated_at' => now(),
63     ]);
64
65     Auth::login(user: $user);
66
67     return redirect(to: '/');
68 }
```

➤ Routes/web.php :

```
14 // Ruta otentikasi
15 Route::get(uri: 'login', action: [AuthController::class, 'login'])->name(name: 'login');
16 Route::post(uri: 'login', action: [AuthController::class, 'postlogin']);
17 Route::get(uri: '/register', action: [AuthController::class, 'register']);
18 Route::post(uri: '/register', action: [AuthController::class, 'postRegister']);
19 Route::post(uri: '/logout', action: [AuthController::class, 'logout'])->middleware(middleware: 'auth')->name(name: 'logout');
```

➤ Views/auth/register.blade.php

```
resources > views > auth > register.blade.php
1 @extends('layouts.app')
2
3 @section('content')
4 <div class="container">
5     <div class="row justify-content-center">
6         <div class="col-md-6">
7             <div class="card">
8                 <div class="card-header">Registrasi</div>
9
10                <div class="card-body">
11                    <form method="POST" action="{{ url(path: '/register') }}">
12                        @csrf
13
14                        <div class="mb-3">
15                            <label for="username" class="form-label">Username</label>
16                            <input type="text" class="form-control" id="username" name="username" required>
17                        </div>
18
19                        <div class="mb-3">
20                            <label for="nama" class="form-label">Nama Lengkap</label>
21                            <input type="text" class="form-control" id="nama" name="nama" required>
22                        </div>
23
24                        <div class="mb-3">
25                            <label for="password" class="form-label">Password</label>
26                            <input type="password" class="form-control" id="password" name="password" required>
27                        </div>
28
29                        <div class="mb-3">
30                            <label for="password_confirmation" class="form-label">Konfirmasi Password</label>
31                            <input type="password" class="form-control" id="password_confirmation" name="password_confirmation" required>
32                        </div>
33
34                        <div class="mb-3">
35                            <label for="level_id" class="form-label">Level</label>
36                            <select class="form-control" id="level_id" name="level_id">
37                                <option value="1">Admin</option>
38                                <option value="2">Manager</option>
39                                <option value="3">Staff/Kasir</option>
40                                <option value="4">Pelanggan</option>
41                            </select>
42                        </div>
43
44                        <button type="submit" class="btn btn-primary">Daftar</button>
45                        <a href="{{ url(path: '/login') }}" class="btn btn-link">Sudah punya akun?</a>
46                    </form>
47                </div>
48            </div>
49        </div>
50    </div>
51</div>
```

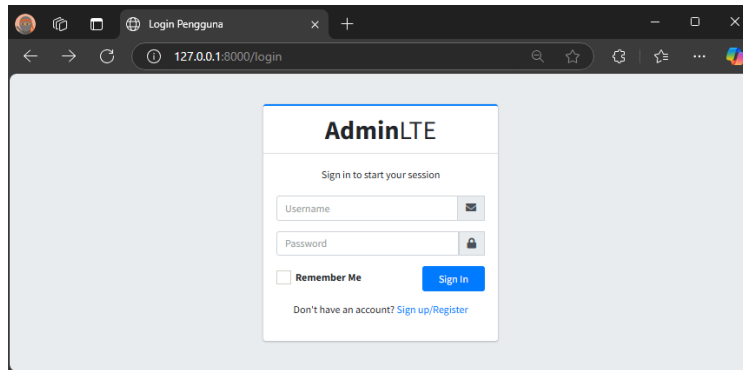
➤ Views/auth/login.blade.php

```
55 <div class="row">
56     <div class="col-8">
57         <div class="icheck-primary">
58             <input type="checkbox" id="remember">
59             <label for="remember">Remember Me</label>
60         </div>
61     </div>
62     <div class="col-4">
63         <button type="submit" class="btn btn-primary btn-block">Sign In</button>
64     </div>
65 </div>
66
67 <div class="text-center mt-3">
68     <p>Don't have an account? <a href="{{ url(path: 'register') }}" class="text-primary">Sign up</a></p>
69 </div>
70 </form>
71 </div>
```

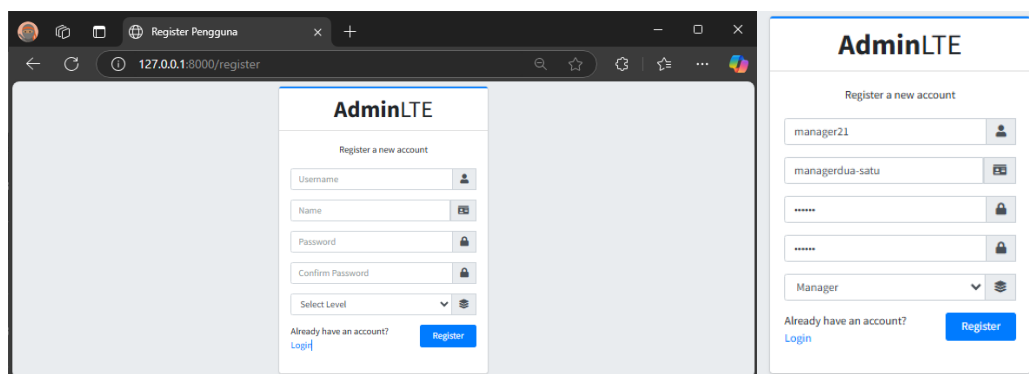


2. Screenshot hasil yang kalian kerjakan

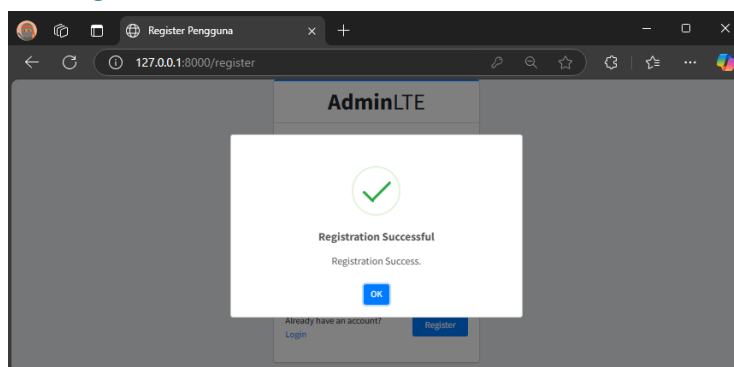
➤ Menambahkan tombol Sign Up/Register di view login :



➤ Halaman Register :



➤ Register berhasil :



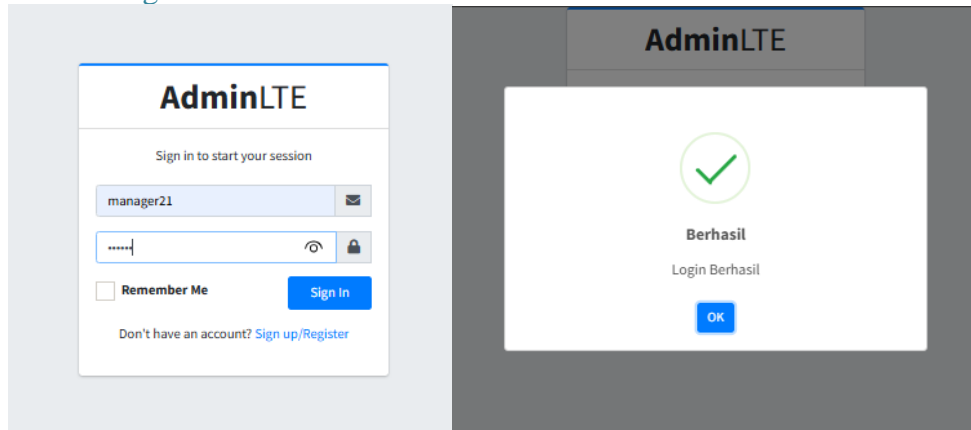
➤ Cek isi tabel m_user apakah sudah terisi :

	user_id	username	nama	password	level_id	created_at	updated_at
	1	admin	Admin	\$2y\$12\$SRH94NY5JqU3n6X33CTjuCA9c6V9ywf6c8k5FjGz...	1	NULL	2025-03-27 17:07:49
	2	manager	Manager	\$2y\$12\$NBMPRE8ztPBzB6oiAahXuu1fhePBzC3kY6BIS0mJa...	2	NULL	NULL
	3	staff	StaffKasir	\$2y\$12\$kyE/4aqOsm95pntNhuusv6B/QRmWWKachnBqT...	3	NULL	NULL
	7	customer-1	Pelanggan Pertama	\$2y\$12\$UJB68J8nttchl.guOMHok5 SvkskBJOstP74vKuhh8...	4	NULL	2025-03-05 03:51:54
	8	manager_dua	Manager 2	\$2y\$12\$ouPpe yf61f6ulleXQbbuZioY9DGLHczbgd7g0t...	2	2025-03-06 06:16:38	2025-03-06 06:16:38
	9	manager22	Manager Dua Dua	\$2y\$12\$Nhm1y7Kcpg0nZgfaOrq7bdySvW1u0k6VbR4Q...	2	2025-03-08 16:02:27	2025-03-08 16:02:27
	10	manager33	Manager Tiga Tiga	\$2y\$12\$JL3E KT7dvgSPnpZUV0PH.FHm2B0B3glpGemmFa0...	2	2025-03-08 16:21:24	2025-03-08 16:21:24
	11	manager55	Manager55	\$2y\$12\$KgpvGi6Hn56KD Udmjnu3k7xE QXwwiD2Vcbjg...	2	2025-03-08 16:34:13	2025-03-08 16:34:13
	15	admin1	admin1	\$2y\$12\$ECcjKMzKxeD02vUetrbGeZLha8ZzmIPPv37evil...	1	2025-03-27 16:39:51	2025-03-27 16:39:51
	16	manager21	managerdua-satu	\$2y\$12\$uTDZUPJH5KYLmqg9KCKIM elluqzT0wRB0VawBqJDJs...	2	2025-03-28 06:13:46	2025-03-28 06:13:46



KEMENTERIAN PENDIDIKAN, KEBUDAYAAN, RISET, DAN TEKNOLOGI
POLITEKNIK NEGERI MALANG
JURUSAN TEKNOLOGI INFORMASI
Jl. Soekarno Hatta No. 9, Jatimulyo, Lowokwaru, Malang 65141
Telp. (0341) 404424 – 404425, Fax (0341) 404420
<http://www.polinema.ac.id>

➤ Cek login user baru:



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