

PRAKTIKUM PEMROGRAMAN WEB LANJUT

TUGAS PERTEMUAN 7

Authentication dan Authorization di Laravel

DISUSUN OLEH :

GILANG PURNOMO

NIM:2341720042



PROGRAM STUDI D-IV TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

MARET 2025

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

```
Minggu 7 > PWL_POS > app > Models > UserModel.php > ...
1  <?php
2
3  namespace App\Models;
4
5  use Illuminate\Database\Eloquent\Factories\HasFactory;
6  use Illuminate\Database\Eloquent\Model;
7  use Illuminate\Database\Eloquent\Relations\BelongsTo;
8  use Illuminate\Foundation\Auth\User as Authenticatable;
9
10 class UserModel extends Authenticatable
11 {
12     use HasFactory;
13
14     protected $table = 'm_user';
15     protected $primaryKey = 'user_id';
16
17     protected $fillable = ['level_id', 'username', 'nama', 'password', 'create_at', 'update_at'];
18
19     protected $hidden = ['password'];
20
21     protected $casts = ['password' => 'hashed'];
22
23     public function level(): BelongsTo
24     {
25         return $this->belongsTo(LevelModel::class, 'level_id', 'level_id');
26     }
27 }
28
```

3. Selanjutnya kita buat [AuthController.php](#) untuk memproses login yang akan kita lakukan

```

Minggu 7 > PWL_POS > app > Http > Controllers > AuthController.php > PHP Intelephense > AuthController
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  {
10     public function login()
11     {
12         if(Auth::check()) {
13             return redirect('/');
14         }
15         return view('auth.login');
16     }
17
18     public function postLogin(Request $request)
19     {
20         if($request -> ajax() || $request -> wantsJson()) {
21             $credentials = $request->only('username', 'password');
22
23             if(Auth::attempt($credentials)) {
24                 return response()->json([
25                     'status' => true,
26                     'message' => 'Login Berhasil',
27                     'redirect' => url('/')
28                 ]);
29             }
30
31             return response()->json([
32                 'statud' => false,
33                 'message' => 'Login Gagal'
34             ]);
35         }
36         return redirect('login');
37     }
38
39     public function logout(Request $request)
40     {
41         Auth::logout();
42
43         $request->session()->invalidate();
44         $request->session()->regenerateToken();
45         return redirect('/login');
46     }
47
48 }

```

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)

Minggu 7 > PwL_POS > resources > views > auth > login.blade.php > html > body.hold-transition.login-page

```
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">
6     <title>Login Pengguna</title>
7     <!-- Google Font: Source Sans Pro -->
8     <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Source+Sans+Pro:300,400,400i,700&display=fallback">
9     <!-- Font Awesome -->
10    <link rel="stylesheet" href="{{ asset('adminlte/plugins/fontawesome-free/css/all.min.css') }}">
11    <!-- iCheck bootstrap -->
12    <link rel="stylesheet" href="{{ asset('adminlte/plugins/iCheck-bootstrap/iCheck-bootstrap.min.css') }}">
13    <!-- SweetAlert2 -->
14    <link rel="stylesheet" href="{{ asset('adminlte/plugins/sweetalert2-theme-bootstrap-4/bootstrap4.min.css') }}">
15    <!-- Theme style -->
16    <link rel="stylesheet" href="{{ asset('adminlte/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"><a href="{{ url('/') }}" class="h1"><b>AdminLTE</b></a></div>
23             <div class="card-body">
24                 <p class="login-box-msg">Sign in to start your session</p>
25                 <form action="{{ url('login') }}" method="POST" id="form-login">
26                     @csrf
27                     <div class="input-group mb-3">
28                         <input type="text" id="username" name="username" class="form-control" placeholder="Username">
29                         <div class="input-group-append">
30                             <div class="input-group-text">
31                                 <span class="fas fa-envelope"></span>
32                             </div>
33                         </div>
34                         <small id="error-username" class="error-text text-danger"></small>
35                     </div>
36                     <div class="input-group mb-3">
37                         <input type="password" id="password" name="password" class="form-control" placeholder="Password">
38                         <div class="input-group-append">
39                             <div class="input-group-text">
40                                 <span class="fas fa-lock"></span>
41                             </div>
42                         </div>
43                         <small id="error-password" class="error-text text-danger"></small>
44                     </div>
45                     <div class="row">
46                         <div class="col-8">
47                             <div class="icheck-primary">
48                                 <input type="checkbox" id="remember"><label for="remember">Remember Me</label>
49                             </div>
50                         </div>
51                         <!-- /.col -->
52                         <div class="col-4">
53                             <button type="submit" class="btn btn-primary btn-block">Sign In</button>
54                         </div>
55                         <!-- /.col -->
56                     </div>
57                 </form>
58             </div>
59         </div>
60     </div>
61     <!-- /.card -->
62 </div>
63 <!-- /.login-box -->
64
65 <!-- jQuery -->
66 <script src="{{ asset('adminlte/plugins/jquery/jquery.min.js') }}"></script>
67 <!-- Bootstrap 4 -->
68 <script src="{{ asset('adminlte/plugins/bootstrap/js/bootstrap.bundle.min.js') }}"></script>
69 <!-- jquery-validation -->
70 <script src="{{ asset('adminlte/plugins/jquery-validation/jquery.validate.min.js') }}"></script>
71 <script src="{{ asset('adminlte/plugins/jquery-validation/additional-methods.min.js') }}"></script>
72 <!-- SweetAlert2 -->
73 <script src="{{ asset('adminlte/plugins/sweetalert2/sweetalert2.min.js') }}"></script>
74 <!-- AdminLTE App -->
75 <script src="{{ asset('adminlte/dist/js/adminlte.min.js') }}"></script>
76
77 <script>
78     $.ajaxSetup({
79         headers: {
80             'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('content')
81         }
82     });
83
84     $(document).ready(function() {
85         $("#form-login").validate({
86             rules: {
87                 username: {required: true, minlength: 4, maxlength: 20},
88                 password: {required: true, minlength: 5, maxlength: 20}
89             },
90         });
91     });
92 </script>
```

```

90 submitHandler: function(form) { // ketika valid, maka bagian yg akan dijalankan
91     $.ajax({
92         url: form.action,
93         type: form.method,
94         data: $(form).serialize(),
95         success: function(response) {
96             if(response.status){ // jika sukses
97                 Swal.fire({
98                     icon: 'success',
99                     title: 'Berhasil',
100                     text: response.message,
101                 }).then(function() {
102                     window.location = response.redirect;
103                 });
104             }else{ // jika error
105                 $('.error-text').text('');
106                 $.each(response.msgField, function(prefix, val) {
107                     $('#error-'+prefix).text(val[0]);
108                 });
109                 Swal.fire({
110                     icon: 'error',
111                     title: 'Terjadi Kesalahan',
112                     text: response.message
113                 });
114             }
115         }
116     });
117     return false;
118 },
119 errorElement: 'span',
120 errorPlacement: function (error, element) {
121     error.addClass('invalid-feedback');
122     element.closest('.input-group').append(error);
123 },
124 highlight: function (element, errorClass, validClass) {
125     $(element).addClass('is-invalid');
126 },
127 unhighlight: function (element, errorClass, validClass) {
128     $(element).removeClass('is-invalid');
129 }
130 });
131 });
132 </script>

```

```

133 </body>
134 </html>

```

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

```

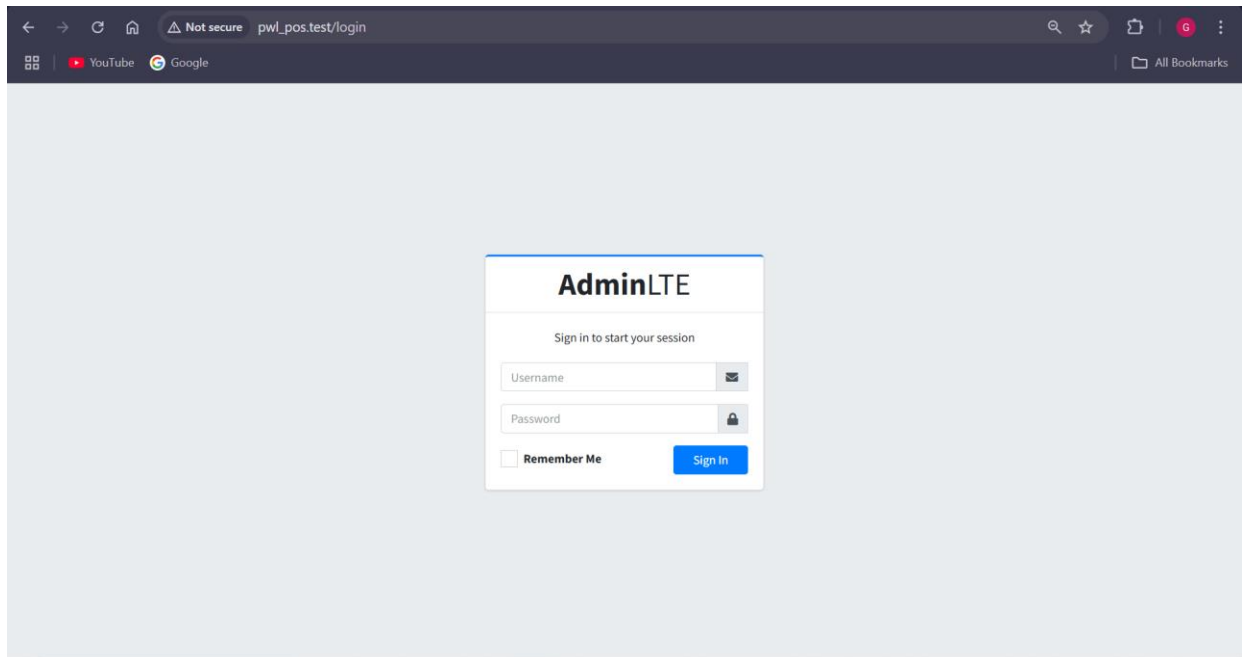
Route::pattern('id', '[0-9]+');

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 () {
    //
});

```

6. Ketika kita coba mengakses halaman localhost/PWL_POS/public makan akan tampil halaman awal untuk login ke aplikasi

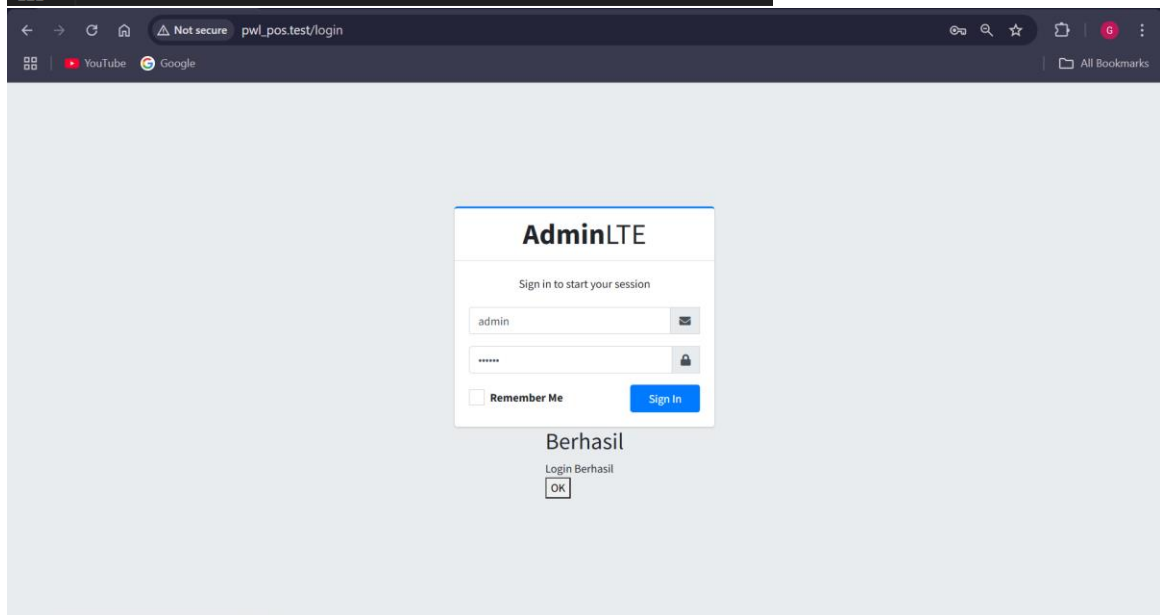


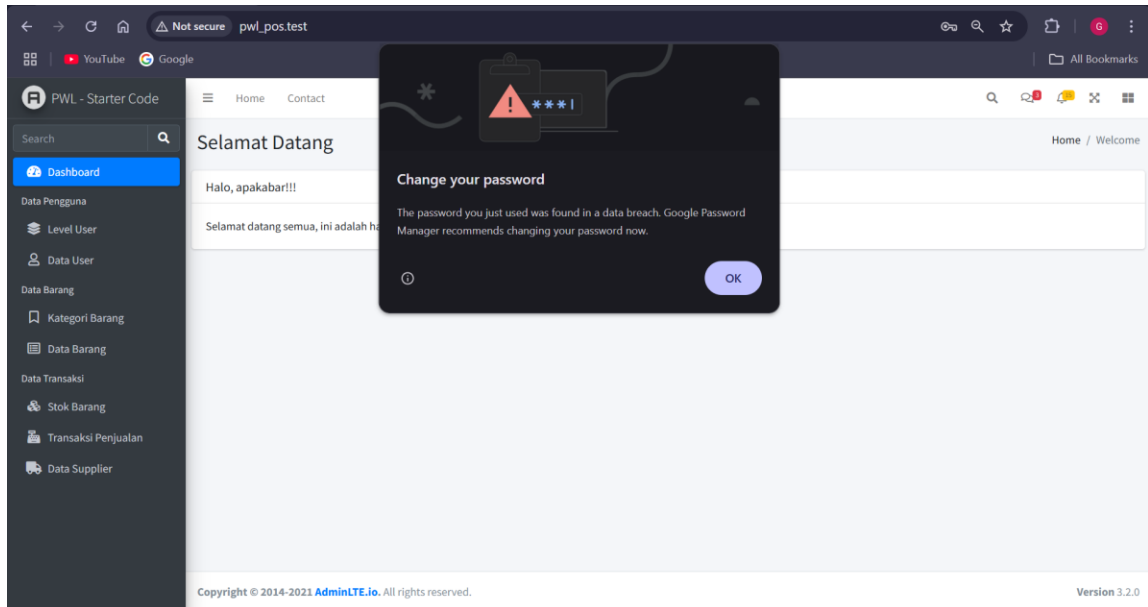
Tugas 1 – Implementasi Authentication :

1. Silahkan implementasikan proses login pada project kalian masing-masing

Jawab :

```
114 Route::get('login', [AuthController::class, 'login'])->name('login');
115 Route::post('login', [AuthController::class, 'postLogin']);
116 Route::get('logout', [AuthController::class, 'logout'])->middleware('auth');
117
118 Route::middleware(['auth'])->group(function () {
119     Route::get('/', [WelcomeController::class, 'index']);
120 });
121
```





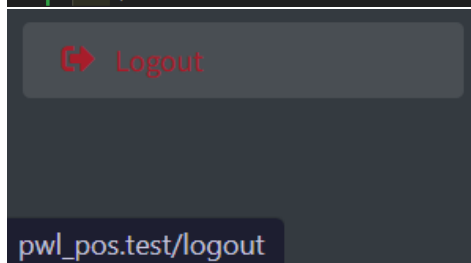
2. Silahkan implementasi proses logout pada halaman web yang kalian buat

Jawab :

```

70 <nav class="mt-5">
71 <ul class="nav nav-pills nav-sidebar flex-column">
72 <li class="nav-item">
73 <a href="/logout" class="nav-link text-danger">
74 <i class="nav-icon fas fa-sign-out-alt"></i>
75 <p>Logout</p>
76 </a>
77 </li>
78 </ul>
79 </nav>

```



3. Amati dan jelaskan tiap tahapan yang kalian kerjakan, dan jabarkan dalam laporan
 Jawab : Jadi proses login ini pertama mengakses /login untuk menampilkan halaman login, isi username dan password, ketika menekan sign in maka data dikirim melalui postLogin() dengan menggunakan ajax. Jika valid Laravel melakukan Auth::attempt(). Jika sukses, akan diarahkan ke halaman utama (/), jika gagal, muncul pesan error. Untuk logout ketika klik logout, Laravel menjalankan Auth::logout(), menghapus sesi, dan mengarahkan ke halaman login.

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

Jawab :

https://github.com/Gilangp/Pemrograman_Web_Lanjut/commit/2c58e1f8bcdecf6cac3066791533dfd579a93a3c

Praktikum 2 – Implementasi Authorizaton di Laravel dengan Middleware

1. Kita modifikasi [UserModel.php](#) dengan menambahkan kode berikut

```
23     public function level(): BelongsTo
24     {
25         return $this->belongsTo(LevelModel::class, 'level_id', 'level_id');
26     }
27
28     // Mendapatkan nama role
29     public function getRoleName(): string
30     {
31         return $this->level->level_nama;
32     }
33
34     // Cek apakah user memiliki role tertentu
35     public function hasRole($role): bool
36     {
37         return $this->level->level_kode == $role;
38     }
39 }
```

2. Kemudian kita buat middleware dengan nama [AuthorizeUser.php](#). Kita bisa buat middleware dengan mengetikkan perintah pada terminal/CMD

```
PS C:\laragon\www\Web_Lanjut\Minggu 7\PWL_POS> php artisan make:middleware AuthorizeUser
INFO Middleware [C:\laragon\www\Web_Lanjut\Minggu 7\PWL_POS\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

```
Minggu 7 > PWL_POS > app > Http > Middleware > AuthorizeUser.php > PHP Intelephense > AuthorizeUser > handle
1  <?php
2
3  namespace App\Http\Middleware;
4
5  use Closure;
6  use Illuminate\Http\Request;
7  use Symfony\Component\HttpFoundation\Response;
8
9  class AuthorizeUser
10 {
11     /**
12      * Handle an incoming request.
13      *
14      * @param  \Closure(\Illuminate\Http\Request): (\Symfony\Component\HttpFoundation\Response)  $next
15      */
16     public function handle(Request $request, Closure $next, $role = ''): Response
17     {
18         $user = $request->user();
19
20         if($user->hasRole($role)) {
21             return $next($request);
22         }
23         abort(403, 'Fobidden. Kamu tidak punya akses ke halaman ini');
24     }
25 }
26
```

4. 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\AuthenticateWithBasicAuth::class,
```

5. Sekarang kita perhatikan tabel [m_level](#) yang menjadi tabel untuk menyimpan level/group/role dari user ada

level_id	level_code	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

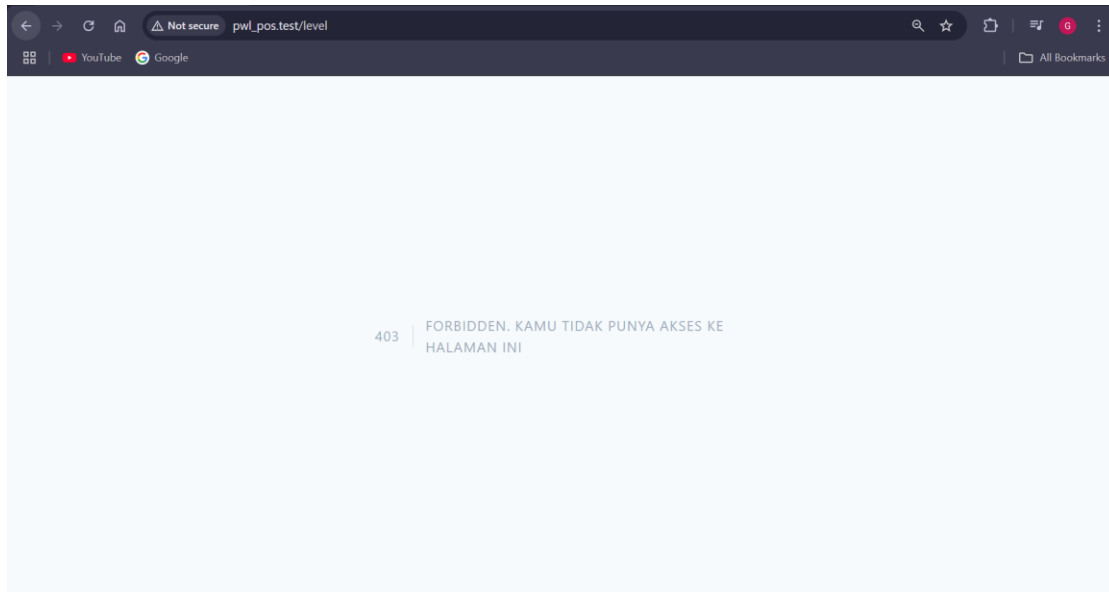
```

118 Route::middleware(['auth'])->group(function () {
119     Route::get('/', [WelcomeController::class, 'index']);
120
121     // semua route harus punya role ADM
122     Route::middleware(['authorize:ADM'])->group(function () {
123         Route::group(['prefix' => 'level'], function () {
124             Route::get('/', [LevelController::class, 'index']);
125             Route::post('/list', [LevelController::class, 'list']);
126             Route::get('/create', [LevelController::class, 'create']);
127             Route::get('/create_ajax', [LevelController::class, 'create_ajax']);
128             Route::post('/ajax', [LevelController::class, 'store_ajax']);
129             Route::post('/', [LevelController::class, 'store']);
130             Route::get('/{id}', [LevelController::class, 'show']);
131             Route::get('/{id}/show_ajax', [LevelController::class, 'show_ajax']);
132             Route::get('/{id}/edit', [LevelController::class, 'edit']);
133             Route::put('/{id}', [LevelController::class, 'update']);
134             Route::get('/{id}/edit_ajax', [LevelController::class, 'edit_ajax']);
135             Route::put('/{id}/update_ajax', [LevelController::class, 'update_ajax']);
136             Route::get('/{id}/delete_ajax', [LevelController::class, 'confirm_ajax']);
137             Route::delete('/{id}/delete_ajax', [LevelController::class, 'delete_ajax']);
138             Route::delete('/{id}', [LevelController::class, 'destroy']);
139         });
140     });
141 });

```

Pada kode yang ditandai merah, terdapat `authorize:ADM`. Kode `ADM` adalah nilai dari `level_code` 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



Jadi ketika login dengan menggunakan user staff maka tidak akan bisa mengakses menu level.

Tugas 2 – Implementasi Authoriization :

1. Apa yang kalian pahami pada praktikum 2 ini?
Jawab : praktikum 2 ini untuk otorisasi user menggunakan Laravel yang dimana agar semua user memiliki hak akses tertentu.
2. Amati dan jelaskan tiap tahapan yang kalian kerjakan, dan jabarkan dalam laporan
Jawab : Model UserModel untuk menangani data pengguna dan memiliki fungsi untuk mengecek role user. Middleware AuthorizeUser digunakan untuk memfilter akses berdasarkan role user. Route menggunakan middleware authorize:ADM agar hanya admin yang dapat mengakses halaman tersebut.
3. Submit kode untuk impementasi Authorization pada repository github kalian
Jawab :
https://github.com/Gilangp/Pemrograman_Web_Lanjut/commit/1f36457d19647ffcf63a3e6b7c21b40d7787a8d

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()`

```

28 // Mendapatkan nama role
29 public function getRoleName(): string
30 {
31     return $this->level->level_nama;
32 }
33
34 // Cek apakah user memiliki role tertentu
35 public function hasRole($role): bool
36 {
37     return $this->level->level_kode == $role;
38 }
39
40 // untuk mendapatkan kode role
41 public function getRole()
42 {
43     return $this->level->level_kode;
44 }
45 }

```

2. Selanjutnya, Kita modifikasi middleware [AuthorizeUser.php](#) dengan kode berikut

```

Minggu 7 > PWL_POS > app > Http > Middleware > AuthorizeUser.php > ...
1 <?php
2
3 namespace App\Http\Middleware;
4
5 use Closure;
6 use Illuminate\Http\Request;
7 use Symfony\Component\HttpFoundation\Response;
8
9 class AuthorizeUser
10 {
11     /**
12      * Handle an incoming request.
13      *
14      * @param \Closure(\Illuminate\Http\Request): (\Symfony\Component\HttpFoundation\Response) $next
15      */
16     public function handle(Request $request, Closure $next, ... $roles): Response
17     {
18         $user_role = $request->user()->getRole();
19
20         if(in_array($user_role, $roles)) {
21             return $next($request);
22         }
23         abort(403, 'Forbidden. Kamu tidak punya akses ke halaman ini');
24     }
25 }
26

```

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

```

141
142 Route::middleware(['authorize:ADM,MNG'])->group(function(){
143     Route::group(['prefix' => 'barang'], function () {
144         Route::get('/', [BarangController::class, 'index']);
145         Route::post('/list', [BarangController::class, 'list']);
146         Route::get('/create', [BarangController::class, 'create']);
147         Route::get('/create_ajax', [BarangController::class, 'create_ajax']);
148         Route::post('/ajax', [BarangController::class, 'store_ajax']);
149         Route::post('/', [BarangController::class, 'store']);
150         Route::get('/{id}', [BarangController::class, 'show']);
151         Route::get('/{id}/show_ajax', [BarangController::class, 'show_ajax']);
152         Route::get('/{id}/edit', [BarangController::class, 'edit']);
153         Route::put('/{id}', [BarangController::class, 'update']);
154         Route::get('/{id}/edit_ajax', [BarangController::class, 'edit_ajax']);
155         Route::put('/{id}/update_ajax', [BarangController::class, 'update_ajax']);
156         Route::get('/{id}/delete_ajax', [BarangController::class, 'confirm_ajax']);
157         Route::delete('/{id}/delete_ajax', [BarangController::class, 'delete_ajax']);
158         Route::delete('/{id}', [BarangController::class, 'destroy']);
159     });
160 });

```

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
Jawab : pada praktikum 3 ini yang dibagikan usermodel untuk mendapatkan role user/pengguna, lalu bagian middleware untuk memeriksa hak akses user. Pada web.php/routing ini menerapkan middleware tersebut agar hanya pengguna dengan role tertentu yang bisa mengakses menu-menu tertentu.
3. Implementasikan multi-level authorization untuk semua Level/Jenis User dan Menumenu yang sesuai dengan Level/Jenis User

Jawab :

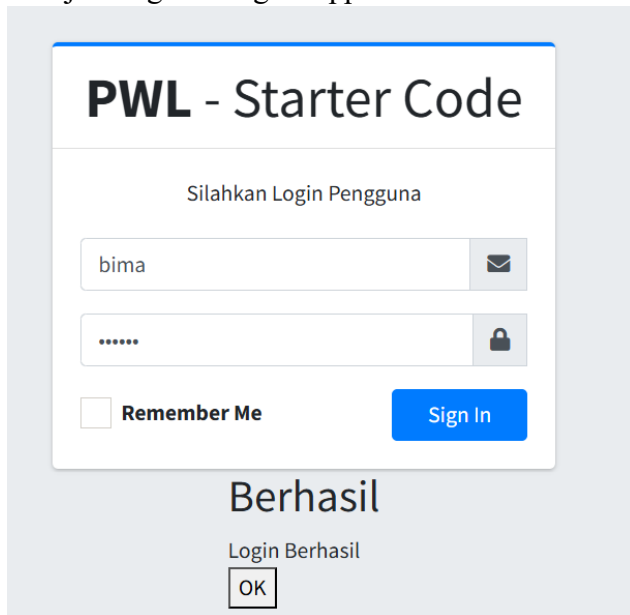
```
145 | // Hak akses untuk role ADM
146 | Route::middleware(['authorize:ADM'])->group(function(){
147 |     Route::group(['prefix' => 'level'], function () {
148 |         Route::get('/', [LevelController::class, 'index']);
149 |         Route::post('/list', [LevelController::class, 'list']);
150 |         Route::get('/create', [LevelController::class, 'create']);
151 |         Route::get('/create_ajax', [LevelController::class, 'create_ajax']);
152 |         Route::post('/ajax', [LevelController::class, 'store_ajax']);
153 |         Route::post('/', [LevelController::class, 'store']);
154 |         Route::get('/{id}', [LevelController::class, 'show']);
155 |         Route::get('/{id}/show_ajax', [LevelController::class, 'show_ajax']);
156 |         Route::get('/{id}/edit', [LevelController::class, 'edit']);
157 |         Route::put('/{id}', [LevelController::class, 'update']);
158 |         Route::get('/{id}/edit_ajax', [LevelController::class, 'edit_ajax']);
159 |         Route::put('/{id}/update_ajax', [LevelController::class, 'update_ajax']);
160 |         Route::get('/{id}/delete_ajax', [LevelController::class, 'confirm_ajax']);
161 |         Route::delete('/{id}/delete_ajax', [LevelController::class, 'delete_ajax']);
162 |         Route::delete('/{id}', [LevelController::class, 'destroy']);
163 |     });
164 | });
165 |
166 | // Hak akses hanya admin dan manager
167 | Route::middleware(['authorize:ADM,MNG'])->group(function(){
168 |     Route::group(['prefix' => 'barang'], function () {
169 |         Route::get('/', [BarangController::class, 'index']);
170 |         Route::post('/list', [BarangController::class, 'list']);
171 |         Route::get('/create', [BarangController::class, 'create']);
172 |         Route::get('/create_ajax', [BarangController::class, 'create_ajax']);
173 |         Route::post('/ajax', [BarangController::class, 'store_ajax']);
174 |         Route::post('/', [BarangController::class, 'store']);
175 |         Route::get('/{id}', [BarangController::class, 'show']);
176 |         Route::get('/{id}/show_ajax', [BarangController::class, 'show_ajax']);
177 |         Route::get('/{id}/edit', [BarangController::class, 'edit']);
178 |         Route::put('/{id}', [BarangController::class, 'update']);
179 |         Route::get('/{id}/edit_ajax', [BarangController::class, 'edit_ajax']);
180 |         Route::put('/{id}/update_ajax', [BarangController::class, 'update_ajax']);
181 |         Route::get('/{id}/delete_ajax', [BarangController::class, 'confirm_ajax']);
182 |         Route::delete('/{id}/delete_ajax', [BarangController::class, 'delete_ajax']);
183 |         Route::delete('/{id}', [BarangController::class, 'destroy']);
184 |     });
185 | });
```

```

186         Route::group(['prefix' => 'user'], function () {
187             Route::get('/', [UserController::class, 'index']);
188             Route::post('/list', [UserController::class, 'list']);
189             Route::get('/create', [UserController::class, 'create']);
190             Route::post('/', [UserController::class, 'store']);
191             Route::get('/create_ajax', [UserController::class, 'create_ajax']);
192             Route::post('/ajax', [UserController::class, 'store_ajax']);
193             Route::get('/{id}', [UserController::class, 'show']);
194             Route::get('/{id}/show_ajax', [UserController::class, 'show_ajax']);
195             Route::get('/{id}/edit', [UserController::class, 'edit']);
196             Route::put('/{id}', [UserController::class, 'update']);
197             Route::get('/{id}/edit_ajax', [UserController::class, 'edit_ajax']);
198             Route::put('/{id}/update_ajax', [UserController::class, 'update_ajax']);
199             Route::get('/{id}/delete_ajax', [UserController::class, 'confirm_ajax']);
200             Route::delete('/{id}/delete_ajax', [UserController::class, 'delete_ajax']);
201             Route::delete('/{id}', [UserController::class, 'destroy']);
202         });
203     });
204
205     // Hak akses hanya admin, manager, dan staff
206     Route::middleware(['authorize:ADM,MNG,STF'])->group(function(){
207         Route::group(['prefix' => 'stok'], function () {
208             Route::get('/', [StokController::class, 'index']);
209             Route::post('/list', [StokController::class, 'list']);
210             Route::get('/create', [StokController::class, 'create']);
211             Route::post('/', [StokController::class, 'store']);
212             Route::get('/{id}', [StokController::class, 'show']);
213             Route::get('/{id}/edit', [StokController::class, 'edit']);
214             Route::put('/{id}', [StokController::class, 'update']);
215             Route::delete('/{id}', [StokController::class, 'destroy']);
216         });
217     });
218
219     // Hak akses hanya admin, manager, staff, dan customer
220     Route::middleware(['authorize:ADM,MNG,STF,CUS'])->group(function(){
221         Route::group(['prefix' => 'penjualan'], function () {
222             Route::get('/', [PenjualanController::class, 'index']);
223             Route::post('/list', [PenjualanController::class, 'list']);
224             Route::get('/create', [PenjualanController::class, 'create']);
225             Route::post('/', [PenjualanController::class, 'store']);
226             Route::get('/{id}', [PenjualanController::class, 'show']);
227             Route::get('/{id}/edit', [PenjualanController::class, 'edit']);
228             Route::put('/{id}', [PenjualanController::class, 'update']);
229             Route::delete('/{id}', [PenjualanController::class, 'destroy']);
230         });
231     });
232
233     // Hak akses hanya admin, manager, staff, dan supplier
234     Route::middleware(['authorize:ADM,MNG,STF,SUP'])->group(function(){
235         Route::group(['prefix' => 'supplier'], function () {
236             Route::get('/', [SupplierController::class, 'index']);
237             Route::post('/list', [SupplierController::class, 'list']);
238             Route::get('/create', [SupplierController::class, 'create']);
239             Route::get('/create_ajax', [SupplierController::class, 'create_ajax']);
240             Route::post('/ajax', [SupplierController::class, 'store_ajax']);
241             Route::post('/', [SupplierController::class, 'store']);
242             Route::get('/{id}', [SupplierController::class, 'show']);
243             Route::get('/{id}/show', [SupplierController::class, 'show']);
244             Route::get('/{id}/show_ajax', [SupplierController::class, 'show_ajax']);
245             Route::get('/{id}/edit', [SupplierController::class, 'edit']);
246             Route::put('/{id}', [SupplierController::class, 'update']);
247             Route::get('/{id}/edit_ajax', [SupplierController::class, 'edit_ajax']);
248             Route::put('/{id}/update_ajax', [SupplierController::class, 'update_ajax']);
249             Route::get('/{id}/delete_ajax', [SupplierController::class, 'confirm_ajax']);
250             Route::delete('/{id}/delete_ajax', [SupplierController::class, 'delete_ajax']);
251             Route::delete('/{id}', [SupplierController::class, 'destroy']);
252         });
253     });
254
255     // Hak akses untuk semua level
256     Route::middleware(['authorize:ADM,MNG,STF,CUS,SUP'])->group(function(){
257         Route::group(['prefix' => 'kategori'], function () {
258             Route::get('/', [KetegoriController::class, 'index']);
259             Route::post('/list', [KetegoriController::class, 'list']);
260             Route::get('/create', [KetegoriController::class, 'create']);
261             Route::get('/create_ajax', [KetegoriController::class, 'create_ajax']);
262             Route::post('/ajax', [KetegoriController::class, 'store_ajax']);
263             Route::post('/', [KetegoriController::class, 'store']);
264             Route::get('/{id}', [KetegoriController::class, 'show']);
265             Route::get('/{id}/show_ajax', [KetegoriController::class, 'show_ajax']);
266             Route::get('/{id}/edit', [KetegoriController::class, 'edit']);
267             Route::put('/{id}', [KetegoriController::class, 'update']);
268             Route::get('/{id}/edit_ajax', [KetegoriController::class, 'edit_ajax']);
269             Route::put('/{id}/update_ajax', [KetegoriController::class, 'update_ajax']);
270             Route::get('/{id}/delete_ajax', [KetegoriController::class, 'confirm_ajax']);
271             Route::delete('/{id}/delete_ajax', [KetegoriController::class, 'delete_ajax']);
272             Route::delete('/{id}', [KetegoriController::class, 'destroy']);
273         });
274     });
275
276 });
277

```

Jadi jika login sebagai supplier



PWL - Starter Code

Silahkan Login Pengguna

bima

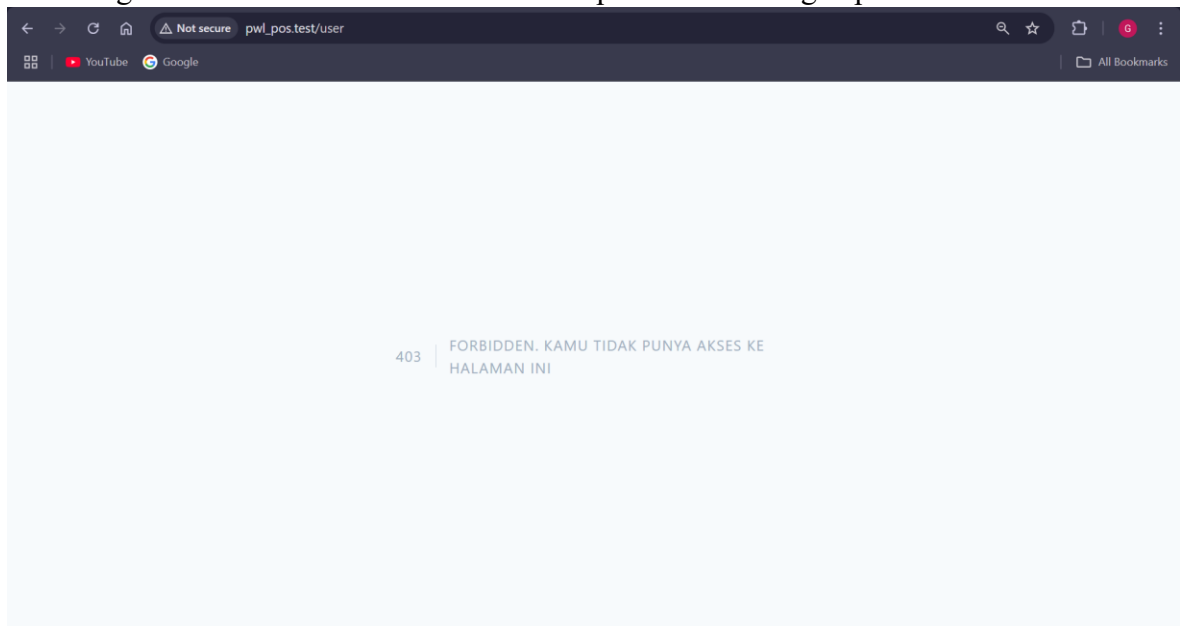
.....

☐ Remember Me

Berhasil

Login Berhasil

dan mengakses menu Data User maka menampilkan error dengan pesan tersebut.



karena tidak memiliki hak akses untuk menu Data User.

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

Jawab :

https://github.com/Gilangp/Pemrograman_Web_Lanjut/commit/d7513ec6a52889bfd112117842b89d9315611bc0

Tugas 4 – Implementasi Form Registrasi :

1. Silahkan implementasikan form untuk registrasi user.

Route

```
138 Route::get('/register', [AuthController::class, 'register'])->name('register');
139 Route::post('/register', [AuthController::class, 'storeRegister']);
```

AuthController

```
52 // Register
53 public function register()
54 {
55     if (Auth::check()) {
56         return redirect('/');
57     }
58
59     $levels = LevelModel::all();
60     return view('auth.register', ['levels' => $levels]);
61 }
62
63 public function storeRegister(Request $request)
64 {
65     if ($request->ajax() || $request->wantsJson()) {
66         $validator = Validator::make($request->all(), [
67             'level_id' => 'required|exists:m_level,level_id',
68             'username' => 'required|unique:m_user,username',
69             'nama' => 'required|string|max:100',
70             'password' => 'required|min:6|confirmed',
71         ]);
72
73         if ($validator->fails()) {
74             return response()->json([
75                 'status' => false,
76                 'message' => 'Validasi gagal!',
77                 'msgField' => $validator->errors()
78             ]);
79         }
80
81         UserModel::create([
82             'level_id' => $request->level_id,
83             'username' => $request->username,
84             'nama' => $request->nama,
85             'password' => Hash::make($request->password),
86         ]);
87
88         return response()->json([
89             'status' => true,
90             'message' => 'Registrasi berhasil!',
91             'redirect' => route('login')
92         ]);
93     }
94
95     return redirect()->route('register')->with('success', 'Registrasi berhasil!');
96 }
```

Form registrasi

```

Minggu 7 > PWL_POS > resources > views > auth > register.blade.php > html > body.hold-transition.login-page > div.login-box > div.card.card-outline.card-primary > div.card-body >
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">
6 <title>Register Pengguna</title>
7 <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Source+Sans+Pro:300,400,400i,700&display=fallback">
8 <link rel="stylesheet" href="{{ asset('adminlte/plugins/fontawesome-free/css/all.min.css') }}">
9 <link rel="stylesheet" href="{{ asset('adminlte/plugins/ichack-bootstrap/ichack-bootstrap.min.css') }}">
10 <link rel="stylesheet" href="{{ asset('adminlte/plugins/sweetalert2-theme-bootstrap-4/bootstrap4.min.css') }}">
11 <link rel="stylesheet" href="{{ asset('adminlte/dist/css/adminlte.min.css') }}">
12 </head>
13 <body class="hold-transition login-page">
14 <div class="login-box">
15 <div class="card card-outline card-primary">
16 <div class="card-header text-center">
17 <a href="{{ url('/') }}" class="h1"><b>PWL</b> - Starter Code</a>
18 </div>
19 <div class="card-body">
20 <p class="login-box-msg">Registrasi Pengguna Baru</p>
21 <form action="{{ url('/register') }}" method="POST" id="form-register">
22 @csrf
23
24 <div class="form-group mb-3">
25 <select name="level_id" class="form-control">
26 <option value=""-- Pilih Level --</option>
27 @foreach($levels as $level)
28 <option value="{{ $level->level_id }}">{{ $level->level_nama }} ({{ $level->level_kode }})</option>
29 @endforeach
30 </select>
31 <small id="error-level_id" class="error-text text-danger"></small>
32 </div>
33
34 <div class="input-group mb-3">
35 <input type="text" name="username" class="form-control" placeholder="Username">
36 <div class="input-group-append">
37 <div class="input-group-text"><span class="fas fa-user"></span></div>
38 </div>
39 <small id="error-username" class="error-text text-danger"></small>
40 </div>
41
42 <div class="input-group mb-3">
43 <input type="text" name="nama" class="form-control" placeholder="Nama Lengkap">
44 <div class="input-group-append">
45 <div class="input-group-text"><span class="fas fa-id-card"></span></div>
46 </div>
47 <small id="error-nama" class="error-text text-danger"></small>
48 </div>
49
50 <div class="input-group mb-3">
51 <input type="password" name="password" class="form-control" placeholder="Password">
52 <div class="input-group-append">
53 <div class="input-group-text"><span class="fas fa-lock"></span></div>
54 </div>
55 <small id="error-password" class="error-text text-danger"></small>
56 </div>
57
58 <div class="input-group mb-3">
59 <input type="password" name="password_confirmation" class="form-control" placeholder="Konfirmasi Password">
60 <div class="input-group-append">
61 <div class="input-group-text"><span class="fas fa-lock"></span></div>
62 </div>
63 <small id="error-password_confirmation" class="error-text text-danger"></small>
64 </div>
65
66 <div class="row">
67 <div class="col-12">
68 <button type="submit" class="btn btn-primary btn-block">Daftar</button>
69 </div>
70 </div>
71 </form>
72 <p class="mt-3 text-center">
73 Sudah punya akun? <a href="{{ url('/login') }}">Login di sini</a>
74 </p>
75 </div>
76 </div>
77 </div>
78
79 <!-- Scripts -->
80 <script src="{{ asset('adminlte/plugins/jquery/jquery.min.js') }}"></script>
81 <script src="{{ asset('adminlte/plugins/bootstrap/js/bootstrap.bundle.min.js') }}"></script>
82 <script src="{{ asset('adminlte/plugins/jquery-validation/jquery.validate.min.js') }}"></script>
83 <script src="{{ asset('adminlte/plugins/jquery-validation/additional-methods.min.js') }}"></script>
84 <script src="{{ asset('adminlte/plugins/sweetalert2/sweetalert2.min.js') }}"></script>
85 <script src="{{ asset('adminlte/dist/js/adminlte.min.js') }}"></script>
86

```

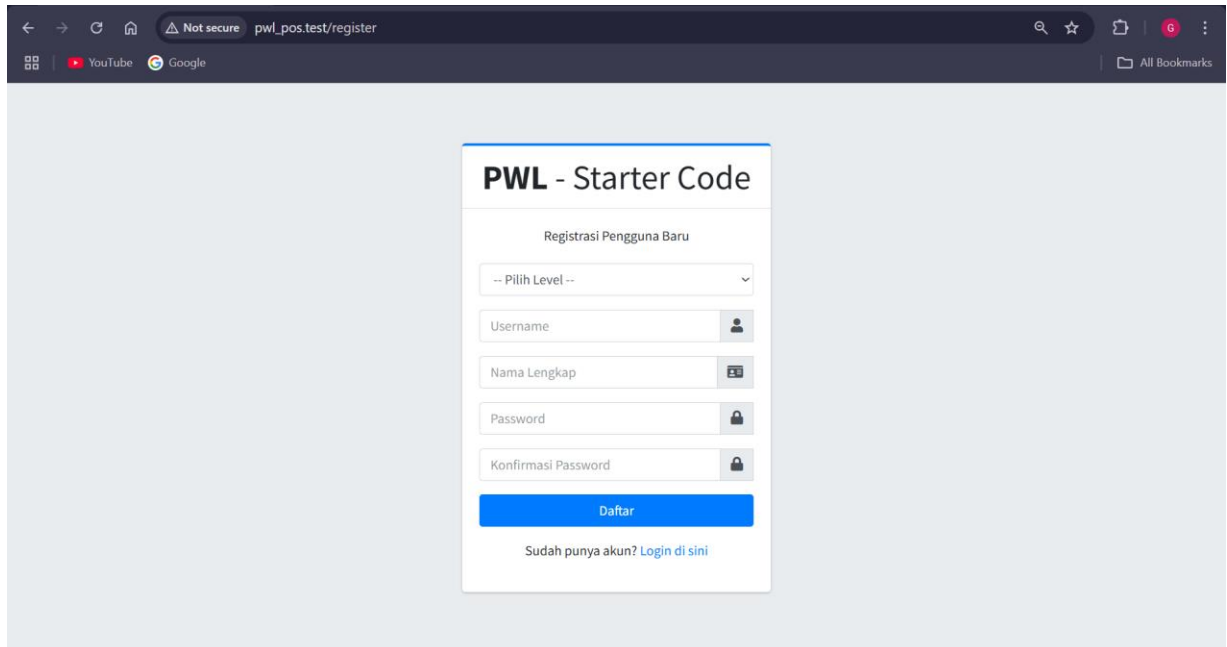


```

87 <script>
88 $(document).ready(function() {
89     $('#form-register').validate({
90         rules: {
91             level_id: { required: true },
92             username: { required: true, minlength: 4, maxlength: 20 },
93             nama: { required: true },
94             password: { required: true, minlength: 6, maxlength: 20 },
95             password_confirmation: { required: true, equalTo: '[name="password"]' },
96         },
97         submitHandler: function(form) {
98             $.ajax({
99                 url: form.action,
100                 type: 'POST',
101                 data: $(form).serialize(),
102                 success: function(response) {
103                     if(response.status) {
104                         Swal.fire({
105                             icon: 'success',
106                             title: 'Berhasil',
107                             text: response.message
108                         }).then(function() {
109                             window.location.href = response.redirect || "{{ url('/login') }}"
110                         });
111                     } else {
112                         $('#error-text').text('');
113                         if (response.msgField) {
114                             $.each(response.msgField, function(prefix, val) {
115                                 $('#error-' + prefix).text(val[0]);
116                             });
117                         }
118                         Swal.fire({
119                             icon: 'error',
120                             title: 'Gagal',
121                             text: response.message
122                         });
123                     }
124                 }
125             });
126             return false;
127         },
128         errorElement: 'span',
129         errorPlacement: function (error, element) {
130             error.addClass('invalid-feedback');
131             element.closest('.input-group, .form-group').append(error);
132             element.closest('.input-group, .form-group').append(error);
133         },
134         highlight: function (element) {
135             $(element).addClass('is-invalid');
136         },
137         unhighlight: function (element) {
138             $(element).removeClass('is-invalid');
139         }
140     });
141 </script>
142 </body>
143 </html>

```

2. Screenshot hasil yang kalian kerjakan



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

Jawab :

https://github.com/Gilangp/Pemrograman_Web_Lanjut/commit/dc6238766359a6e5685176af36dde0c2e425c109