TUGAS

Analisis Desain

PERTEMUAN 5

JS 7 Laravel Starter Code

OLEH

MUHAMMAD NUR AZIZ NIM. 2341720237

PROGRAM STUDI D-IV TEKNIK INFORMATIKA

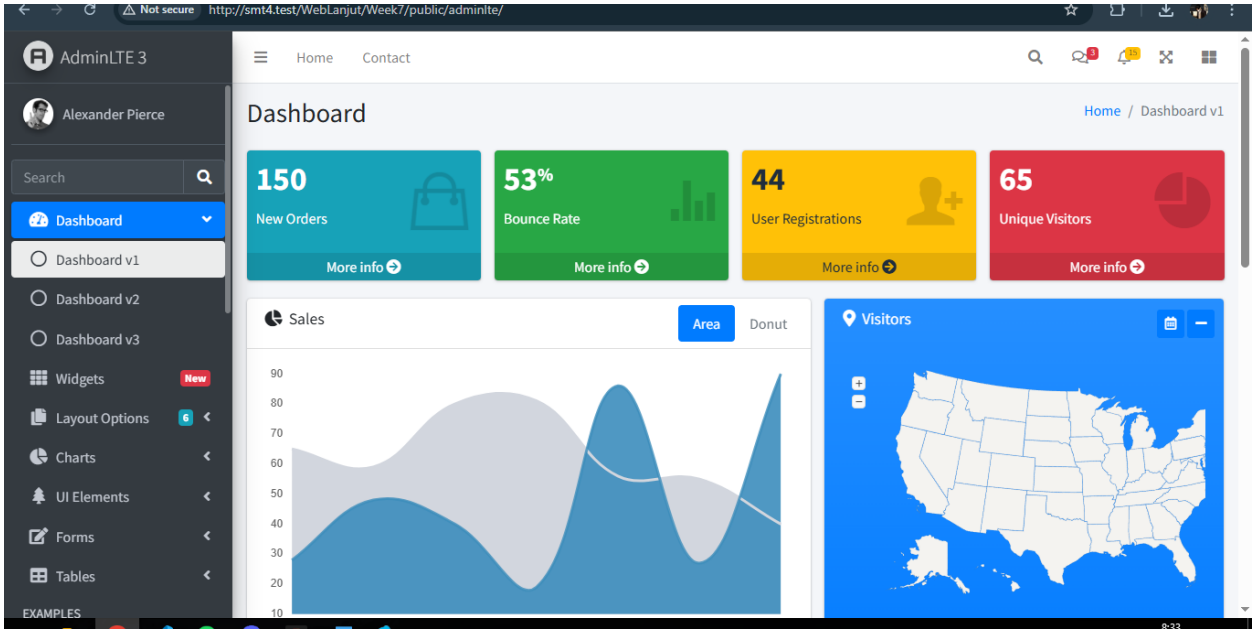
JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

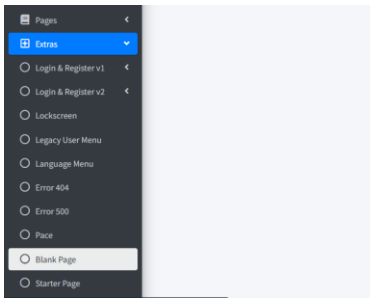
20 Maret 2025

Praktikum 1 – Layouting AdminLTE:

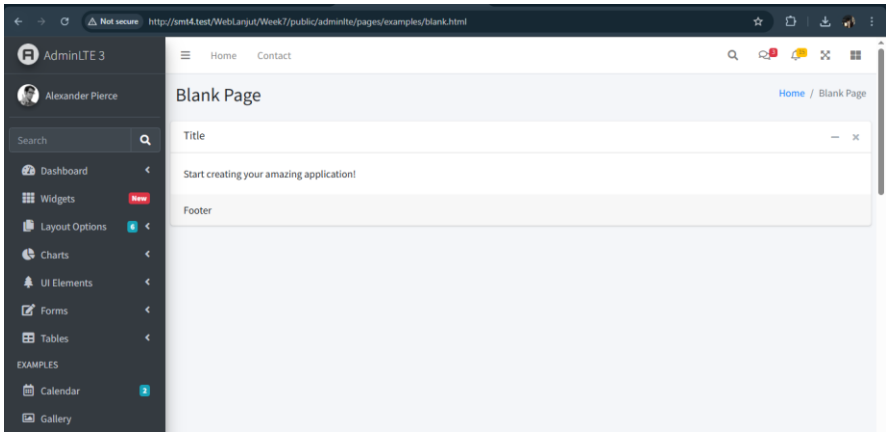
1. Kita download AdminLTE v3.2.0 yang rilis pada 8 Feb 2022
2. Seleteh kita berhasil download, kita ekstrak file yang sudah di download ke folder project PWL_POS/public, kemudian kita rename folder cukup menjadi adminlte
3. Selanjutnya kita buka di browser dengan alamat http://localhost/PWL_POS/public/adminlte maka akan muncul tampilan seperti berikut



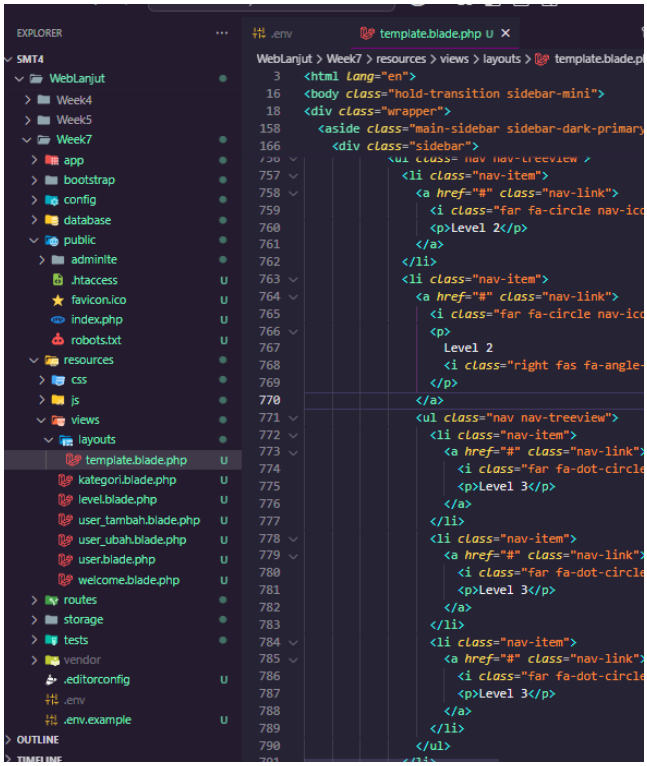
4. Kita klik menu Extras > Blank Page, page inilah yang akan menjadi dasar web template



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, kemudia 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

```
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>{{ config('app.name') }} - PM Laravel Starter Code</title>

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

<!-- Google Font: Source Sans Pro -->
<link rel="stylesheet"
    href="https://fonts.googleapis.com/css?family=Source+Sans+Pro:300,400,400i,700&displa

<!-- Font Awesome -->
<link rel="stylesheet" href="{{ asset('adminlte/plugins/fontawesome-free/css/all.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.

```
@include('layouts.header')
<!-- /.navbar -->
```

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

```
    <!-- Main Sidebar Container -->
    <aside class="main-sidebar sidebar-dark-primary elevation-4">
        <!-- Brand Logo -->
        <a href="{{ url('/') }}" class="brand-link">
            <img src="{{ asset('adminlte/dist/img/AdminLTELogo.png') }}" alt="AdminLTE Lo
                class="brand-image img">
            <span class="brand-text font-weight-light">PWL - Starter Code</span>
        </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

```
    <!-- Sidebar -->
    @include('layouts.sidebar')
    <!-- /.sidebar -->
</aside>
```

13. Selanjutnya perhatikan baris 87-98 (baris untuk element 5-footer), lalu kita cut, dan paste-kan di file PWL_POS/resource/view/layouts/footer.blade.php (buat file footer.blade.php jika belum ada). Sehingga tampilan dari file template.blade.php menjadi seperti berikut

```
            @yield('content');
        </section>
        <!-- /.content -->
    </div>
    <!-- /.content-wrapper -->
    @include('layouts.footer')
```

14. Kemudian kita modifikasi file template.blade.php baris 91-100

```
<!-- jQuery -->
<script src="{{ asset('adminlte/plugins/jquery/jquery.min.js') }}"></script>

<!-- Bootstrap 4 -->
<script src="{{ asset('adminlte/plugins/bootstrap/js/bootstrap.bundle.min.js') }}"></scri
```

```
<!-- AdminLTE App -->
<script src="{{ asset('adminlte/dist/js/adminlte.min.js') }}"></script>
```

15. Sekarang masuk pada bagian konten. Konten kita bagi menjadi 2, yaitu elemen untuk breadcrumb dan elemen untuk content.
16. Perhatikan file template.blade.php pada baris 38-52 kita jadikan sebagai elemen 4-breadcrumb. Kita blok baris 38-52 lalu kita cut, dan paste-kan di file PWL_POS/resource/view/layouts/breadcrumb.blade.php (buat file breadcrumb.blade.php jika belum ada). Sehingga tampilan dari file template.blade.php menjadi seperti berikut

```
<!-- Content Wrapper. Contains page content -->
<div class="content-wrapper">
    @include('layouts.breadcumb')
```

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

```
    <!-- Navbar -->
    @include('layouts.header')
    <!-- /.navbar -->

    <!-- Main Sidebar Container -->
    <aside class="main-sidebar sidebar-dark-primary elevation-4">
        <!-- Brand Logo -->
        <a href="{{ url('/') }}" class="brand-link">
            <img src="{{ asset('adminlte/dist/img/AdminLTELogo.png') }}" alt="AdminLTE Lo
                class="brand-image img">
            <span class="brand-text font-weight-light">PWL - Starter Code</span>
        </a>

        <!-- Sidebar -->
        @include('layouts.sidebar')
        <!-- /.sidebar -->
    </aside>

    <!-- Content Wrapper. Contains page content -->
    <div class="content-wrapper">
        @include('layouts.breadcumb')

        <!-- Main content -->
        <section class="content">
            @yield('content');
        </section>
        <!-- /.content -->
    </div>
    <!-- /.content-wrapper -->
    @include('layouts.footer')

</div>
```

20. Selamat kalian sudah selesai dalam melakukan layouting website di laravel.

Praktikum 2 – Penerapan Layouting:

1. Kita buat file controller dengan nama WelcomeController.php

WebLanjut > Week7 > app > Http > Controllers > WelcomeController.php > ...

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class WelcomeController extends Controller
{
    public function index()
    {
        $breadcrumb = (object) [
            'title' => 'Selamat Datang',
            'list'  => ['Home', 'Welcome']
        ];

        $activeMenu = 'dashboard';

        return view('welcome', ['breadcrumb' => $breadcru
    }
}
```

2. Kita buat file pada PWL_POS/resources/views/welcome.blade.php

WebLanjut > Week7 > resources > views > welcome.blade.php > ...

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

@section('content')
    <div class="card">
        <div class="card-header">
            <h3 class="card-title">Halo, apa kabar!!!</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

```
WebLanjut > Week7 > resources > views > layouts > breadcumb.blade.php > ...
<section class="content-header">
    <div class="container-fluid">
        <div class="row mb-2">
            <div class="col-sm-6">
                <h1>{{ $breadcrumb->title }}</h1>
            </div>
            <div class="col-sm-6">
                <ol class="breadcrumb float-sm-right">
                    @foreach ($breadcrumb->list as $key => $value)
                        @if ($key == count($breadcrumb->list) - 1)
                            <li class="breadcrumb-item active">{{ $value }}</li>
                        @else
                            <li class="breadcrumb-item">{{ $value }}</li>
                        @endif
                    @endforeach
                </ol>
            </div>
        </div>
    </div>
</section>
```

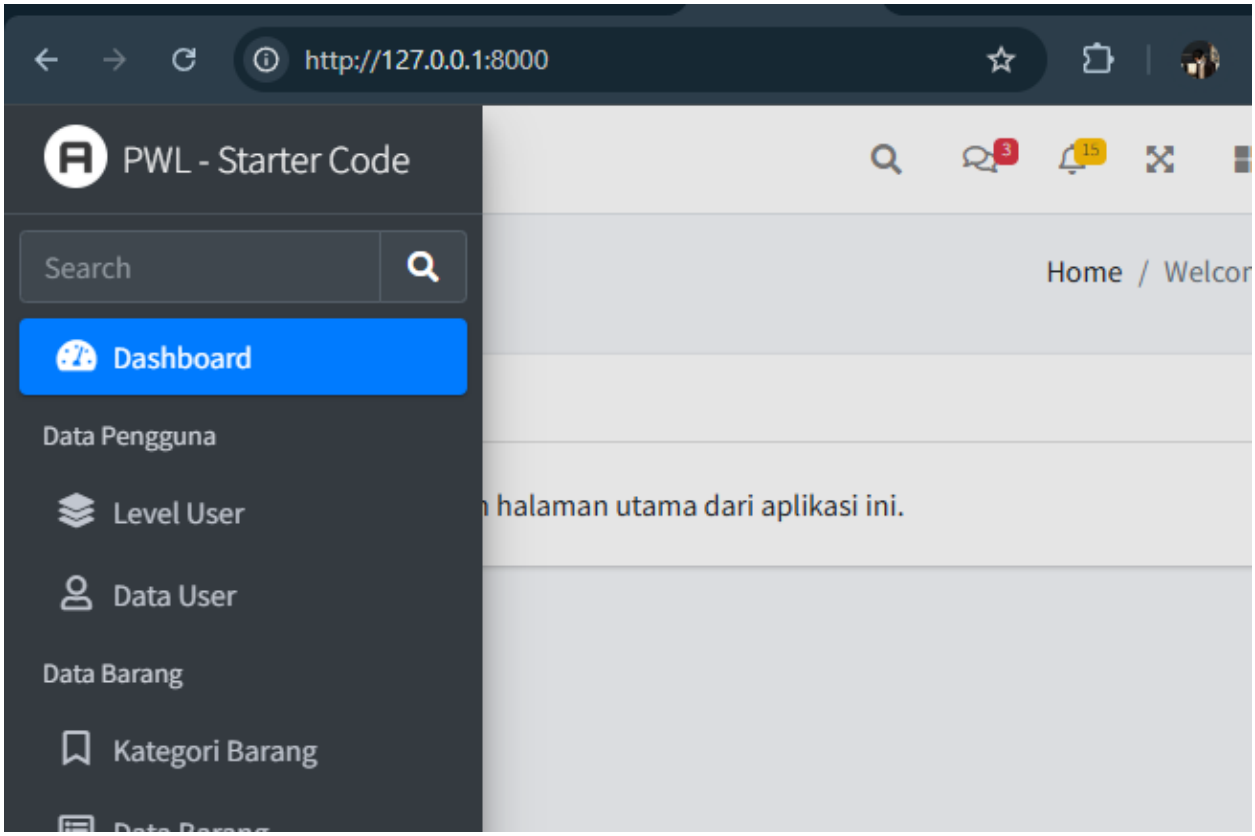
4. Kita modifikasi file PWL_POS/resources/views/layouts/sidebar.blade.php

```
WebLanjut > Week7 > resources > views > layouts > sidebar.blade.php > ...
<div class="sidebar">
    <!-- SidebarSearch Form -->
    <div class="form-inline mt-2">
        <div class="input-group" data-widget="sidebar-search">
            <input class="form-control form-control-sidebar" type="search" placeholder="Searc
            <div class="input-group-append">
                <button class="btn btn-sidebar">
                    <i class="fas fa-search fa-fw"></i>
                </button>
            </div>
        </div>
    </div>

    <!-- Sidebar Menu -->
    <nav class="mt-2">
        <ul class="nav nav-pills nav-sidebar flex-column" data-widget="treeview" role="menu"
            <li class="nav-item">
                <a href="{{ url('/') }}" class="nav-link {{ $activeMenu == 'dashboard' ? 'act
                    <i class="nav-icon fas fa-tachometer-alt"></i>
                    <p>Dashboard</p>
                </a>
            </li>

            <li class="nav-header">Data Pengguna</li>
            <li class="nav-item">
                <a href="{{ url('/level') }}" class="nav-link {{ $activeMenu == 'level' ? 'ac
                    <i class="nav-icon fas fa-layer-group"></i>
                    <p>Level User</p>
                </a>
            </li>
            <li class="nav-item">
                <a href="{{ url('/user') }}" class="nav-link {{ $activeMenu == 'user' ? 'acti
                    <i class="nav-icon far fa-user"></i>
                    <p>Data User</p>
                </a>
            </li>
```

5. Kita tambahkan kode berikut router web.php

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

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



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

```php
use App\Http\Controllers\UserController;
use Illuminate\Support\Facades\Route;
use App\Http\Controllers\WelcomeController;

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

// USER
Route::group(['prefix' => 'user'], function () {
    Route::get('/', [UserController::class, 'index']);          // menampilkan halaman awal us
    Route::post('/list', [UserController::class, 'list']);      // menampilkan data user dalam
    Route::get('/create', [UserController::class, 'create']);   // menampilkan halaman form t
    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 e
    Route::put('/{id}', [UserController::class, 'update']);     // menyimpan perubahan data us
    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

```php
Codeium: Refactor | Explain
class UserController extends Controller
{
    // Menampilkan halaman awal user
    Codeium: Refactor | Explain | ×
    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();

        return view('user.index', ['breadcrumb' => $breadcrumb, 'page' => $page, 'level' => $l
    }
}
```

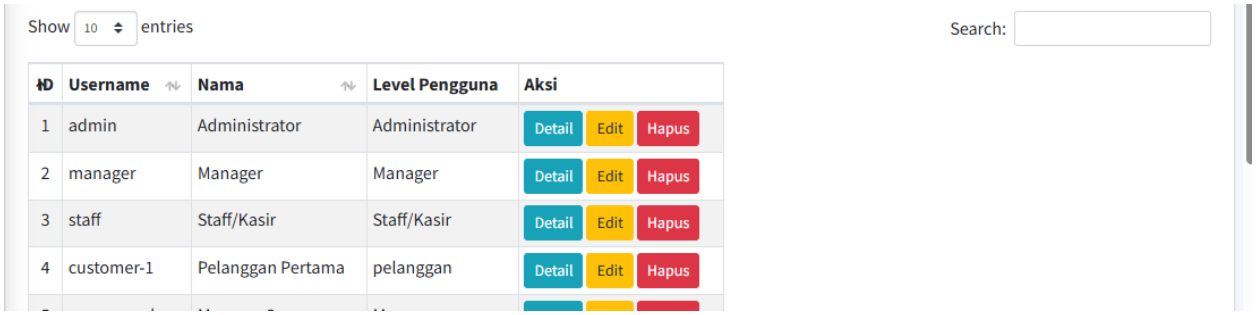
5. Lalu kita buat view pada PWL_POS/resources/views/user/index.blade.php

```
@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
            </div>
        </div>
        <div class="card-body">
            @if (session('success'))
                <div class="alert alert-success">{{ session('success') }}</div>
            @endif
```

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

```
<!-- DataTables & Plugins -->
<script src="{{ asset('adminlte/plugins/datatables/jquery.dataTables.min.js') }}"></scri
<script src="{{ asset('adminlte/plugins/datatables-bs4/js/dataTables.bootstrap4.min.js')
<script src="{{ asset('adminlte/plugins/datatables-responsive/js/dataTables.responsive.mi
<script src="{{ asset('adminlte/plugins/datatables-responsive/js/responsive.bootstrap4.mi
<script src="{{ asset('adminlte/plugins/datatables-buttons/js/dataTables.buttons.min.js'
<script src="{{ asset('adminlte/plugins/datatables-buttons/js/buttons.bootstrap4.min.js'
<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 src="{{ asset('adminlte/plugins/datatables-buttons/js/buttons.print.min.js') }}"
<script src="{{ asset('adminlte/plugins/datatables-buttons/js/buttons.colVis.min.js') }}'
```

7. Untuk bisa menangkap request data untuk datatable, kita buat fungsi list() pada UserController.php seperti berikut
8. Sekarang coba jalankan browser, dan klik menu Data User..!!! perhatikan dan amati apa yang terjadi.



Menampilkan data melalui usercontroller list dan memanggil ajax pada index.blade

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

```
// Menampilkan halaman form tambah user
Codeium: Refactor | Explain | X
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' =>
}
```

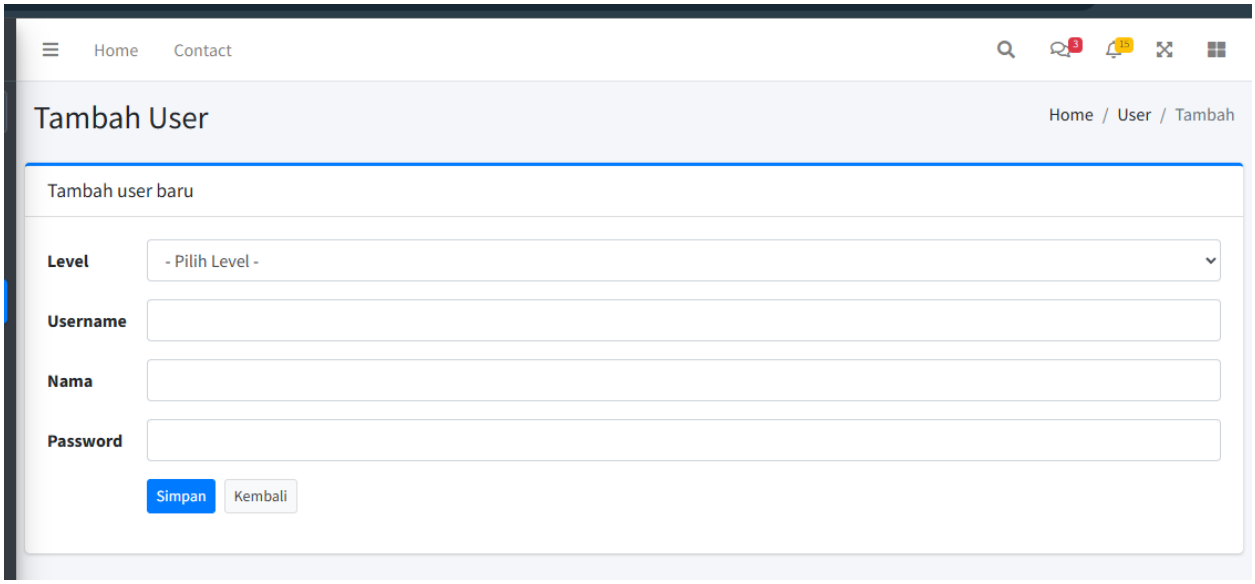
10. Sekarang kita buat form untuk menambah data, kita buat file PWL_POS/resources/views/user/create.blade.php

```
table_user                                   Aa ab .* No results
@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
                            @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">
```

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

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

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



Ketika mengkiik tambah, maka akan diarahkan ke halaman tambah user

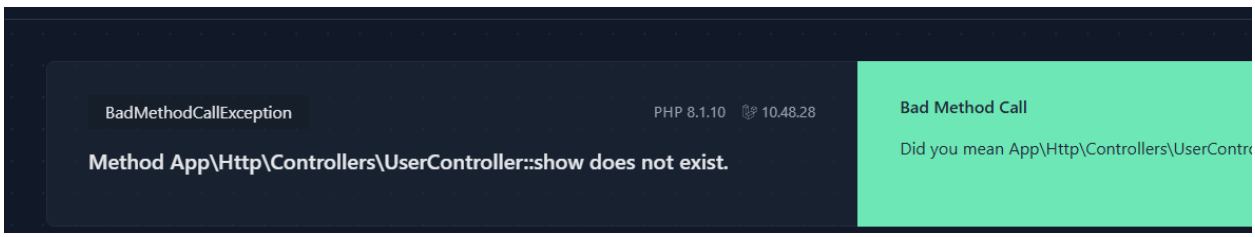
13. Selanjutnya, kita masuk pada bagian menampilkan detail data user (klik tombol) pada halaman user. Route yang bertugas untuk menangkap request detail adalah
14. Jadi kita buat/modifikasi fungsi show() pada UserController.php seperti berikut

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

15. Kemudian kita buat view di PWL_POS/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>
```

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



error

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

```php
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

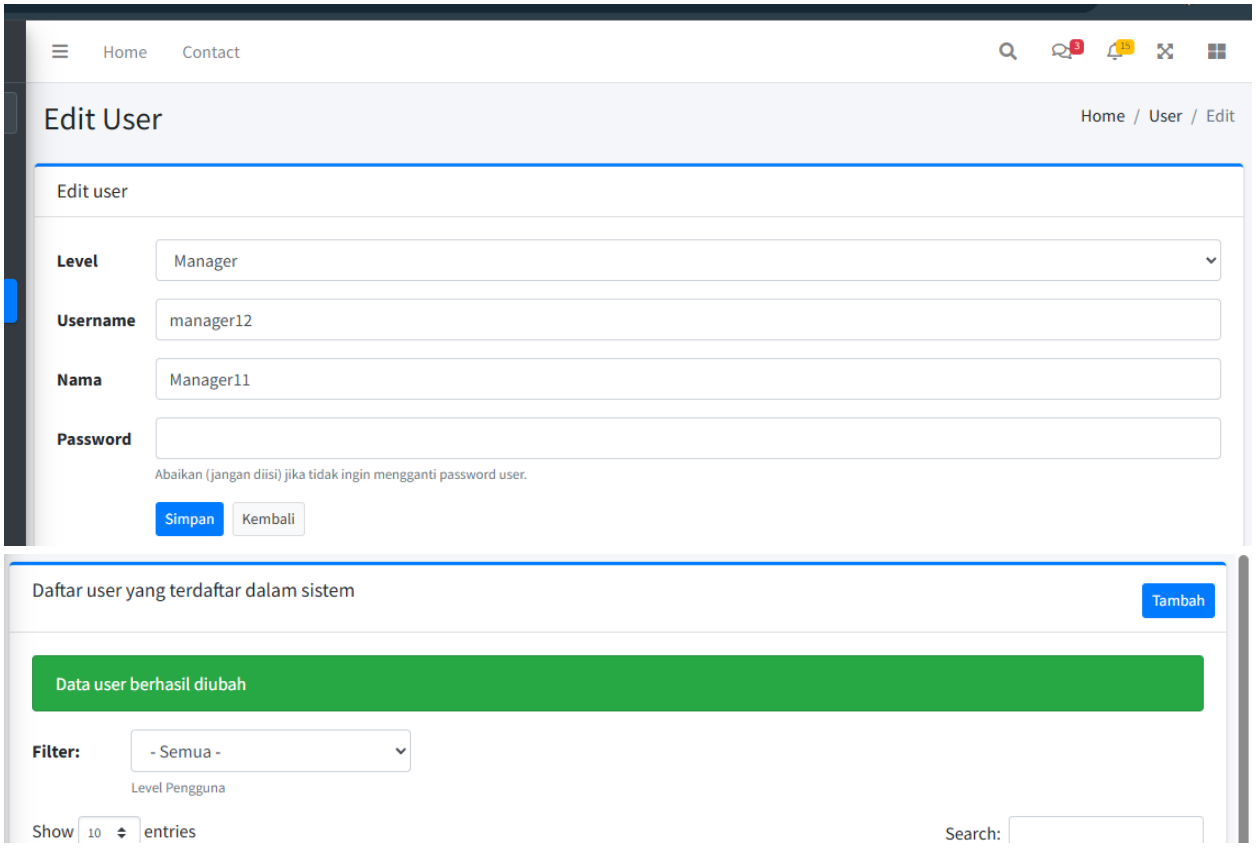
```php
// Menampilkan halaman form edit user
Codeium: Refactor | Explain | ×
public function edit(string $id)
{
    $user = UserModel::find($id);
    $level = LevelModel::all();

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

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

    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
    ]);
```

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

```
VebLanjut > Week7 > resources > views > user > edit.blade.php > ...
1  @extends('layouts.template')
2
3  @section('content')
4
5      <div class="card card-outline card-primary">
6          <div class="card-header">
7              <h3 class="card-title">{{ $page->title }}</h3>
8              <div class="card-tools"></div>
9          </div>
10
11         <div class="card-body">
12             @empty($user)
13                 <div class="alert alert-danger alert-dismissible">
14                     <h5><i class="icon fas fa-ban"></i> Kesalahan!</h5>
15                     Data yang Anda cari tidak ditemukan.
16                 </div>
17                 <a href="{{ url('user') }}" class="btn btn-sm btn-default mt-2">Kembali</a>
18             @else
19                 <form method="POST" action="{{ url('/user/' . $user->user_id) }}" class="form-horizontal">
20                     @csrf
21                     {!! method_field('PUT') !!} <!-- tambahkan baris ini untuk proses edit yang butuh method PUT -->
22
```

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



Data berhasil dirubah, ketika user mengedit nanti data akan terupdate ke database

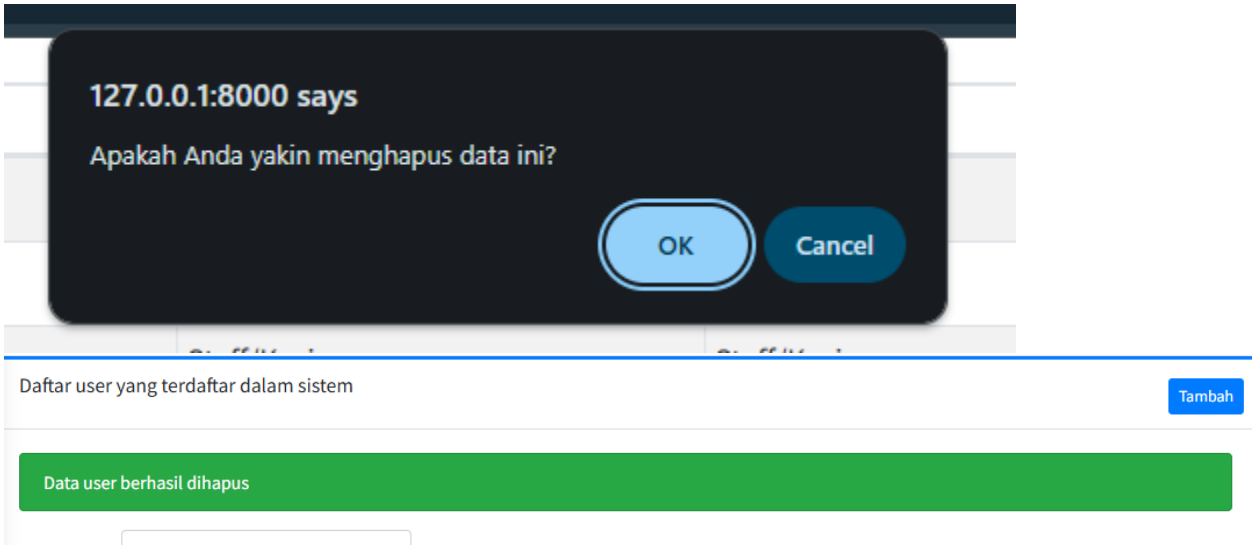
21. Selanjutnya kita akan membuat penanganan untuk tombol hapus. Router web.php yang berfungsi untuk menangkap request hapus dengan method DETELE adalah Route::delete('/{id}', [UserController::class, 'destroy']);

```
Route::delete('/{id}', [UserController::class, 'destroy']); // menghapus data user
```

22. Jadi kita buat fungsi destroy() pada UserController.php

```php
// Menghapus data user
Codeium: Refactor | Explain | X
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_POS/resources/views/user/index.blade.php untuk menambahkan tampilan jika ada pesan error
24. Kemudian jalankan browser untuk menghapus salah satu data user. Amati dan laporkan!



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

Praktikum 4 – Implementasi Filtering Datatables:

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

```php
$level = LevelModel::all(); // filter level user

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

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

```blade
{{-- filter opsi --}}
<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

```
    type: "POST",
    data: function(d) {
        d.level_id = $('#level_id').val();
    }
```

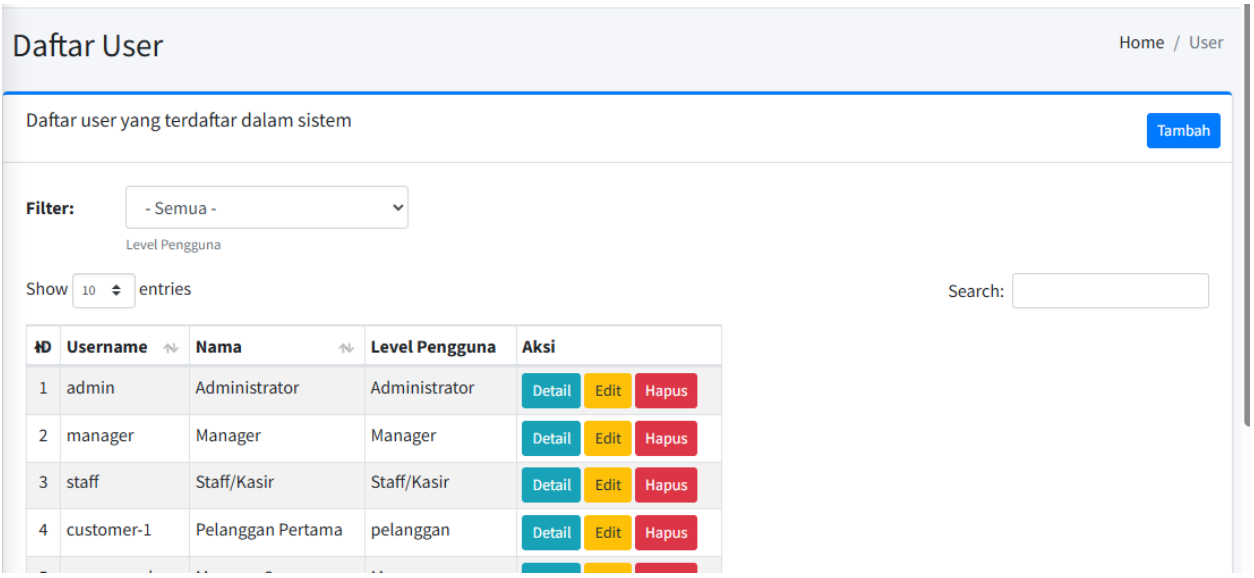
4. Kemudian kita edit pada bagian akhir script @push(‘js’) untuk menambahkan listener jika data filtering dipilih

```
    });

    $('#level_id').on('change', function() {
        dataUser.ajax.reload();
    })
});
```

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

```
    // filter data user by 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.

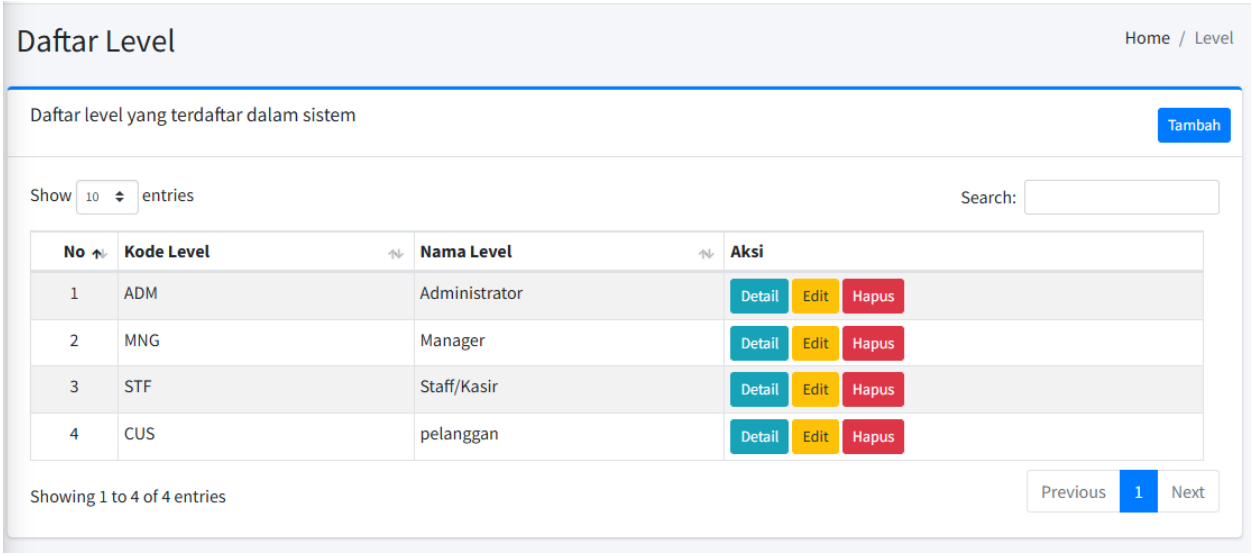
PERTANYAAN

1. Apa perbedaan frontend template dengan backend template?
    - **Frontend template** berfokus pada tampilan
    - **backend template** digunakan untuk mengelola struktur dan logika
2. Apakah layouting itu penting dalam membangun sebuah website?
    - **Layouting** penting dalam karena memastikan tampilan yang rapi, responsif, dan mudah dinavigasi oleh pengguna.
3. Jelaskan fungsi dari komponen laravel blade berikut: @include(), @extend(), @section(), @push(), @yield(), dan @stack()
    - @include(): Memasukkan file Blade lain ke dalam template utama
    - @extends(): Mewarisi tampilan dari template induk
    - @section(): Mendefinisikan bagian konten yang dapat diisi di dalam template induk
    - @push(): Menambahkan skrip atau gaya CSS ke dalam stack tertentu
    - @yield(): Menampilkan konten dari section yang didefinisikan di dalam template turunan
    - @stack(): Menampilkan semua elemen yang telah ditambahkan ke dalam stack tertentu dengan @push()
4. Apa fungsi dan tujuan dari variable $activeMenu?
    - Digunakan untuk menandai menu yang sedang aktif pada saat web sedang diakses

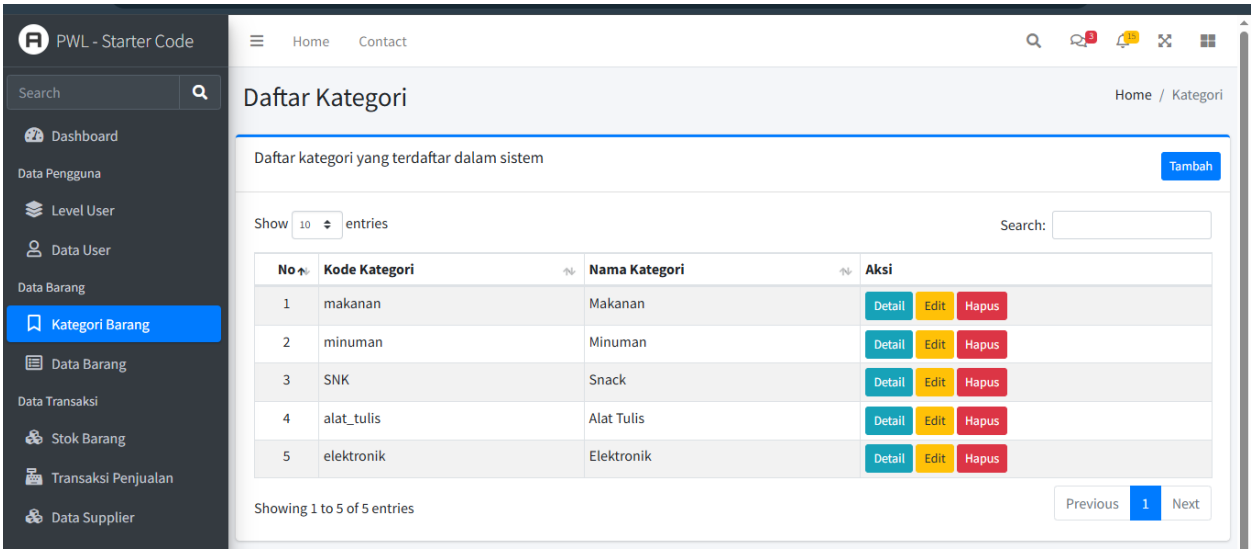
TUGAS

Implementasikan menu yang belum ada di laravel starter code ini sesuai dengan Studi Kasus Point of Sales Sederhana. Silahkan terapkan kode di laravel starter code untuk menu-menu yang sesuai dengan menu di samping ini.

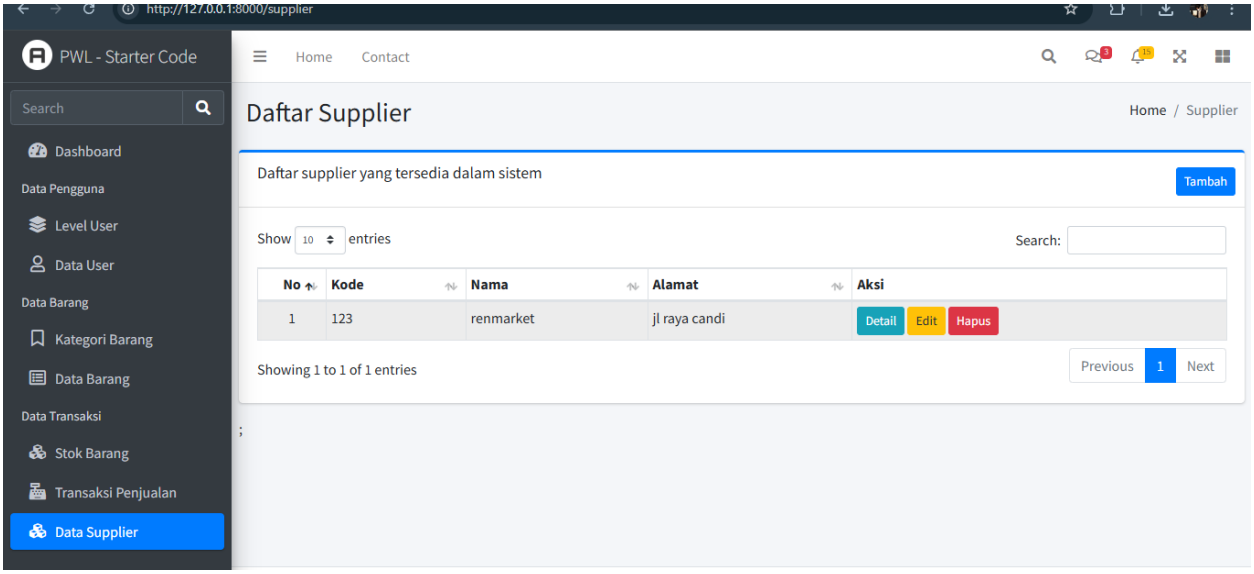
Level



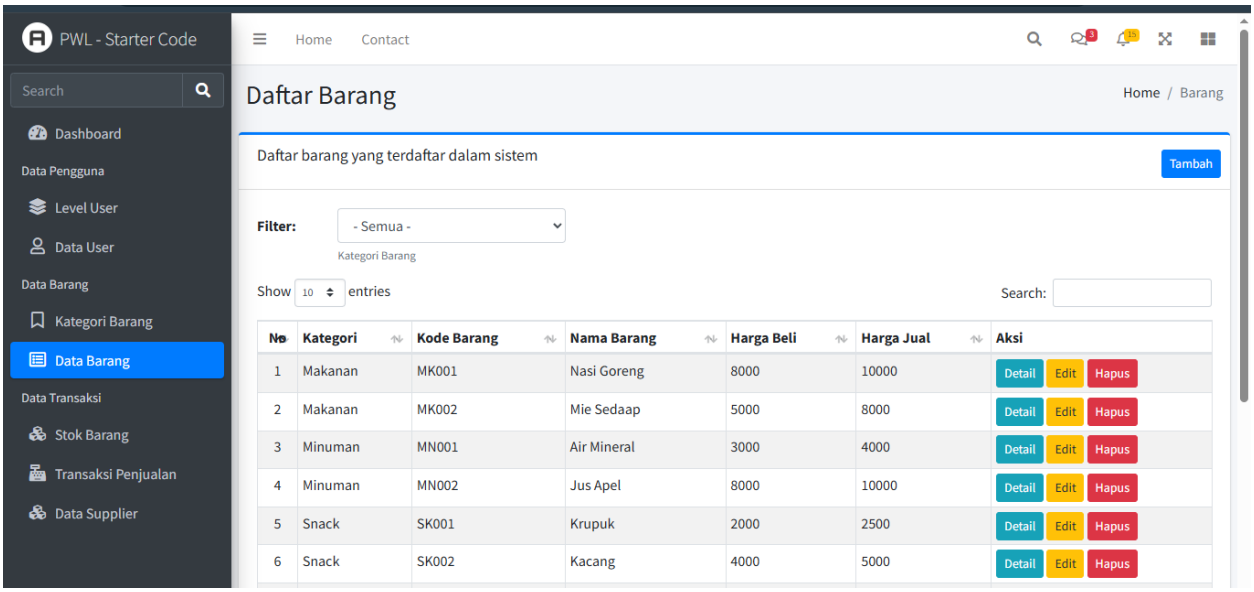
Kategori



Supplier


http://127.0.0.1:8000/supplier
PWL - Starter Code
Home
Contact
Search
Dashboard
Data Pengguna
Level User
Data User
Data Barang
Kategori Barang
Data Barang
Data Transaksi
Stok Barang
Transaksi Penjualan
Data Supplier
Daftar Supplier
Home / Supplier
Daftar supplier yang tersedia dalam sistem
Tambah
Show 10 entries
Search:
No
Kode
Nama
Alamat
Aksi
1
123
renmarket
jl raya candi
Detail
Edit
Hapus
Showing 1 to 1 of 1 entries
Previous
1
Next

Barang


PWL - Starter Code
Home
Contact
Search
Dashboard
Data Pengguna
Level User
Data User
Data Barang
Kategori Barang
Data Barang
Data Transaksi
Stok Barang
Transaksi Penjualan
Data Supplier
Daftar Barang
Home / Barang
Daftar barang yang terdaftar dalam sistem
Tambah
Filter:
- Semua -
Kategori Barang
Show 10 entries
Search:
No
Kategori
Kode Barang
Nama Barang
Harga Beli
Harga Jual
Aksi
1 Makanan MK001 Nasi Goreng 8000 10000
2 Makanan MK002 Mie Sedaap 5000 8000
3 Minuman MN001 Air Mineral 3000 4000
4 Minuman MN002 Jus Apel 8000 10000
5 Snack SK001 Krupuk 2000 2500
6 Snack SK002 Kacang 4000 5000
Detail
Edit
Hapus