

**Pemrograman Web Lanjut**  
**Jobsheet 5 - Blade View, Web Templating(AdminLTE), Datatables**

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



**Nama : Athallah Ayudya Paramesti**  
**NIM : 2341760061**  
**Program Studi : D-IV Sistem Informasi Bisnis**

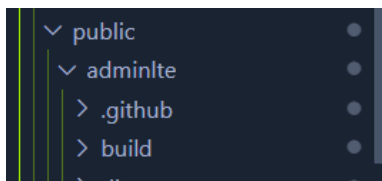
**Jurusan Teknologi Informasi**  
**Politeknik Negeri Malang**  
**2025**

## Praktikum 1 - Layouting AdminLTE

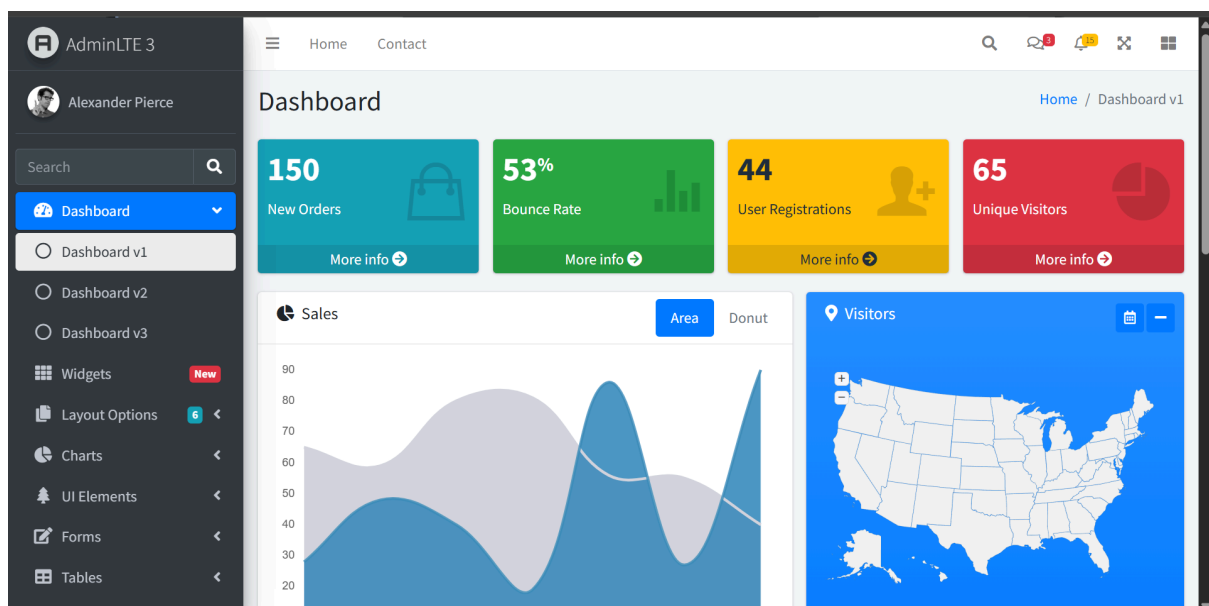
1. Download AdminLTE v3.2.0 yang rilis pada 8 Feb 2022



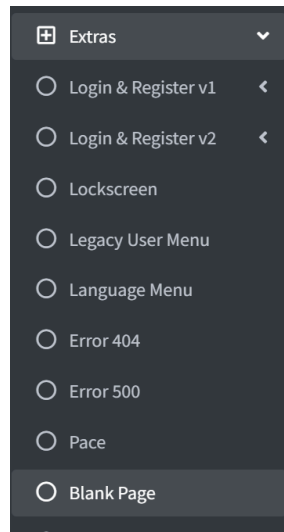
2. Setelah berhasil download, ekstrak file yang sudah di download ke folder project `PWL_POS/public`, kemudian kita *rename* folder cukup menjadi `adminlte`



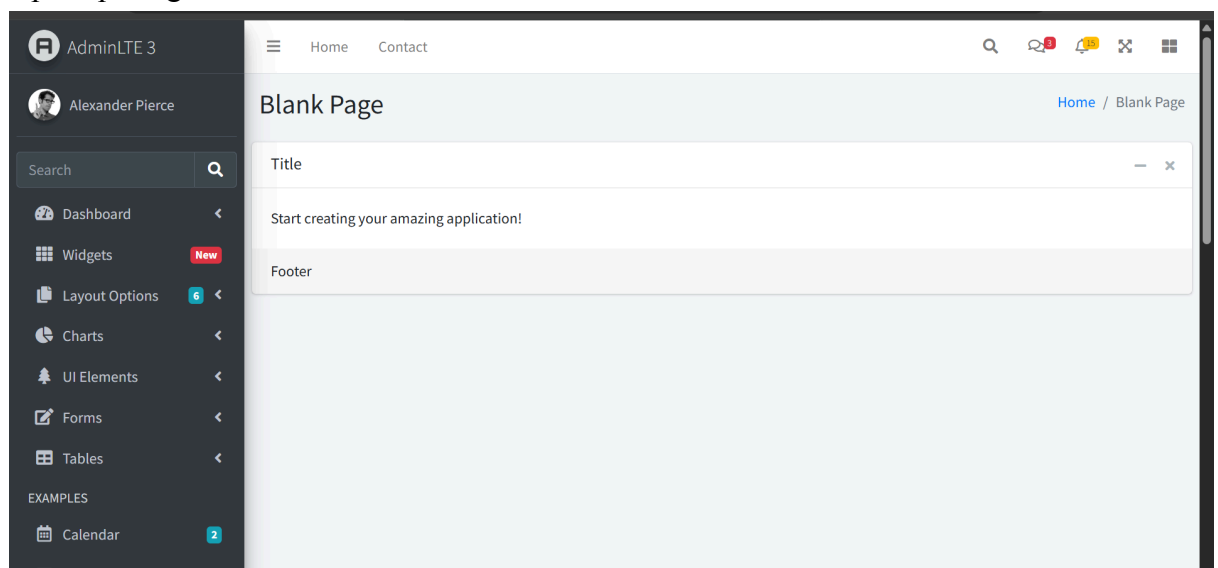
3. Selanjutnya kita buka di browser dengan alamat `http://localhost/PWL_POS/public/adminlte` maka akan muncul tampilan seperti berikut:



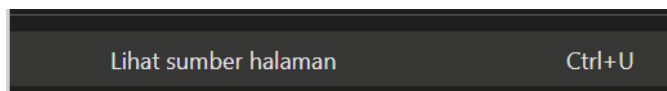
4. Kita klik menu [Extras > Blank Page](#), page inilah yang akan menjadi dasar web template



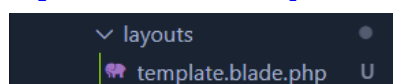
5. Dari sini kita bisa melakukan layouting halaman Blank Page ini menjadi 4 element seperti pada gambar berikut



6. Selanjutnya kita klik kanan halaman [Blank Page](#) dan klik [view page source](#)



7. Selanjutnya kita copy page source dari halaman [Blank Page](#), kemudian kita *paste* pada [PWL\\_POS/resource/view/layouts/template.blade.php](#) (buat dulu folder [layouts](#) dan file [template.blade.php](#))



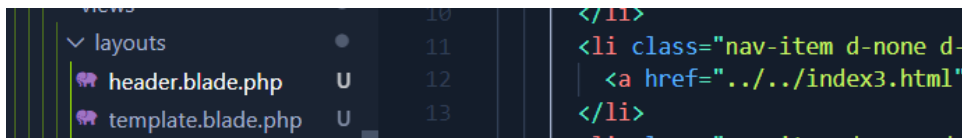
8. File [layouts/template.blade.php](#) adalah file utama untuk templating website
9. Pada baris 1-14 file [template.blade.php](#), kita modifikasi menjadi

```

1
2 <!DOCTYPE html>
3 <html lang="en">
4 <head>
5   <meta charset="utf-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1">
7   <title>{{ config('app.name', 'PWL Laravel Starter Code') }}</title>
8
9   <!-- Google Font: Source Sans Pro -->
10  <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Source+Sans+Pro:300,400,400i,700&display=fallback">
11  <!-- Font Awesome -->
12  <link rel="stylesheet" href="{{ asset('adminlte/plugins/fontawesome-free/css/all.min.css') }}">
13  <!-- Theme style -->
14  <link rel="stylesheet" href="{{ asset('adminlte/dist/css/adminlte.min.css') }}">

```

10. Kemudian kita blok baris 19-153 (baris untuk element 1-header), lalu kita *cut*, dan *paste*-kan di file `PWL_POS/resource/view/layouts/header.blade.php` (buat dulu file `header.blade.php` jika belum ada).



Sehingga tampilan dari file `template.blade.php` menjadi seperti berikut

```

14 </head>
15 <body class="hold-transition sidebar-mini">
16 <!-- Site wrapper -->
17 <div class="wrapper">
18   <!-- Navbar -->
19   @include('layouts.header')
20   <!-- /.navbar -->
21
22   <!-- Main Sidebar Container -->
23   <aside class="main-sidebar sidebar-dark-primary elevation-4">
24     <!-- Brand Logo -->
25     <a href="../../index3.html" class="brand-link">
26       
27       <span class="brand-text font-weight-light">AdminLTE 3</span>
28     </a>

```

Baris 19 adalah komponen Blade untuk memanggil elemen `layouts/header.blade.php` agar menjadi satu dengan `template.blade.php` saat di-render nanti.

11. Kita modifikasi baris 25 dan 26 pada `template.blade.php` menjadi

```

22 <!-- Main Sidebar Container -->
23 <aside class="main-sidebar sidebar-dark-primary elevation-4">
24   <!-- Brand Logo -->
25   <a href="{{ url('/') }}" class="brand-link">
26     
27     <span class="brand-text font-weight-light">PWL - Starter Code</span>
28   </a>

```

12. Selanjutnya kita blok baris 31-693 (baris untuk element 2-sidebar), lalu kita *cut*, dan *paste*-kan di file `PWL_POS/resource/view/layouts/sidebar.blade.php` (buat dulu file `sidebar.blade.php` jika belum ada). Sehingga tampilan dari file `template.blade.php` menjadi seperti berikut



Sehingga tampilan dari file `template.blade.php` menjadi seperti berikut

```
35 <!-- Content Wrapper. Contains page content -->
36 <div class="content-wrapper">
37     <!-- Content Header (Page header) -->
38     @include('layouts.breadcrumb')
39
40     <!-- Main content -->
41     <section class="content">
42         <!-- Default box -->
43         <div class="card">
44             <div class="card-header">
45                 <h3 class="card-title">Title</h3>
```

17. Layout terakhir adalah pada bagian konten. Layout untuk konten bisa kita buat dinamis, sesuai dengan apa yang ingin kita sajikan pada web yang kita bangun.

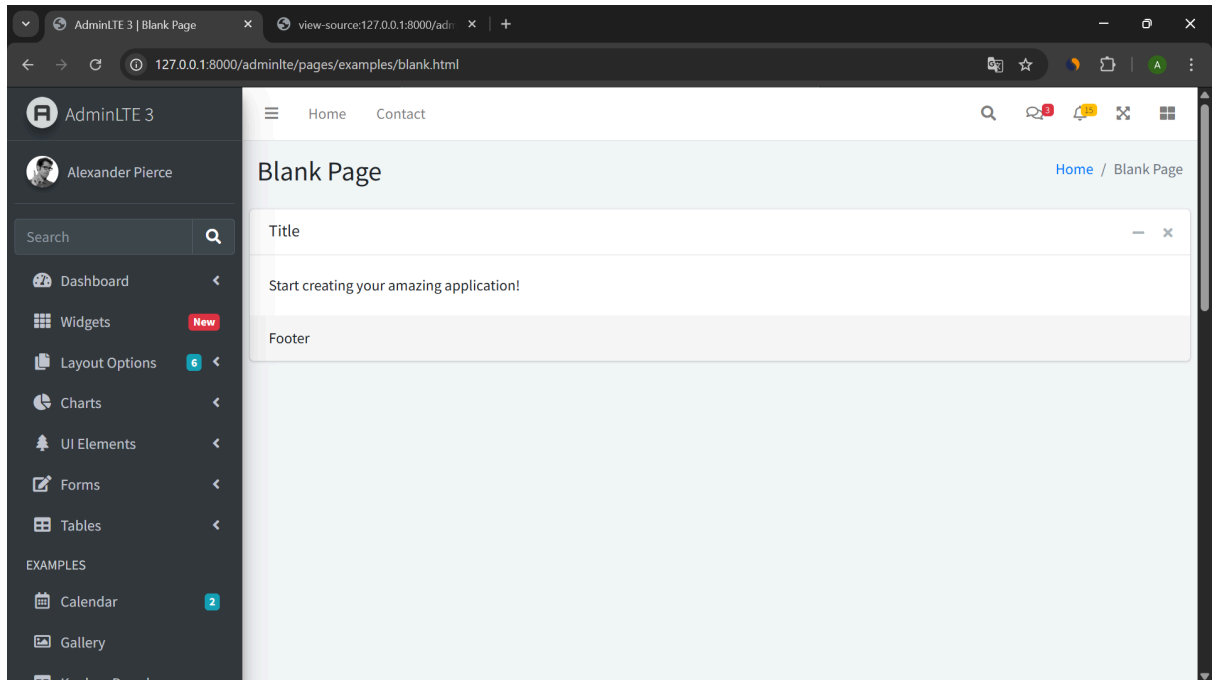
18. Untuk content, kita akan menghapus baris 42-66 pada file `template.blade.php`. dan kita ganti dengan kode seperti ini `@yield('content')`

```
40 <!-- Main content -->
41 <section class="content">
42     @yield('content')
43 </section>
44 <!-- /.content -->
```

19. Hasil akhir pada file utama `layouts/template.blade.php` adalah seperti berikut

```
2 <html lang="en">
15 <body class="hold-transition sidebar-mini">
17 <div class="wrapper">
18     <!-- Navbar -->
19     @include('layouts.header')
20     <!-- /.navbar -->
21
22     <!-- Main Sidebar Container -->
23     <aside class="main-sidebar sidebar-dark-primary elevation-4">
24         <!-- Brand Logo -->
25         <a href="{{ url('/') }}" class="brand-link">
26             
27             <span class="brand-text font-weight-light">PWL - Starter Code</span>
28         </a>
29
30         <!-- Sidebar -->
31         @include('layouts.sidebar')
32         <!-- /.sidebar -->
33     </aside>
34
35     <!-- Content Wrapper. Contains page content -->
36     <div class="content-wrapper">
37         <!-- Content Header (Page header) -->
38         @include('layouts.breadcrumb')
39
40         <!-- Main content -->
41         <section class="content">
42             @yield('content')
43         </section>
44         <!-- /.content -->
45     </div>
46     <!-- /.content-wrapper -->
47
48     @include('layouts.footer')
49 </div>
50 <!-- /.wrapper -->
51
52 <!-- jQuery -->
53 <script src="{{ asset('adminlte/plugins/jquery/jquery.min.js') }}"></script>
```

20. Selamat kalian sudah selesai dalam melakukan layouting website di laravel.
21. Jangan lupa commit dan push ke github untuk praktikum 1 ini



## Praktikum 2 - Penerapan Layouting

1. Kita buat file controller dengan nama WelcomeController.php

```
1 <?php
2 namespace App\Http\Controllers;
3
4 use PhpParser\Node\Expr\Cast\Object_;
5
6 class WelcomeController extends Controller{
7     public function index(){
8         $breadcrumb = (object)[
9             'title' => 'Selamat Datang',
10            'list' => ['Home', 'Welcome']
11        ];
12
13        $activeMenu = 'dashboard';
14
15        return view('welcome', ['breadcrumb' => $breadcrumb, 'activeMenu' => $activeMenu]);
16    }
17 }
```

2. Kita buat file pada PWL `POS/resources/views/welcome.blade.php`

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

@section('content')

<div class="card">
    <div class="card-header">
        <h3 class="card-title">Halo, apakabar!!!</h3>
        <div class="card-tools"></div>
    </div>
    <div class="card-body">
        Selamat datang semua, ini adalah halaman utama dari aplikasi ini
    </div>
</div>

@endsection
```

3. Kita modifikasi file PWL\_POS/resources/views/layouts/breadcrumb.blade.php

```
1 <section class="content-header">
2     <div class="container-fluid">
3         <div class="row mb-2">
4             <div class="col-sm-6"><h1>{{ $breadcrumb->title }}</h1></div>
5             <div class="col-sm-6">
6                 <ol class="breadcrumb float-sm-right">
7                     @foreach($breadcrumb->list as $key => $value)
8                         @if($key == count($breadcrumb->list) - 1)
9                             <li class="breadcrumb-item active">{{ $value }}</li>
10                        @else
11                            <li class="breadcrumb-item">{{ $value }}</li>
12                        @endif
13                    @endforeach
14                </ol>
15            </div>
16        </div>
17    </div><!-- /.container-fluid -->
18 </section>
```

4. Kita modifikasi file PWL\_POS/resources/views/layouts/sidebar.blade.php

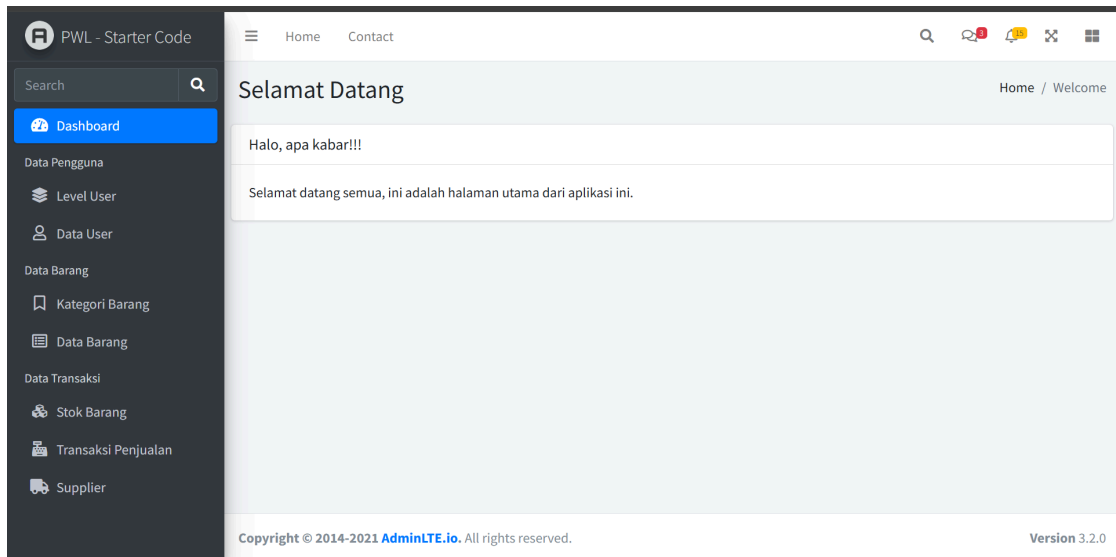
```
14 <!-- Sidebar Menu -->
15 <nav class="mt-2">
16     <ul class="nav nav-pills nav-sidebar flex-column" data-widget="treeview"
17     role="menu" data-accordion="false">
18         <li class="nav-item">
19             <a href="{{ url('/') }}" class="nav-link {{ ($activeMenu == 'dashboard')?
20             'active' : '' }}">
21                 <i class="nav-icon fas fa-tachometer-alt"></i>
22                 <p>Dashboard</p>
23             </a>
24         </li>
25         <li class="nav-header">Data Pengguna</li>
26         <li class="nav-item">
27             <a href="{{ url('/level') }}" class="nav-link {{ ($activeMenu == 'level')?
28             'active' : '' }}">
29                 <i class="nav-icon fas fa-layer-group"></i>
30                 <p>Level User</p>
31             </a>
32         </li>
33         <li class="nav-item">
34             <a href="{{ url('/user') }}" class="nav-link {{ ($activeMenu == 'user')?
35             'active' : '' }}">
```

5. Kita tambahkan kode berikut router web.php

```
Route::get('/', [WelcomeController::class, 'index']);
```

6. Sekarang kita coba jalankan di browser dengan mengetikkan url  
http://localhost/PWL\_POS/public





7. Jangan lupa commit dan push ke github PWL\_POS kalian

### Praktikum 3 – Implementasi jQuery Datatable di AdminLTE

1. Kita modifikasi proses CRUD pada tabel m\_user pada praktikum ini
2. Kita gunakan library Yajra-datatable dengan mengetikkan perintah pada CMD  
composer require yajra/laravel-datatables:^10.0 atau composer require yajra/laravel-datatables-oracle
3. Kita modifikasi route web.php untuk proses CRUD user

```
Route::group(['prefix' => 'user'], function () {
    Route::get('/', [UserController::class, 'index']); // menampilkan halaman awal user
    Route::post('/list', [UserController::class, 'list']); // menampilkan data user dalam bentuk JSON untuk datatables
    Route::get('/create', [UserController::class, 'create']); // menampilkan halaman form tambah user
    Route::post('/', [UserController::class, 'store']); // menyimpan data user baru
    Route::get('/{id}', [UserController::class, 'show']); // menampilkan detail user
    Route::get('/{id}/edit', [UserController::class, 'edit']); // menampilkan halaman form edit user
    Route::put('/{id}', [UserController::class, 'update']); // menyimpan perubahan data user
    Route::delete('/{id}', [UserController::class, 'destroy']); // menghapus data user
});
```

4. Kita buat atau modifikasi penuh untuk UserController.php. Kita buat fungsi index() untuk menampilkan halaman awal user

```
class UserController extends Controller
{
    // Menampilkan halaman awal user
    public function index()
    {
        $breadcrumb = (object) [
            'title' => 'Daftar User',
            'list' => ['Home', 'User']
        ];

        $page = (object) [
            'title' => 'Daftar user yang terdaftar dalam sistem'
        ];

        $activeMenu = 'user'; // set menu yang sedang aktif

        return view('user.index', ['breadcrumb' => $breadcrumb, 'page' => $page, 'activeMenu' => $activeMenu]);
    }
}
```

5. Lalu kita buat view pada PWL/resources/views/user/index.blade.php

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

@section('content')
    <div class="card card-outline card-primary">
        <div class="card-header">
            <h3 class="card-title">{{ $page->title }}</h3>
            <div class="card-tools">
                <a class="btn btn-sm btn-primary mt-1" href="{{ url('user/create') }}">Tambah</a>
            </div>
        </div>
        <div class="card-body">
            <table class="table table-bordered table-striped table-hover table-sm"
id="table_user">
                <thead>
                    <tr><th>ID</th><th>Username</th><th>Nama</th><th>Level</th><th>Aksi</th></tr>
                </thead>
                <tbody>
                    <tr>
                        <td>1</td>
                        <td>Pengguna</td>
                        <td>Pengguna</td>
                        <td>Pengguna</td>
                        <td>Pengguna</td>
                    </tr>
                </tbody>
            </table>
        </div>
    </div>
@endsection

@push('css')
@endpush

@push('js')
    <script>
        $(document).ready(function() {
            var dataUser = $('#table_user').DataTable({
                // serverSide: true, jika ingin menggunakan server side processing
                serverSide: true,
                ajax: {
                    "url": "{{ url('user/list') }}",
                    "dataType": "json",
                    "type": "POST"
                },
                columns: [
                    {

```

```

        // nomor urut dari laravel datatable addIndexColumn()
        data: "DT_RowIndex",
        className: "text-center",
        orderable: false,
        searchable: false
    }, {
        data: "username",
        className: "",
        // orderable: true, jika ingin kolom ini bisa diurutkan
        orderable: true,
        // searchable: true, jika ingin kolom ini bisa dicari
        searchable: true
    }, {
        data: "nama",
        className: "",
        orderable: true,
        searchable: true
    }, {
        // mengambil data level hasil dari ORM berelasi
        data: "level.level_nama",
        className: "",
        orderable: false,
        searchable: false
    }, {
        data: "aksi",
        className: "",
        orderable: false,
        searchable: false
    }
    ]
});
});
</script>
@endpush

```

6. Kemudian kita modifikasi file template.blade.php untuk menambahkan library jquery datatables dari template AdminLTE yang kita download dan berada di folder public

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>{{ config('app.name', 'PWL Laravel Starter Code') }}</title>

    <meta name="csrf-token" content="{{ csrf_token() }}"> <!-- untuk mengirimkan token Laravel CSRF pada setiap request ajax -->

    <!-- 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('adminlte/plugins/fontawesome-free/css/all.min.css') }}">
    <!-- DataTables -->
    <link rel="stylesheet" href="{{ asset('adminlte/plugins/datatables-bs4/css/dataTables.bootstrap4.min.css') }}">
    <link rel="stylesheet" href="{{ asset('adminlte/plugins/datatables-responsive/css/responsive.bootstrap4.min.css') }}">
    <link rel="stylesheet" href="{{ asset('adminlte/plugins/datatables-buttons/css/buttons.bootstrap4.min.css') }}">
    <!-- Theme style -->
    <link rel="stylesheet" href="{{ asset('adminlte/dist/css/adminlte.min.css') }}">

    @stack('css') <!-- Digunakan untuk memanggil custom CSS dari perintah push('css') pada masing-masing view -->
</head>

```

```

<!-- jQuery -->
<script src="{{ asset('adminlte/plugins/jquery/jquery.min.js') }}"></script>
<!-- Bootstrap 4 -->
<script src="{{ asset('adminlte/plugins/bootstrap/js/bootstrap.bundle.min.js') }}"></script>
<!-- DataTables & Plugins -->
<script src="{{ asset('adminlte/plugins/datatables/jquery.dataTables.min.js') }}"></script>
<script src="{{ asset('adminlte/plugins/datatables-bs4/js/dataTables.bootstrap4.min.js') }}"></script>
<script src="{{ asset('adminlte/plugins/datatables-responsive/js/dataTables.responsive.min.js') }}"></script>
<script src="{{ asset('adminlte/plugins/datatables-responsive/js/responsive.bootstrap4.min.js') }}"></script>
<script src="{{ asset('adminlte/plugins/datatables-buttons/js/dataTables.buttons.min.js') }}"></script>
<script src="{{ asset('adminlte/plugins/datatables-buttons/js/buttons.bootstrap4.min.js') }}"></script>
<script src="{{ asset('adminlte/plugins/jszip/jszip.min.js') }}"></script>
<script src="{{ asset('adminlte/plugins/pdfmake/pdfmake.min.js') }}"></script>
<script src="{{ asset('adminlte/plugins/pdfmake/vfs_fonts.js') }}"></script>
<script src="{{ asset('adminlte/plugins/datatables-buttons/js/buttons.html5.min.js') }}"></script>
<script src="{{ asset('adminlte/plugins/datatables-buttons/js/buttons.print.min.js') }}"></script>
<script src="{{ asset('adminlte/plugins/datatables-buttons/js/buttons.colVis.min.js') }}"></script>
<!-- AdminLTE App -->
<script src="{{ asset('adminlte/dist/js/adminlte.min.js') }}"></script>
</script>
// Untuk mengirimkan token Laravel CSRF pada setiap request ajax
$.ajaxSetup({headers: { 'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('content')}});
</script>
@stack('js') <!-- Digunakan untuk memanggil custom js dari perintah push('js') pada masing-masing view -->
</body>
</html>

```

7. Untuk bisa menangkap request data untuk datatable, kita buat fungsi list() pada UserController.php seperti berikut

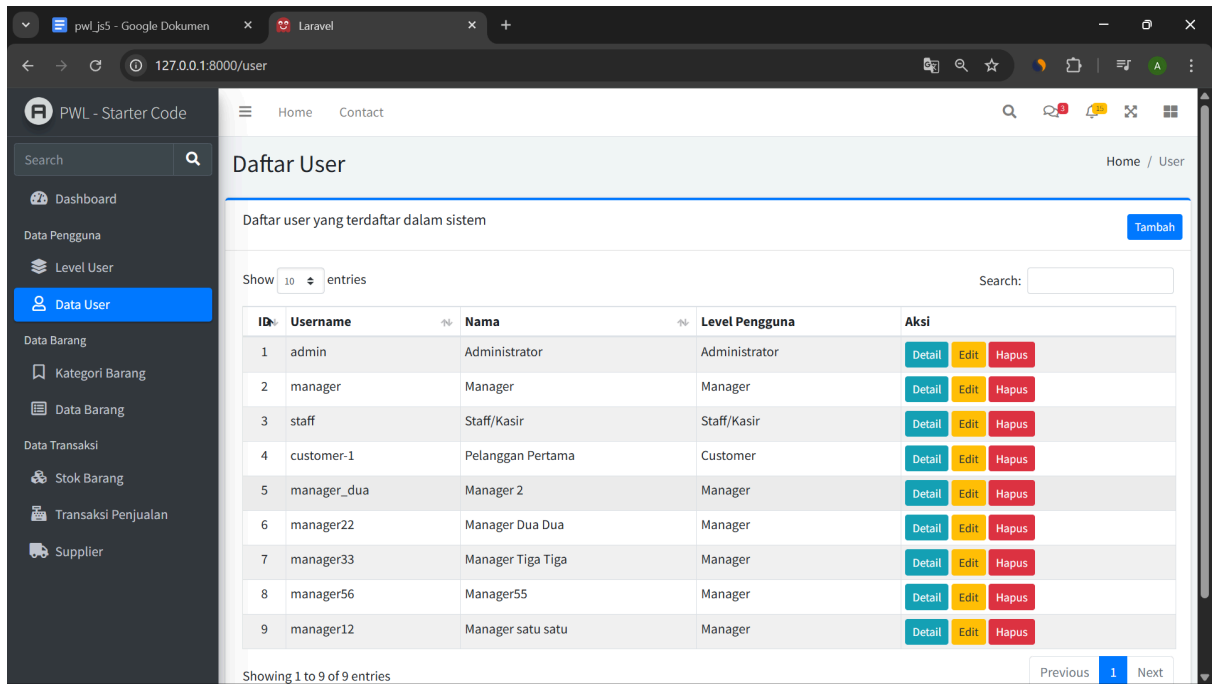
```

// Ambil data user dalam bentuk json untuk datatables
public function list(Request $request)
{
    $users = UserModel::select('user_id', 'username', 'nama', 'level_id')
        ->with('level');

    return DataTables::of($users)
        // menambahkan kolom index / no urut (default nama kolom: DT_RowIndex)
        ->addIndexColumn()
        ->addColumn('aksi', function ($user) { // menambahkan kolom aksi
            $btn = '<a href="'.url('/user/'. $user->user_id).' " class="btn btn-info btn-sm">Detail</a> ';
            $btn .= '<a href="'.url('/user/'. $user->user_id . '/edit')." " class="btn btn-warning btn-sm">Edit</a> ';
            $btn .= '<form class="d-inline-block" method="POST" action="'.url('/user/'. $user->user_id).' ">
                . csrf_field() . method_field('DELETE') .
                '<button type="submit" class="btn btn-danger btn-sm" onclick="return
                    confirm(\'Apakah Anda yakin menghapus data ini?\');">Hapus</button></form>';
            return $btn;
        })
        ->rawColumns(['aksi']) // memberitahu bahwa kolom aksi adalah HTML
        ->make(true);
}

```

8. Sekarang coba jalankan browser, dan klik menu Data User..!!! perhatikan dan amati apa yang terjadi.



9. Selanjutnya kita modifikasi UserController.php untuk form tambah data user

```
// Menampilkan halaman form tambah user
public function create()
{
    $breadcrumb = (object) [
        'title' => 'Tambah User',
        'list' => ['Home', 'User', 'Tambah']
    ];

    $page = (object) [
        'title' => 'Tambah user baru'
    ];

    $level = LevelModel::all(); // ambil data level untuk ditampilkan di form
    $activeMenu = 'user'; // set menu yang sedang aktif

    return view('user.create', ['breadcrumb' => $breadcrumb, 'page' => $page, 'level' => $level, 'activeMenu' => $activeMenu]);
}
```

10. Sekarang kita buat form untuk menambah data, kita buat file  
PWL/resources/views/user/create.blade.php

```

<5 > Jobsheet5.2 > resources > views > user > create.blade.php > ...
@extends('layouts.template')

@section('content')
<div class="card card-outline card-primary">
    <div class="card-header">
        <h3 class="card-title">{{ $page->title }}</h3>
        <div class="card-tools"></div>
    </div>
    <div class="card-body">
        <form method="POST" action="{{ url('user') }}" class="form-horizontal">
            @csrf
            <div class="form-group row">
                <label class="col-1 control-label col-form-label">Level</label>
                <div class="col-11">
                    <select class="form-control" id="level_id" name="level_id" required>
                        <option value="">- Pilih Level -</option>
                        @foreach($level as $item)
                            <option value="{{ $item->level_id }}">{{ $item->level_nama }}</option>
                        @endforeach
                    </select>
                    @error('level_id')
                        <small class="form-text text-danger">{{ $message }}</small>
                    @enderror
                </div>
            </div>

            <div class="form-group row">
                <label class="col-1 control-label col-form-label">Username</label>
                <div class="col-11">
                    <input type="text" class="form-control" id="username" name="username"
                        value="{{ old('username') }}" required>
                    @error('username')
                        <small class="form-text text-danger">{{ $message }}</small>
                    @enderror
                </div>
            </div>

            <div class="form-group row">
                <label class="col-1 control-label col-form-label">Nama</label>
                <div class="col-11">
                    <input type="text" class="form-control" id="nama" name="nama"
                        value="{{ old('nama') }}" required>
                    @error('nama')
                        <small class="form-text text-danger">{{ $message }}</small>
                    @enderror
                </div>
            </div>

            <div class="form-group row">
                <label class="col-1 control-label col-form-label">Password</label>
                <div class="col-11">
                    <input type="password" class="form-control" id="password" name="password" required>
                    @error('password')
                        <small class="form-text text-danger">{{ $message }}</small>
                    @enderror
                </div>
            </div>

            <div class="form-group row">
                <label class="col-1 control-label col-form-label"></label>
                <div class="col-11">
                    <button type="submit" class="btn btn-primary btn-sm">Simpan</button>
                    <a class="btn btn-sm btn-default ml-1" href="{{ url('user') }}">Kembali</a>
                </div>
            </div>
        </form>
    </div>
</div>
@endsection

@push('css')
@endpush

@push('js')
@endpush

```

11. Kemudian untuk bisa menghandle data yang akan disimpan ke database, kita buat fungsi store() di UserController.php

```
// Menyimpan data user baru
public function store(Request $request)
{
    $request->validate([
        // username harus diisi, berupa string, minimal 3 karakter, dan bernilai unik di tabel m_user kolom username
        'username' => 'required|string|min:3|unique:m_user,username',
        'nama' => 'required|string|max:100', // nama harus diisi, berupa string, dan maksimal 100 karakter
        'password' => 'required|min:5', // password harus diisi dan minimal 5 karakter
        'level_id' => 'required|integer' // level_id harus diisi dan berupa angka
    ]);

    UserModel::create([
        'username' => $request->username,
        'nama' => $request->nama,
        'password' => bcrypt($request->password), // password dienkripsi sebelum disimpan
        'level_id' => $request->level_id
    ]);

    return redirect('/user')->with('success', 'Data user berhasil disimpan');
}
```

12. Sekarang coba kalian buka form tambah data user dengan klik tombol tambah. Amati dan pelajari...!!!

The screenshot shows a web browser at the URL 127.0.0.1:8000/user/create. The page title is 'Tambah User'. The left sidebar shows a navigation menu with 'Data User' highlighted. The main content area has the heading 'Tambah user baru'. The form fields are as follows:

- Level:** A dropdown menu with the selected option '- Pilih Level -'.
- Username:** A text input field containing 'customer-2'.
- Nama:** A text input field containing 'Pelanggan Dua'.
- Password:** A text input field with masked characters '.....'.

At the bottom of the form are two buttons: 'Simpan' (Save) and 'Kembali' (Back). The footer of the page indicates 'Copyright © 2014-2021 AdminLTE.io. All rights reserved.' and 'Version 3.2.0'.

This screenshot shows the same 'Tambah User' form, but with the 'Level' dropdown menu now set to 'Customer'. The other fields remain the same: 'Username' is 'customer-2', 'Nama' is 'Pelanggan Dua', and 'Password' is masked. The 'Simpan' and 'Kembali' buttons are still present at the bottom.

Data user berhasil disimpan

Show 10 entries Search:

ID	Username	Nama	Level Pengguna	Aksi
1	admin	Administrator	Administrator	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
2	manager	Manager	Manager	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
3	staff	Staff/Kasir	Staff/Kasir	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
4	customer-1	Pelanggan Satu	Customer	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
5	manager_dua	Manager 2	Manager	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
6	manager22	Manager Dua Dua	Manager	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
7	manager33	Manager Tiga Tiga	Manager	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
8	manager56	Manager55	Manager	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
9	manager12	Manager satu satu	Manager	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
10	customer-2	Pelanggan Dua	Customer	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>

13. Selanjutnya, kita masuk pada bagian menampilkan detail data user (klik tombol [Detail](#) ) pada halaman user. Route yang bertugas untuk menangkap request detail adalah

```
Route::get('/{id}', [UserController::class, 'show']); // menampilkan detail user
```

14. Jadi kita buat/modifikasi fungsi show() pada UserController.php seperti berikut

```
// Menampilkan detail user
public function show(string $id)
{
    $user = UserModel::with('level')->find($id);

    $breadcrumb = (object) [
        'title' => 'Detail User',
        'list' => ['Home', 'User', 'Detail']
    ];

    $page = (object) [
        'title' => 'Detail user'
    ];

    $activeMenu = 'user'; // set menu yang sedang aktif

    return view('user.show', ['breadcrumb' => $breadcrumb, 'page' => $page, 'user' => $user, 'activeMenu' => $activeMenu]);
}
```

15. Kemudian kita buat view di PWL/resources/views/user/show.blade.php



```

@extends('layouts.template')

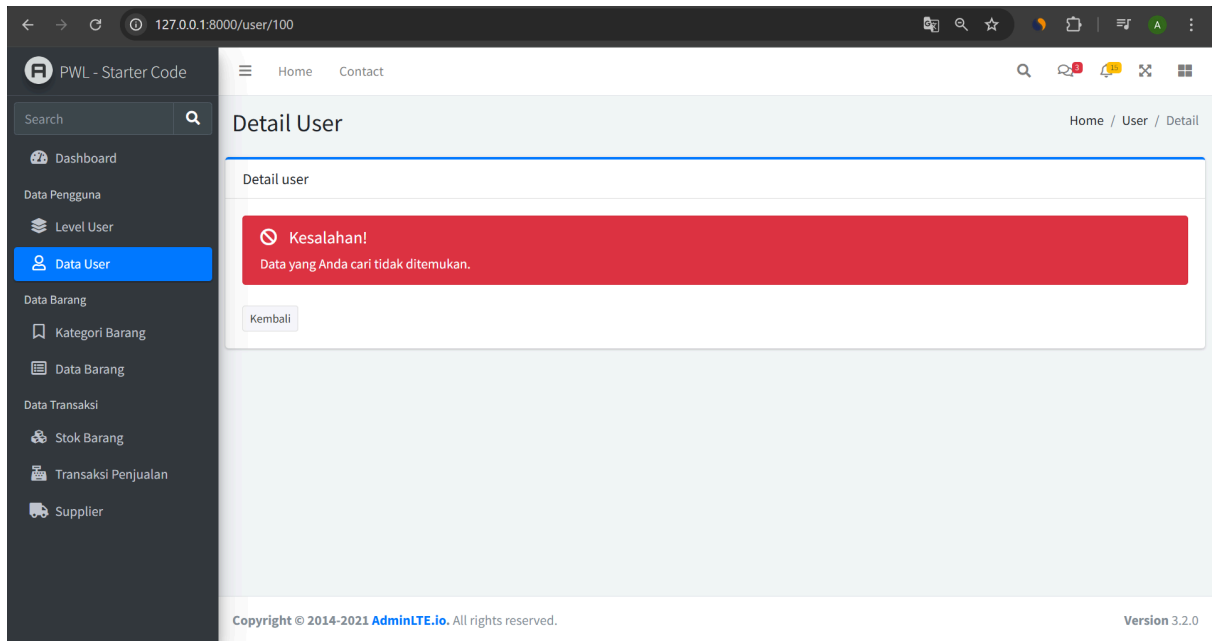
@section('content')
    <div class="card card-outline card-primary">
        <div class="card-header">
            <h3 class="card-title">{{ $page->title }}</h3>
            <div class="card-tools"></div>
        </div>
        <div class="card-body">
            @empty($user)
                <div class="alert alert-danger alert-dismissible">
                    <h5><i class="icon fas fa-ban"></i> Kesalahan!</h5>
                    Data yang Anda cari tidak ditemukan.
                </div>
            @else
                <table class="table table-bordered table-striped table-hover table-sm">
                    <tr>
                        <th>ID</th>
                        <td>{{ $user->user_id }}</td>
                    </tr>
                    <tr>
                        <th>Level</th>
                        <td>{{ $user->level->level_nama }}</td>
                    </tr>
                    <tr>
                        <th>Username</th>
                        <td>{{ $user->username }}</td>
                    </tr>
                    <tr>
                        <th>Nama</th>
                        <td>{{ $user->nama }}</td>
                    </tr>
                    <tr>
                        <th>Password</th>
                        <td>*****</td>
                    </tr>
                </table>
            @endempty
        </div>
        <a href="{{ url('user') }}" class="btn btn-sm btn-default mt-2">Kembali</a>
    </div>
</div>
@endsection

@push('css')
@endpush

@push('js')
@endpush

```

16. Sekarang kalian coba untuk melihat detail data user di browser, dan coba untuk mengetikkan id yang salah contoh <http://localhost/PWL/public/user/100> amati apa yang terjadi, dan laporkan!!!



17. Selanjutnya, kita masuk pada bagian untuk memodifikasi data user. Route yang bertugas untuk menangkap request edit adalah

```
Route::get('/{id}/edit', [UserController::class, 'edit']); // menampilkan halaman form edit user
Route::put('/{id}', [UserController::class, 'update']); // menyimpan perubahan data user
```

18. Jadi kita buat fungsi edit() dan update() pada UserController.php

```
// Menampilkan halaman form edit user
public function edit(string $id)
{
    $user = UserModel::find($id);
    $level = LevelModel::all();

    $breadcrumb = (object) [
        'title' => 'Edit User',
        'list' => ['Home', 'User', 'Edit']
    ];

    $page = (object) [
        'title' => 'Edit user'
    ];

    $activeMenu = 'user'; // set menu yang sedang aktif

    return view('user.edit', [
        'breadcrumb' => $breadcrumb,
        'page' => $page,
        'user' => $user,
        'level' => $level,
        'activeMenu' => $activeMenu
    ]);
}
```

```
// Menyimpan perubahan data user
public function update(Request $request, string $id)
{
    $request->validate([
        // username harus diisi, berupa string, minimal 3 karakter,
        // dan bernilai unik di tabel m_user kolom username kecuali untuk user dengan id yang sedang diedit
        'username' => 'required|string|min:3|unique:m_user,user,username,',$id.',user_id',
        // nama harus diisi, berupa string, dan maksimal 100 karakter
        'nama' => 'required|string|max:100',
        // password bisa diisi (minimal 5 karakter) dan bisa tidak diisi
        'password' => 'nullable|min:5',
        // level_id harus diisi dan berupa angka
        'level_id' => 'required|integer'
    ]);

    UserModel::find($id)->update([
        'username' => $request->username,
        'nama' => $request->nama,
        'password' => $request->password ? bcrypt($request->password) : UserModel::find($id)->password,
        'level_id' => $request->level_id
    ]);

    return redirect('/user')->with('success', 'Data user berhasil diubah');
}
```

19. Selanjutnya, kita buat view untuk melakukan proses edit data user di PWL/resources/views/user/edit.blade.php

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

@section('content')
<div class="card card-outline card-primary">
    <div class="card-header">
        <h3 class="card-title">{{ $page->title }}</h3>
        <div class="card-tools"></div>
    </div>
    <div class="card-body">
        @empty($user)
            <div class="alert alert-danger alert-dismissible">
                <h5><i class="icon fas fa-ban"></i> Kesalahan!</h5>
                Data yang Anda cari tidak ditemukan.
            </div>
            <a href="{{ url('user') }}" class="btn btn-sm btn-default mt-2">Kembali</a>
        @else
            <form method="POST" action="{{ url('/user/'.$user->user_id) }}" class="form-horizontal">
                @csrf
                {!! method_field('PUT') !!} <!-- tambahkan baris ini untuk proses edit yang butuh method PUT -->
                <div class="form-group row">
                    <label class="col-1 control-label col-form-label">Level</label>
                    <div class="col-11">
                        <select class="form-control" id="level_id" name="level_id" required>
                            <option value="">- Pilih Level -</option>
                            @foreach($level as $item)
                                <option value="{{ $item->level_id }}" @if($item->level_id == $user->level_id) selected @endif>
                                    {{ $item->level_nama }}
                                </option>
                            @endforeach
                        </select>
                        @error('level_id')
                            <small class="form-text text-danger">{{ $message }}</small>
                        @enderror
                    </div>
                </div>
                <div class="form-group row">
                    <label class="col-1 control-label col-form-label">Username</label>
                    <div class="col-11">
                        <input type="text" class="form-control" id="username" name="username">
                    </div>
                </div>
            </form>
        @endelse
    </div>
</div>
```

```

        value="{{ old('username', $user->username) }}" required>
        @error('username')
            <small class="form-text text-danger">{{ $message }}</small>
        @enderror
    </div>
</div>
<div class="form-group row">
    <label class="col-1 control-label col-form-label">Nama</label>
    <div class="col-11">
        <input type="text" class="form-control" id="nama" name="nama"
            value="{{ old('nama', $user->nama) }}" required>
        @error('nama')
            <small class="form-text text-danger">{{ $message }}</small>
        @enderror
    </div>
</div>
<div class="form-group row">
    <label class="col-1 control-label col-form-label">Password</label>
    <div class="col-11">
        <input type="password" class="form-control" id="password" name="password">
        @error('password')
            <small class="form-text text-danger">{{ $message }}</small>
        @else
            <small class="form-text text-muted">Abaikan (jangan diisi) jika tidak ingin
                mengganti password user.</small>
        @enderror
    </div>
</div>
<div class="form-group row">
    <label class="col-1 control-label col-form-label"></label>
    <div class="col-11">
        <button type="submit" class="btn btn-primary btn-sm">Simpan</button>
        <a class="btn btn-sm btn-default ml-1" href="{{ url('user') }}">Kembali</a>
    </div>
</div>
</form>
@empty
</div>
</div>
@endsection

@push('css')
@endpush
@push('js')
@endpush

```

20. Sekarang kalian coba untuk mengedit data user di browser, amati, pahami, dan laporkan!

4	customer-1	Pelanggan Pertama	Customer	Detail	Edit	Hapus
---	------------	-------------------	----------	--------	------	-------

Edit User
Home / User / Edit

Edit user

Level
Customer

Username
customer-1

Nama
Pelanggan Satu

Password

Abaikan (jangan diisi) jika tidak ingin mengganti password user.

Simpan
Kembali

21. Selanjutnya kita akan membuat penanganan untuk tombol hapus. Router web.php yang berfungsi untuk menangkap request hapus dengan method DELETE adalah `Route::delete('/{id}', [UserController::class, 'destroy']);`
22. Jadi kita buat fungsi `destroy()` pada `UserController.php`

```
// Menghapus data user
public function destroy(string $id)
{
    $check = UserModel::find($id);
    if (!$check) { // untuk mengecek apakah data user dengan id yang dimaksud ada atau tidak
        return redirect('/user')->with('error', 'Data user tidak ditemukan');
    }

    try {
        UserModel::destroy($id); // Hapus data Level

        return redirect('/user')->with('success', 'Data user berhasil dihapus');
    } catch (\Illuminate\Database\QueryException $e) {

        // Jika terjadi error ketika menghapus data, redirect kembali ke halaman dengan membawa pesan error
        return redirect('/user')->with('error', 'Data user gagal dihapus karena masih terdapat tabel lain yang terkait dengan data ini');
    }
}
```

23. Selanjutnya kita modifikasi file `PWL/resources/views/user/index.blade.php` untuk menambahkan tampilan jika ada pesan error

```
<div class="card-body">
    @if (session('success'))
        <div class="alert alert-success">{{ session('success') }}</div>
    @endif
    <table class="table table-bordered table-striped table-hover table-sm" id="table_user">
        <thead>
            <tr><th>ID</th><th>Username</th><th>Nama</th><th>Level Pengguna</th><th>Aksi</th></tr>
        </thead>
    </table>
</div>
```

menjadi

```

<div class="card-body">
  @if (session('success'))
    <div class="alert alert-success">{{ session('success') }}</div>
  @endif
  @if (session('error'))
    <div class="alert alert-danger">{{ session('error') }}</div>
  @endif
  <table class="table table-bordered table-striped table-hover table-sm" id="table_user">
    <thead>
      <tr><th>ID</th><th>Username</th><th>Nama</th><th>Level Pengguna</th><th>Aksi</th></tr>
    </thead>
  </table>
</div>

```

24. Kemudian jalankan browser untuk menghapus salah satu data user. Amati dan laporkan!

Daftar User
Home / User

Daftar user yang terdaftar dalam sistem
Tambah

Show 10 entries
Search:

ID	Username	Nama	Level Pengguna	Aksi
1	admin	Administrator	Administrator	Detail Edit Hapus
2	customer-1	Pelanggan Satu	Customer	Detail Edit Hapus
3	customer-2	Pelanggan Dua	Customer	Detail Edit Hapus
4	manager	Manager	Manager	Detail Edit Hapus
5	manager_dua	Manager 2	Manager	Detail Edit Hapus
6	manager12	Manager satu satu	Manager	Detail Edit Hapus
7	manager22	Manager Dua Dua	Manager	Detail Edit Hapus
8	manager33	Manager Tiga Tiga	Manager	Detail Edit Hapus
9	manager56	Manager55	Manager	Detail Edit Hapus
10	staff	Staff/Kasir	Staff/Kasir	Detail Edit Hapus

Showing 1 to 10 of 10 entries
Previous 1 Next

Kita akan mencoba menghapus data dengan username “manager56”

Daftar User Home / User

Daftar user yang terdaftar dalam sistem [Tambah](#)

Data user berhasil dihapus

Show  entries Search:

ID	Username	Nama	Level Pengguna	Aksi
1	admin	Administrator	Administrator	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
2	manager	Manager	Manager	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
3	staff	Staff/Kasir	Staff/Kasir	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
4	customer-1	Pelanggan Satu	Customer	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
5	manager_dua	Manager 2	Manager	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
6	manager22	Manager Dua Dua	Manager	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
7	manager33	Manager Tiga Tiga	Manager	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
8	manager12	Manager satu satu	Manager	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
9	customer-2	Pelanggan Dua	Customer	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>

Showing 1 to 9 of 9 entries Previous **1** Next

25. Selamat, kalian sudah membuat Laravel Starter Code untuk proses CRUD dengan menggunakan template AdminLTE dan plugin jQuery Datatables.

26. Jangan lupa commit dan push ke github PWL\_POS kalian

#### Praktikum 4 – Implementasi Filtering Datatables

1. Kita modifikasi fungsi index() di UserController.php untuk menambahkan data yang ingin dijadikan kategori untuk data filtering

```
// Menampilkan halaman awal user
public function index()
{
    $breadcrumb = (object) [
        'title' => 'Daftar User',
        'list' => ['Home', 'User']
    ];

    $page = (object) [
        'title' => 'Daftar user yang terdaftar dalam sistem'
    ];

    $activeMenu = 'user'; // set menu yang sedang aktif

    $level = LevelModel::all(); // ambil data level untuk filter level

    return view('user.index', ['breadcrumb' => $breadcrumb, 'page' => $page, 'level' => $level,
        'activeMenu' => $activeMenu]);
}
```

2. Kemudian kita modifikasi view untuk menampilkan data filtering pada PWL/resources/views/user/index.blade.php

```

<div class="row">
  <div class="col-md-12">
    <div class="form-group row">
      <label class="col-1 control-label col-form-label">Filter:</label>
      <div class="col-3">
        <select class="form-control" id="level_id" name="level_id" required>
          <option value="">- Semua -</option>
          @foreach($level as $item)
            <option value="{{ $item->level_id }}">{{ $item->level_nama }}</option>
          @endforeach
        </select>
        <small class="form-text text-muted">Level Pengguna</small>
      </div>
    </div>
  </div>
</div>

```

3. Selanjutnya, tetap pada view index.blade.php, kita tambahkan kode berikut pada deklarasi ajax di datatable. Kode ini digunakan untuk mengirimkan data untuk filtering

```

serverSide: true,
ajax: {
  "url": "{{ url('user/list') }}",
  "dataType": "json",
  "type": "POST",
  "data": function (d){
    d.level_id = $('#level_id').val();
  }
},

```

4. Kemudian kita edit pada bagian akhir script @push('js') untuk menambahkan listener jika data filtering dipilih

```

});
$('#level_id').on('change', function(){
  dataUser.ajax.reload();
});
</script>
@endpush

```

5. Tahapan akhir adalah memodifikasi fungsi list() pada UserController.php yang digunakan untuk menampilkan data pada datatable

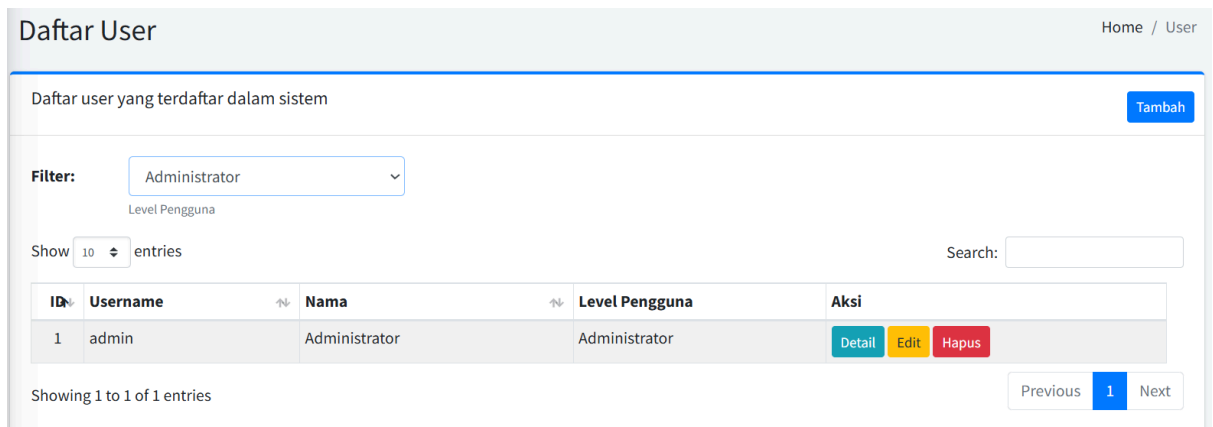
```

//Filter data user berdasarkan level_id
if ($request->level_id) {
  $users->where('level_id', $request->level_id);
}

```

6. Bagian akhir adalah kita coba jalankan di browser dengan akses menu user, maka akan tampil seperti berikut



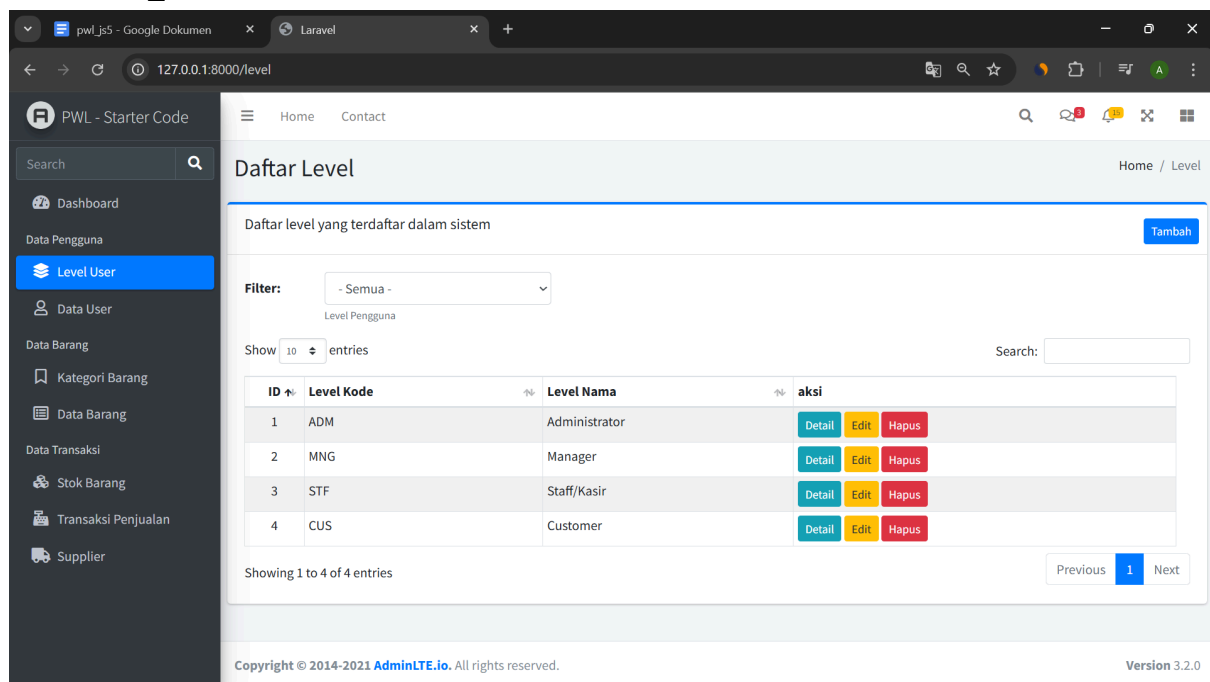


7. Selamat, sekarang Laravel Starter Code sudah ada filtering dan searching data. Starter Code sudah bisa digunakan dalam membangun sebuah sistem berbasis website.
8. Jangan lupa commit dan push ke github PWL\_POS kalian

## TUGAS

Implementasikan web layout dan datatables, pada menu berikut ini

✓ Tabel m\_level



✓ Tabel m\_kategori

pwL - Starter Code

Search

Dashboard

Data Pengguna

Level User

Data User

Data Barang

Kategori Barang

Data Barang

Data Transaksi

Stok Barang

Transaksi Penjualan

Supplier

Home / Kategori

### Daftar Kategori

Daftar kategori yang terdaftar dalam sistem

Tambah

Show 10 entries

Search:

ID	Kode Kategori	Nama Kategori	Aksi
1	FOOD	Makanan	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
2	DRINK	Minuman	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
3	BEAUTY	Kecantikan	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
4	HOME	Peralatan Rumah	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
5	BABY	Perlengkapan Bayi	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
6	SPORT	Perlengkapan Olahraga	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>

Showing 1 to 6 of 6 entries

Previous 1 Next

Copyright © 2014-2021 AdminLTE.io. All rights reserved. Version 3.2.0

## ✓ Tabel m\_supplier

pwL - Starter Code

Search

Dashboard

Data Pengguna

Level User

Data User

Data Barang

Kategori Barang

Data Barang

Data Transaksi

Stok Barang

Transaksi Penjualan

Supplier

Home / Supplier

### Daftar Supplier

Daftar supplier yang terdaftar dalam sistem

Tambah

Show 10 entries

Search:

ID	Kode Supplier	Nama Supplier	Alamat Supplier	Aksi
1	SUP001	PT Maju Mundur	Jl. Mawar No. 12, Surabaya	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
2	SUP002	PT Jaya Makmur	Jl. Kawi No. 06, Malang	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>

Showing 1 to 2 of 2 entries

Previous 1 Next

Copyright © 2014-2021 AdminLTE.io. All rights reserved. Version 3.2.0

## ✓ Tabel m\_barang

The screenshot displays a web application for managing inventory ('Data Barang'). The interface includes a dark sidebar with navigation links and a main content area. The main area has a search bar, a filter dropdown set to '- Semua -', and a table listing 8 items. Each item row contains details like ID, code, name, category, purchase price, selling price, and action buttons (Detail, Edit, Delete).

ID	Kode Barang	Nama Barang	Nama Kategori	Harga Beli	Harga Jual	Aksi
1	BRG001	Nasi Goreng	Makanan	10000	15000	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
2	BRG002	Mie Goreng	Makanan	9000	14000	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
3	BRG003	Es Teh	Minuman	3000	5000	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
4	BRG004	Jus Jeruk	Minuman	5000	8000	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
5	BRG005	Lipstik	Kecantikan	25000	40000	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
6	BRG006	Face Wash	Kecantikan	15000	25000	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
7	BRG007	Sapu Lidi	Peralatan Rumah	10000	20000	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>
8	BRG008	Pel Lantai	Peralatan Rumah	20000	35000	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Hapus</a>