

PEMROGRAMAN WEB LANJUT

“Laravel Starter Code”



Kelas : TI-2H

Disusun Oleh :

Fanesabhirawaning Sulistyo

2241720027 (15)

PROGRAM STUDI D-IV TEKNIK INFORMATIKA

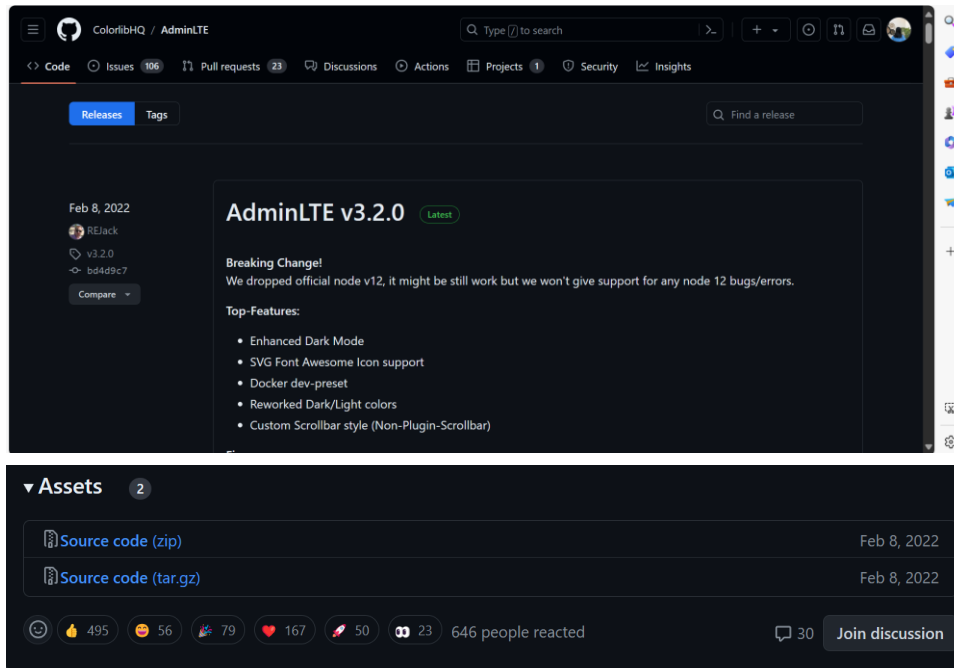
JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

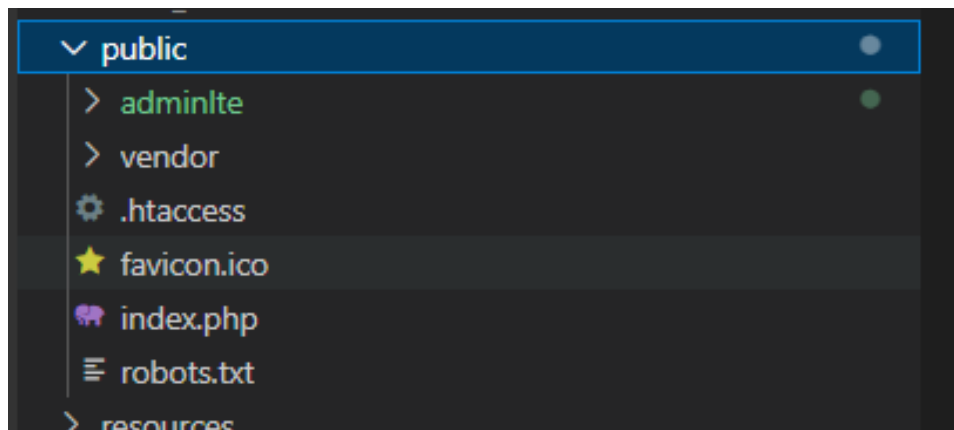
Jl. Soekarno Hatta No.9, Jatimulyo, Kec. Lowokwaru, Kota Malang, Jawa Timur
65141

Praktikum 1 – Layouting AdminLTE:

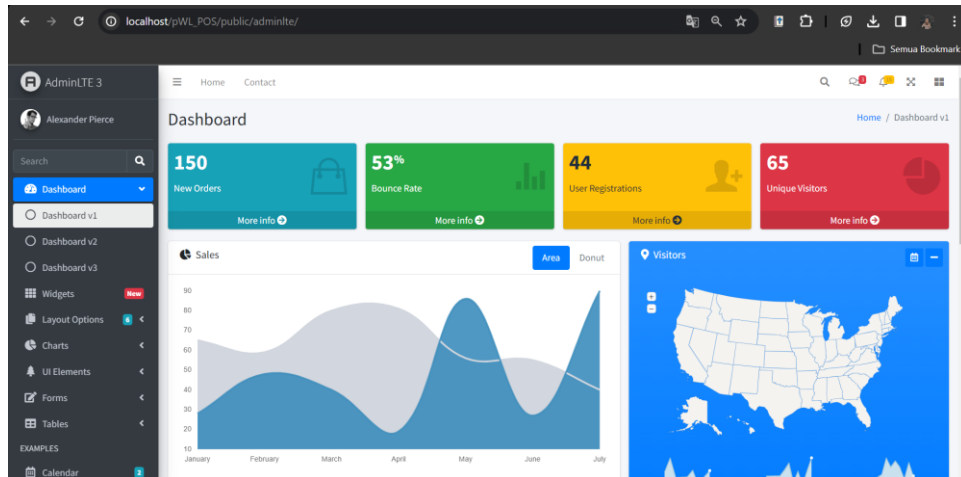
1. Kita download **AdminLTE v3.2.0** yang rilis pada 8 Feb 2022



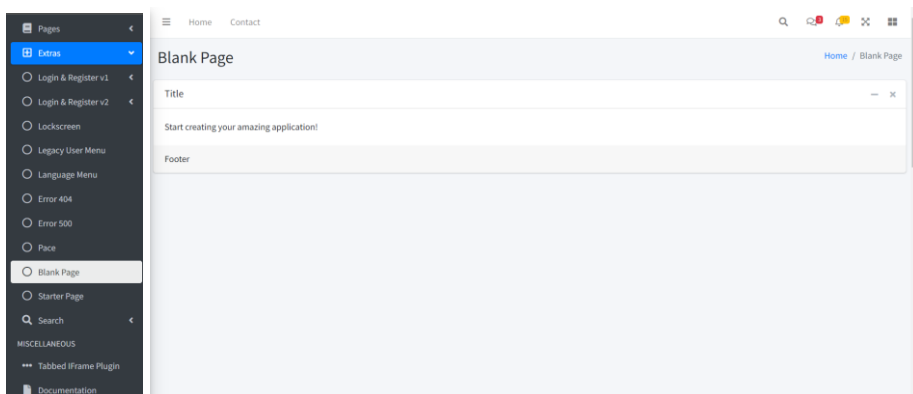
2. Setelah kita berhasil download, kita ekstrak file yang sudah di download ke folder project `PWL_POS/public`, kemudian kita **rename** folder cukup menjadi `adminlte`



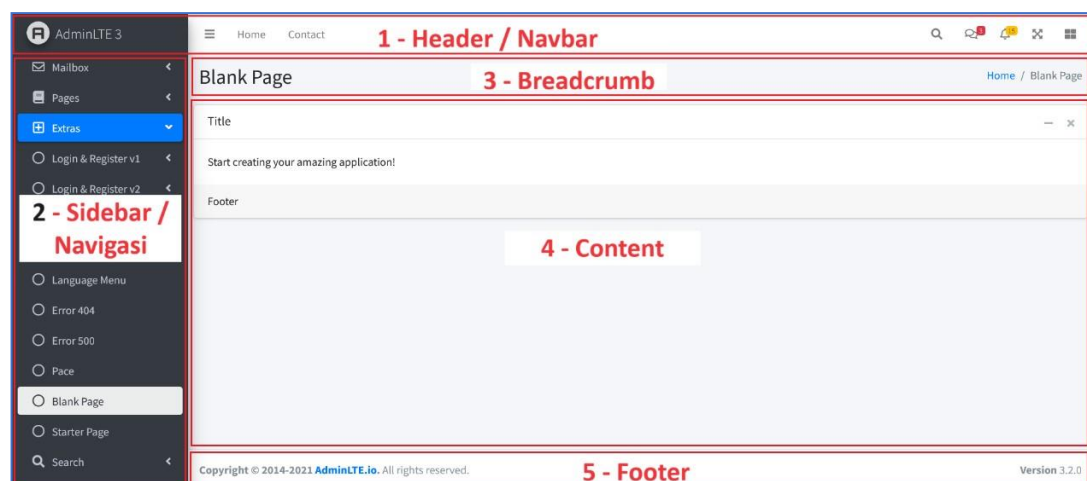
3. Selanjutnya kita buka di browser dengan alamat http://localhost/PWL_POS/public/adminlte maka akan muncul tampilan seperti berikut



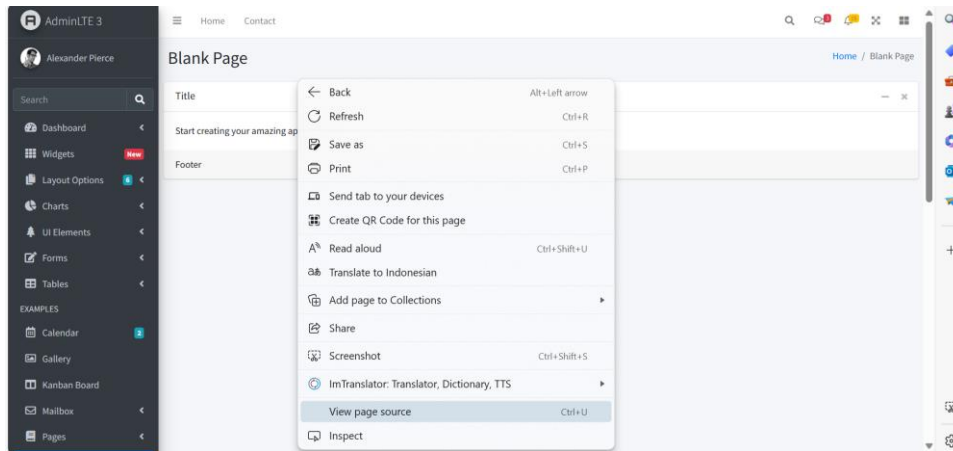
4. Kita klik menu Extras > Blank Page, page inilah yang akan menjadi dasar web template



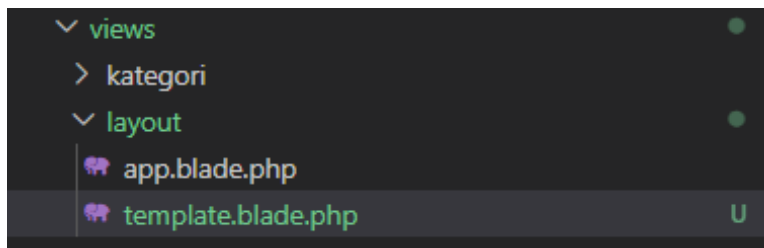
5. Dari sini kita bisa melakukan layouting halaman Blank Page ini menjadi 4 element seperti pada gambar berikut



6. Selanjutnya kita klik kanan halaman **Blank Page** dan klik *view page source*



7. Selanjutnya kita copy page source dari halaman **Blank Page**, 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

```
resources > views > layout > template.blade.php > html > body > hold-transition.sidebar-mini > div.wrapper > nav.main-header.navbar.navbar-expand.navbar
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>AdminLTE 3 | Blank Page</title>
7
8 <!-- Google Font: Source Sans Pro -->
9 <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Source+Sans+Pro:300,400,400i,700&display=fallback">
10 <!-- Font Awesome -->
11 <link rel="stylesheet" href="../../plugins/fontawesome-free/css/all.min.css">
12 <!-- Theme style -->
13 <link rel="stylesheet" href="../../dist/css/adminlte.min.css">
14 </head>
```

Menjadi

```
resources > views > layout > template.blade.php > html > head > link
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>{{config('app.name', 'PWL Laravel Starter Code')}}</title>
7
8 <!-- Google Font: Source Sans Pro -->
9 <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Source+Sans+Pro:300,400,400i,700&display=fallback">
10 <!-- Font Awesome -->
11 <link rel="stylesheet" href="{{asset('adminlte/plugins/fontawesome-free/css/all.min.css')}}">
12 <!-- Theme style -->
13 <link rel="stylesheet" href="{{asset('adminlte/dist/css/adminlte.min.css')}}">
14 </head>
```

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

```
resources > views > layout > template.blade.php > html > body.hold-transition.sidebar-mini > div.wrapper
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>{{config('app.name','PWL Laravel Starter Code')}}</title>
7
8 <!-- Google Font: Source Sans Pro -->
9 <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Source+Sans+Pro:300,400,400i,700&display=fallback">
10 <!-- Font Awesome -->
11 <link rel="stylesheet" href="{{asset('adminlte/plugins/fontawesome-free/css/all.min.css')}}">
12 <!-- Theme style -->
13 <link rel="stylesheet" href="{{asset('adminlte/dist/css/adminlte.min.css')}}">
14 </head>
15 <body class="hold-transition sidebar-mini">
16 <!-- Site wrapper -->
17 <div class="wrapper">
18 <!-- Navbar -->
19 @include('layout.header')
20
21 <!-- Main Sidebar Container -->
22 <aside class="main-sidebar sidebar-dark-primary elevation-4">
23 <!-- Brand Logo -->
24 <a href="../../../index3.html" class="brand-link">
25 
26 <span class="brand-text font-weight-light">AdminLTE 3</span>
27 </a>
28
29 <!-- Sidebar -->
30 <div class="sidebar">
```

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**

```
17 <div class="wrapper">
18 <!-- Navbar -->
19 @include('layout.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>
29
30 <!-- Sidebar -->
31 <div class="sidebar">
```

Menjadi

```
17 <div class="wrapper">
18 <!-- Navbar -->
19 @include('layout.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 <div class="sidebar">
```

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

```

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('layout.sidebar')
32 <!-- /.sidebar -->
33 </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

```

82 </section>
83 <!-- /.content -->
84 </div>
85 <!-- /.content-wrapper -->
86 @include('layout.footer')
87 </div>
88 <!-- ./wrapper -->

```

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

```

91 <script src="../../plugins/jquery/jquery.min.js"></script>
92 <!-- Bootstrap 4 -->
93 <script src="../../plugins/bootstrap/js/bootstrap.bundle.min.js"></script>
94 <!-- AdminLTE App -->
95 <script src="../../dist/js/adminlte.min.js"></script>
96 <!-- AdminLTE for demo purposes -->
97 <script src="../../dist/js/demo.js"></script>
98 </body>
99 </html>
100

```

Menjadi

```

91 <script src="{{asset('adminlte/plugins/jquery/jquery.min.js')}}"></script>
92 <!-- Bootstrap 4 -->
93 <script src="{{asset('adminlte/plugins/bootstrap/js/bootstrap.bundle.min.js')}}"></script>
94 <!-- AdminLTE App -->
95 <script src="{{asset('adminlte/dist/js/adminlte.min.js')}}"></script>
96 <!-- AdminLTE for demo purposes -->
97 <script src="{{asset('adminlte/dist/js/demo.js')}}"></script>
98 </body>
99 </html>

```

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

```

30     <!-- Sidebar -->
31     @include('layout.sidebar')
32     <!-- /.sidebar -->
33 </aside>
34
35 <!-- Content Wrapper. Contains page content -->
36 <div class="content-wrapper">
37
38     @include('layout.breadcrumb')
39
40     <!-- Main content -->
41     <section class="content">
42
43
44     </section>
45     <!-- /.content -->
46 </div>

```

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

```

15 <body class="hold-transition sidebar-mini">
16 <!-- Site wrapper -->
17 <div class="wrapper">
18     <!-- Navbar -->
19     @include('layout.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('layout.sidebar')
32         <!-- /.sidebar -->
33     </aside>
34
35     <!-- Content Wrapper. Contains page content -->
36     <div class="content-wrapper">
37
38         @include('layout.breadcrumb')
39
40         <!-- Main content -->
41         <section class="content">
42             @yield('content')
43         </section>
44         <!-- /.content -->
45     </div>
46     <!-- /.content-wrapper -->
47     @include('layout.footer')
48 </div>
49 <!-- ./wrapper -->

```

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

Praktikum 2 – Penerapan Layouting:

Sekarang kita akan mencoba melakukan penerapan terhadap layouting yang sudah kita lakukan.

1. Kita buat file controller dengan nama `WelcomeController.php`

```
app > Http > Controllers > WelcomeController.php > ...
1  <?php
2  namespace App\Http\Controllers;
3
4  use Illuminate\Http\Request;
5
6  class WelcomeController extends Controller
7  {
8      public function index()
9      {
10         $breadcrumbs = (object) [
11             'title' => 'Selamat Datang',
12             'list' => ['Home', 'Welcome']
13         ];
14
15         $activeMenu = 'dashboard';
16
17         return view('welcome', ['breadcrumbs' => $breadcrumbs, 'activeMenu' => $activeMenu]);
18     }
19 }
```

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

```
resources > views > welcome.blade.php > ...
1  @extends('layout.template')
2
3  @section('content')
4
5      <div class="card">
6          <div class="card-header">
7              <h3 class="card-title">Halo, apakabar!!!</h3>
8              <div class="card-tools"></div>
9          </div>
10         <div class="card-body">
11             Selamat dtang semua, ini adalah halaman utama dari apikasi ini.
12         </div>
13     </div>
14 @endsection
```

3. Kita modifikasi file `PWL_POS/resources/views/layouts/breadcrumb.blade.php`


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

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

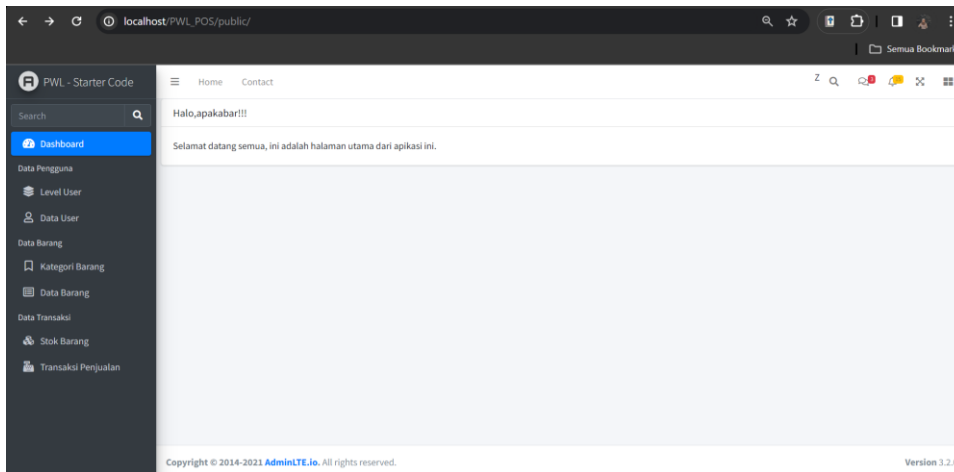
```
resources > views > layout > sidebar.blade.php > div.sidebar > nav.mt-2 > ul.nav.nav-pills.nav-sidebar.flex-column > li.nav-item > a.nav-link.{{ $activeMenu == 'dashboard' ? 'active' : '' }}
1 <div class="sidebar">
2     <!-- Sidebar Search Form -->
3     <div class="form-inline mt-2">
4         <div class="input-group" data-widget="sidebar-search">
5             <input class="form-control form-control-sidebar" type="search" placeholder="Search" aria-label="Search">
6             <div class="input-group-append">
7                 <button class="btn btn-sidebar">
8                     <i class="fas fa-search fa-fw"></i>
9                 </button>
10            </div>
11        </div>
12    </div>
13    <!-- Sidebar Menu -->
14    <nav class="mt-2">
15        <ul class="nav nav-pills nav-sidebar flex-column" data-widget="treeview" role="menu" data-accordion="false">
16            <li class="nav-item">
17                <a href="{{ url('/') }}" class="nav-link {{ ($activeMenu == 'dashboard') ? 'active' : '' }}">
18                    <i class="nav-icon fas fa-tachometer-alt"></i>
19                    <p>Dashboard</p>
20                </a>
21            </li>
22            <li class="nav-item">
23                <a href="{{ url('/level') }}" class="nav-link {{ ($activeMenu == 'level') ? 'active' : '' }}">
24                    <i class="nav-icon fas fa-layer-group"></i>
25                    <p>Level User</p>
26                </a>
27            </li>
28            <li class="nav-item">
29                <a href="{{ url('/user') }}" class="nav-link {{ ($activeMenu == 'user') ? 'active' : '' }}">
30                    <i class="nav-icon fas fa-user"></i>
31                    <p>Data User</p>
32                </a>
33            </li>
34        </ul>
35    </nav>
36 </div>
```

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



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

```
routes > web.php > ...
1  <?php
2
3  use App\Http\Controllers\LevelController;
4  use App\Http\Controllers\KategoriController;
5  use App\Http\Controllers\POSController;
6  use App\Http\Controllers\UserController;
7  use App\Http\Controllers\WelcomeController;
8  use Illuminate\Support\Facades\Route;
9
10 /*
11 |-----|
12 | Web Routes
13 |-----|
14 |
15 | Here is where you register web routes for your application. These
16 | routes are loaded by the RouteServiceProvider and all of them will
17 | be assigned to the "web" middleware group. Make something great!
18 |
19 */
20
21 // Route::get('/', function () {
22 //     return view('welcome');
23 // });
24 Route::get('/', [WelcomeController::class, 'index']);
25
26 Route::group(['prefix' => 'user'], function () {
27     Route::get('/', [UserController::class, 'index']); // halaman awal user
28     Route::post('/list', [UserController::class, 'list']); // data user dalam bentuk json untuk datatables
29     Route::get('/create', [UserController::class, 'create']); // halaman form tambah user
30     Route::post('/', [UserController::class, 'store']); // simpan data user baru
31     Route::get('/{id}', [UserController::class, 'show']); // detail user
32     Route::get('/{id}/edit', [UserController::class, 'edit']); // halaman form edit user
33     Route::put('/{id}', [UserController::class, 'update']); // simpan perubahan data user
34     Route::delete('/{id}', [UserController::class, 'destroy']); // hapus data user
```

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

```
app > Http > Controllers > UserController.php > UserController > tambah_simpan
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use App\Http\Requests\StorePostRequest;
6  use App\Models\UserModel;
7  use Illuminate\Http\Request;
8  use Illuminate\Support\Facades\Hash;
9
10 class UserController extends Controller
11 {
12     public function index()
13     {
14         $breadcrumb = (object)[
15             'title' => 'Daftar User',
16             'list' => ['Home', 'User']
17         ];
18         $page = (object)[
19             'title' => 'Daftar user yang terdaftar dalam sistem'
20         ];
21         $activeMenu = 'user'; //set menu yg sdg aktif
22         return view('user.index', ['breadcrumb' => $breadcrumb, 'page' => $page, 'activeMenu' => $activeMenu]);
23     }
24 }
```

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

```
resources > views > user > index.blade.php > script > <function> > dataUser
1  @extends('layouts.template')
2
3  @section('content')
4      <div class="card card-outline card-primary">
5          <div class="card-header">
6              <h3 class="card-title">{{ $page->title }}</h3>
7              <div class="card-tools">
8                  <a class="btn btn-sm btn-primary mt-1" href="{{ url('user/create') }}">Tambah</a>
9              </div>
10          </div>
11          <div class="card-body">
12              @if (session('success'))
13                  <div class="alert alert-success">{{ session('success') }}</div>
14              @endif
15              @if (session('error'))
16                  <div class="alert alert-danger">{{ session('error') }}</div>
17              @endif
18              <table class="table table-bordered table-striped table-hover table-sm" id="table_user">
19                  <thead>
20                      <tr>
21                          <th>ID</th>
22                          <th>Username</th>
23                          <th>Nama</th>
24                          <th>Level</th>
25                          <th>Pegawai</th>
26                          <th>Aksi</th>
27                      </tr>
28                  </thead>
29                  <tbody>
30                      <tr>
31                          <td></td>
32                      </tr>
33                  </tbody>
34              </table>
35          </div>
36      </div>
37  @endsection
```

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

```

resources > views > layout > template.blade.php > html > body:hold-transition.sidebar-mini > script
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="utf-8">
5 <meta name="viewport" content="width=device-width, initial-scale=1">
6 <title>{{config('app.name', 'PWL Laravel Starter Code')}}</title>
7
8 <meta name="csrf-token" content="{{ csrf_token() }}">
9
10 <!-- Google Font: Source Sans Pro -->
11 <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Source+Sans+Pro:300,400,400i,700&display=fallback">
12 <!-- Font Awesome -->
13 <link rel="stylesheet" href="{{asset('adminlte/plugins/fontawesome-free/css/all.min.css')}}">
14 <!-- DataTables -->
15 <link rel="stylesheet" href="{{asset('adminlte/plugins/datatables-bs4/css/dataTables.bootstrap4.min.css')}}">
16 <link rel="stylesheet" href="{{asset('adminlte/plugins/datatables-responsive/css/responsive.bootstrap4.min.css')}}">
17 <link rel="stylesheet" href="{{asset('adminlte/plugins/datatables-buttons/css/buttons.bootstrap4.min.css')}}">
18 <!-- Theme style -->
19 <link rel="stylesheet" href="{{asset('adminlte/dist/css/adminlte.min.css')}}">
20
21 @stack('css') <!-- Digunakan untuk memanggil custom css dari perintah push(css) pada masing2 view -->
22 </head>
23 <body class="hold-transition sidebar-mini">
24 <!-- Site wrapper -->
25 <div class="wrapper">
26 <!-- Navbar -->
27 @include('layout.header')
28 <!-- /.navbar -->
29
30 <!-- Main Sidebar Container -->
31 <aside class="main-sidebar sidebar-dark-primary elevation-4">
32 <!-- Brand Logo -->
33 <a href="{{url('/')}}" class="brand-link">
34 

```

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

```

app > Http > Controllers > UserController.php > UserController > hapus
74 public function list(Request $request)
75 {
76     $users = UserModel::select('user_id', 'username', 'nama', 'level_id')
77         ->with('level')
78         ->get();
79
80     return DataTables::of($users)
81         ->addIndexColumn()
82         ->addColumn('aksi', function ($user) {
83             $btn = '<a href="'.url('/user/'. $user->user_id).'" class="btn btn-info btn-sm"><i class="fas fa-eye"></i></a> ';
84             $btn .= '<a href="'.url('/user/'. $user->user_id . '/edit').'" class="btn btn-warning btn-sm"><i class="fas fa-pencil-alt"></i></a>';
85             $btn .= '<form class="d-inline-block" method="POST" action="'.url('/user/'. $user->user_id).'">
86                 <csrf_field() . method_field("DELETE")
87                 <button type="submit" class="btn btn-danger btn-sm" onclick="return confirm('\Apakah Anda yakin menghapus data ini?');"><i
88             return $btn;
89
90         })
91         ->rawColumns(['aksi'])
92         ->make(true);
93     }
94 }

```

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

The screenshot shows a web application interface. On the left is a sidebar with a menu. The 'Data User' option is highlighted. The main area is titled 'Daftar User' and contains a table of users. The table has 7 rows of data. Each row has columns for ID, Username, Nama, Level Pengguna, and Aksi. The Aksi column contains three icons: an eye (view), a pencil (edit), and a trash can (delete). Below the table, there is a pagination bar showing 'Showing 1 to 7 of 7 entries' and buttons for 'Previous', '1', and 'Next'.

ID	Username	Nama	Level Pengguna	Aksi
1	admin	Administrator	Administrator	
2	manager	Manager	Manager	
3	staff	Staff/Kasir	Staff/Kasir	
4	manager_dua	Manager 2	Manager	
5	manager22	Manager Dua Dua	Manager	
6	manager33	Manager Tiga Tiga	Manager	
7	manager55	Manager55	Manager	

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

```
95 //menampilkan halaman form tambah user
96 public function create()
97 {
98     $breadcrumb = (object) [
99         'title' => 'Tambah User',
100         'list' => ['Home', 'User', 'Tambah']
101     ];
102     $page = (object) [
103         'title' => 'Tambah user baru'
104     ];
105
106     $level = LevelModel::all();
107     $activeMenu = 'user'; //set menu yg sdg aktif
108
109     return view('user.create', ['breadcrumb' => $breadcrumb, 'page' => $page, 'level' => $level, 'activeMenu' => $activeMenu]);
110 }
```

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

```
resources > views > user > create.blade.php > div.card.card-outline.card-primary > div.card-body > form.form-horizontal > div.form-group.row > label.col-1.control-label.col-form-label
1 @extends('layout.template')
2
3 @section('content')
4 <div class="card card-outline card-primary">
5     <div class="card-header">
6         <h3 class="card-title">{{ $page->title }}</h3>
7         <div class="card-tools"></div>
8     </div>
9     <div class="card-body">
10         <form method="POST" action="{{ url('user') }}" class="form-horizontal">
11             @csrf
12             <div class="form-group row">
13                 <label class="col-1 control-label col-form-label">Level</label>
14                 <div class="col-11">
15                     <select class="form-control" id="level_id" name="level_id" required>
16                         <option value="">- Pilih Level -</option>
17                         @foreach($level as $item)
18                             <option value="{{ $item->level_id }}">{{ $item->level_nama }}</option>
19                         @endforeach
20                     </select>
21                     @error('level_id')
22                         <small class="form-text text-danger">{{ $message }}</small>
23                     @enderror
24                 </div>
25             </div>
26             <div class="form-group row">
27                 <label class="col-1 control-label col-form-label">Username</label>
28                 <div class="col-11">
29                     <input type="text" class="form-control" id="username" name="username" value="{{ old('username') }}" required>
30                     @error('username')
31                         <small class="form-text text-danger">{{ $message }}</small>
32                     @enderror
33                 </div>
34             </div>
35         </form>
36     </div>
37 </div>
```

11. Kemudian untuk bisa *menng-handle* 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' => 'required|string|min:3|unique:m_user,username',
        'nama' => 'required|string|max:100',
        'password' => 'required|string|min:5',
        'level_id' => 'required|integer'
    ]);

    UserModel::create([
        'username' => $request->username,
        'nama' => $request->nama,
        'password' => bcrypt($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..!!!

The screenshot shows a web application interface for adding a new user. The left sidebar contains a menu with options like Dashboard, Data Pengguna, Level User, Data User (highlighted), Data Barang, Kategori Barang, Data Barang, Data Transaksi, Stok Barang, and Transaksi Penjualan. The main content area is titled 'Tambah User' and has a breadcrumb 'Home / User / Tambah'. The form itself is titled 'Tambah user baru' and includes the following fields:

- Level:** A dropdown menu currently showing 'Staff/Kasir'.
- Username:** A text input field containing 'safira123'.
- Nama:** A text input field containing 'Safira'.
- Password:** A password input field with masked characters.

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

The screenshot shows the 'Daftar User' (User List) page. It features a green success message at the top: 'Data user berhasil disimpan'. Below this is a table listing the registered users. The table has the following columns: ID, Username, Nama, Level Pengguna, and Aksi. The data rows are as follows:

ID	Username	Nama	Level Pengguna	Aksi
1	admin	Administrator	Administrator	[Icons]
2	manager	Manager	Manager	[Icons]
3	staff	Staff/Kasir	Staff/Kasir	[Icons]
4	manager_dua	Manager 2	Manager	[Icons]
5	manager22	Manager Dua Dua	Manager	[Icons]
6	manager33	Manager Tiga Tiga	Manager	[Icons]
7	manager55	Manager55	Manager	[Icons]
8	safira123	Safira	Staff/Kasir	[Icons]

The 'Aksi' column contains icons for viewing, editing, and deleting each user record. The page also includes a search bar and a 'Show 10 entries' dropdown.

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

```

24 Route::get('/', [WelcomeController::class, 'index']);
25
26 Route::group(['prefix' => 'user'], function () {
27     Route::get('/', [UserController::class, 'index']); // halaman awal user
28     Route::post('/list', [UserController::class, 'list']); // data user dalam bentuk json untuk datatables
29     Route::get('/create', [UserController::class, 'create']); // halaman form tambah user
30     Route::post('/', [UserController::class, 'store']); // simpan data user baru
31     Route::get('/{id}', [UserController::class, 'show']); // detail user
32     Route::get('/{id}/edit', [UserController::class, 'edit']); // halaman form edit user
33     Route::put('/{id}', [UserController::class, 'update']); // simpan perubahan data user
34     Route::delete('/{id}', [UserController::class, 'destroy']); // hapus data user
35 });
  
```

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

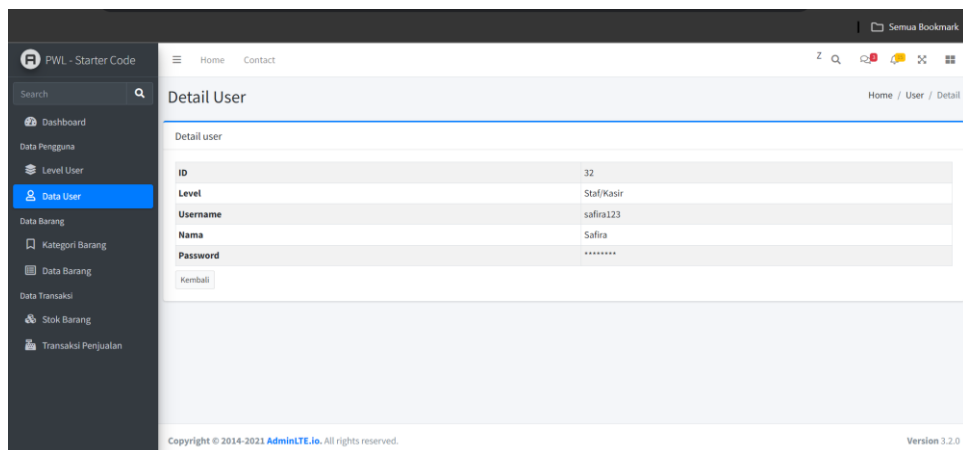
```
//menampilkan detail user
public function show(string $id)
{
    $user = UserModel::with('level')->find($id);
    $breadcrumb = (object) [
        'title' => 'Detail User',
        'list' => ['Home', 'User', 'Detail']
    ];
    $page = (object) [
        'title' => 'Detail user'
    ];
    $activeMenu = 'user'; //set menu yg sdg aktif

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

15. Kemudian kita buat *view* di `PWL_POS/resources/views/user/show.blade.php`

```
resources > views > user > show.blade.php > ...
1 @extends('layout.template')
2
3 @section('content')
4 <div class="card card-outline card-primary">
5     <div class="card-header">
6         <h3 class="card-title">{{ $page->title }}</h3>
7         <div class="card-tools"></div>
8     </div>
9     <div class="card-body">
10         @empty($user)
11             <div class="alert alert-danger alert-dismissible">
12                 <h5><i class="icon fas fa-ban"></i> Kesalahan!</h5>
13                 Data yang Anda cari tidak ditemukan.
14             </div>
15         @else
16             <table class="table table-bordered table-striped table-hover table-sm">
17                 <tr>
18                     <th>ID</th>
19                     <td>{{ $user->user_id }}</td>
20                 </tr>
21                 <tr>
22                     <th>Level</th>
23                     <td>{{ $user->level->level_nama }}</td>
24                 </tr>
25                 <tr>
26                     <th>Username</th>
27                     <td>{{ $user->username }}</td>
28                 </tr>
29                 <tr>
30                     <th>Nama</th>
31                     <td>{{ $user->nama }}</td>
32                 </tr>
33                 <tr>
34                     <th>Password</th>
```

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!!!



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

```
26 Route::group(['prefix' => 'user'], function () {
27     Route::get('/', [UserController::class, 'index']); // halaman awal user
28     Route::post('/list', [UserController::class, 'list']); // data user dalam bentuk json untuk datatables
29     Route::get('/create', [UserController::class, 'create']); // halaman form tambah user
30     Route::post('/', [UserController::class, 'store']); // simpan data user baru
31     Route::get('/{id}', [UserController::class, 'show']); // detail user
32     Route::get('/{id}/edit', [UserController::class, 'edit']); // halaman form edit user
33     Route::put('/{id}', [UserController::class, 'update']); // simpan perubahan data user
34     Route::delete('/{id}', [UserController::class, 'destroy']); // hapus data user
35 });
```

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

```
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 1'];
    $activeMenu = 'user'; // set menu yang sedang aktif

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



```

public function update(Request $request, string $id)
{
    $request->validate([
        'username' => 'required|string|min:3|unique:m_user,username,' . $id . ',user_id',
        'nama' => 'required|string|max:100',
        'password' => 'nullable|min:5',
        'level_id' => 'required|integer'
    ]);

    $user = UserModel::find($id);
    $user->username = $request->username;
    $user->nama = $request->nama;
    $user->password = $request->password ? bcrypt($request->password) : $user->password;
    $user->level_id = $request->level_id;
    $user->save();

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

```

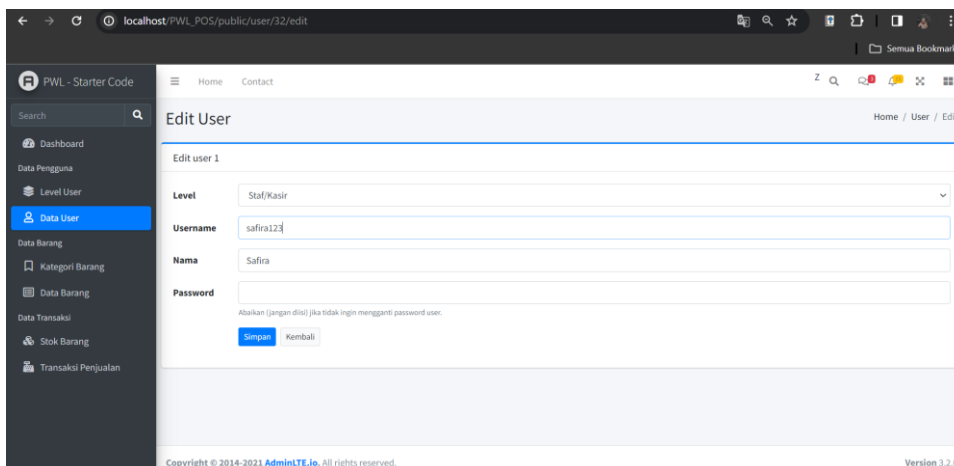
19. Selanjutnya, kita buat *view* untuk melakukan proses edit data user di PWL_POS/resources/views/user/edit.blade.php

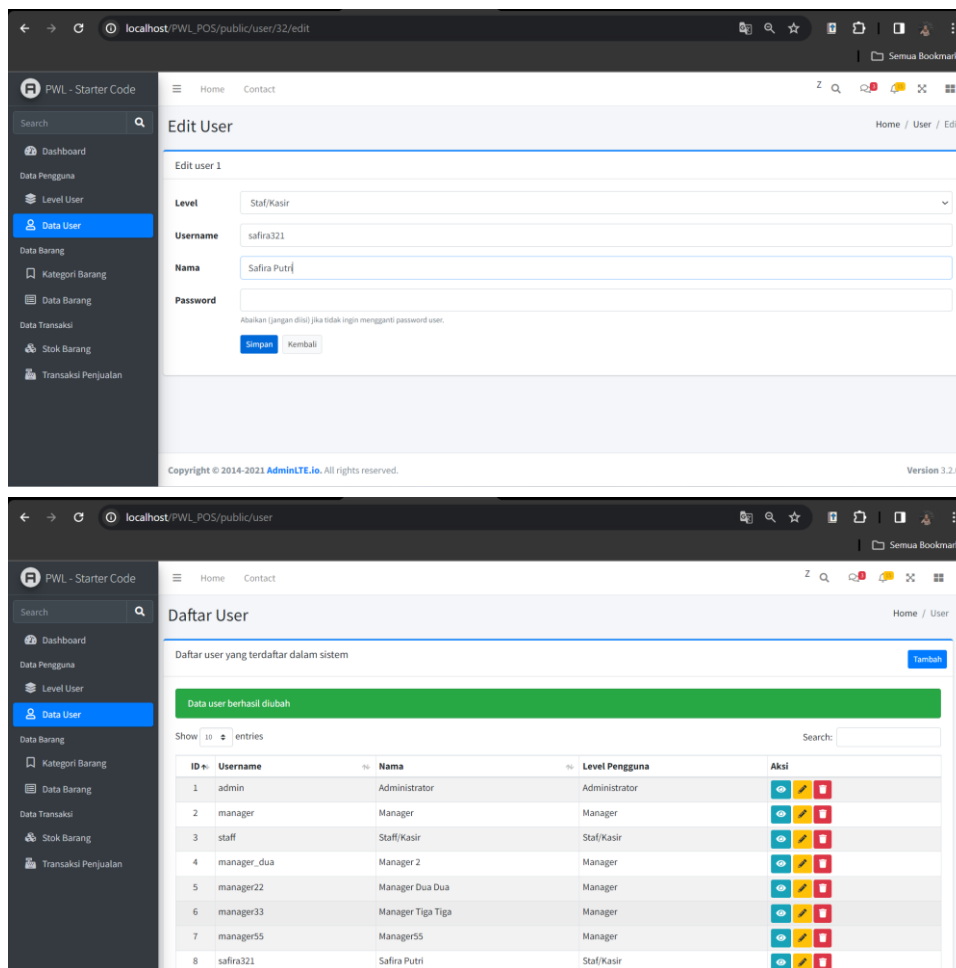
```

resources > views > user > @edit.blade.php > ...
1 @extends('layout.template')
2 @section('content')
3 <div class="card card-outline card-primary">
4 <div class="card-header">
5 <h3 class="card-title">{{ $page->title }}</h3>
6 <div class="card-tools"></div>
7 </div>
8 <div class="card-body">
9 @empty($user)
10 <div class="alert alert-danger alert-dismissible">
11 <div><i class="icon fas fa-ban"></i> Kesalahan!</div>
12 Data yang Anda cari tidak ditemukan.
13 </div>
14 <a href="{{ url('user') }}" class="btn btn-sm btn-default mt-2">Kembali</a>
15 @else
16 <form method="POST" action="{{ url('/user', $user->user_id) }}" class="form-horizontal">
17 @csrf
18 <!-- method field (PUT) --> <!-- tambahkan baris ini untuk proses edit yang butuh method PUT -->
19 <div class="form-group row">
20 <div class="col-1 control-label col-form-label">Level</div>
21 <div class="col-11">
22 <select class="form-control" id="level_id" name="level_id" required>
23 <option value=""> Pilih Level </option>
24 @foreach($level as $item)
25 <option value="{{ $item->level_id }}" @if($item->level_id == $user->level_id) selected @endif>{{ $item->level_nama }}</option>
26 @endforeach
27 </select>
28 @error('level_id')
29 <small class="form-text text-danger">{{ $message }}</small>
30 @enderror
31 </div>
32 </div>
33 <div class="form-group row">
34 <div class="col-1 control-label col-form-label">Username</div>
35 <div class="col-11">
36 <input type="text" class="form-control" id="username" name="username" value="{{ old('username', $user->username) }}" required>
37 @error('username')
38 <small class="form-text text-danger">{{ $message }}</small>
39 @enderror
40 </div>
41 </div>
42 <div class="form-group row">
43 <div class="col-1 control-label col-form-label">Nama</div>

```

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





21. Selanjutnya kita akan membuat penanganan untuk tombol hapus. Router `web.php` yang berfungsi untuk menangkap request hapus dengan method DELETE adalah

```

26 Route::group(['prefix' => 'user'], function () {
27     Route::get('/', [UserController::class, 'index']); // halaman awal user
28     Route::post('/list', [UserController::class, 'list']); // data user dalam bentuk json untuk datatables
29     Route::get('/create', [UserController::class, 'create']); // halaman form tambah user
30     Route::post('/', [UserController::class, 'store']); // simpan data user baru
31     Route::get('/{id}', [UserController::class, 'show']); // detail user
32     Route::get('/{id}/edit', [UserController::class, 'edit']); // halaman form edit user
33     Route::put('/{id}', [UserController::class, 'update']); // simpan perubahan data user
34     Route::delete('/{id}', [UserController::class, 'destroy']); // hapus data user
35 });
  
```

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

```

191 public function destroy(string $id)
192 {
193     $user = UserModel::find($id);
194     if (!$user) {
195         return redirect('/user-')->with('error', 'Data user tidak ditemukan');
196     }
197
198     try {
199         $user->delete(); // Hapus data user
200         return redirect('/user-')->with('success', 'Data user berhasil dihapus');
201     } catch (QueryException $e) { // Tangkap exception QueryException
202         return redirect('/user-')->with('error', 'Data user gagal dihapus karena masih terdapat tabel lain yang terkait dengan data ini');
203     }
204 }
  
```

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

```

11 <div class="card-body">
12     @if (session('success'))
13         <div class="alert alert-success">{{ session('success') }}</div>
14     @endif
15     @if (session('error'))
16         <div class="alert alert-danger">{{ session('error') }}</div>
17     @endif
18     <table class="table table-bordered table-striped table-hover table-sm" id="table_user">
19         <thead>
20             <tr>
21                 <th>ID</th>
22                 <th>Username</th>
23                 <th>Nama</th>
24                 <th>Level
25                 Pengguna</th>
26                 <th>Aksi</th>
27             </tr>
28         </thead>
29     </table>
30 </div>

```

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

The top screenshot shows a web browser at `localhost/PWL_POS/public/user` displaying the 'Daftar User' page. A confirmation dialog box is shown: 'localhost menyatakan Apakah Anda yakin menghapus data ini?' with 'Oke' and 'Batal' buttons. The table below shows 8 users.

ID	Username	Nama	Level Pengguna	Aksi
1	admin	Administrator	Administrator	[Icons]
2	manager	Manager	Manager	[Icons]
3	staff	Staff/Kasir	Staff/Kasir	[Icons]
4	manager_dua	Manager 2	Manager	[Icons]
5	manager22	Manager Dua Dua	Manager	[Icons]
6	manager33	Manager Tiga Tiga	Manager	[Icons]
7	manager55	Manager55	Manager	[Icons]
8	safira321	Safira Putri	Staff/Kasir	[Icons]

The bottom screenshot shows the same page after deletion. A green message bar says 'Data user berhasil dihapus'. The table now shows 7 users.

ID	Username	Nama	Level Pengguna	Aksi
1	admin	Administrator	Administrator	[Icons]
2	manager	Manager	Manager	[Icons]
3	staff	Staff/Kasir	Staff/Kasir	[Icons]
4	manager_dua	Manager 2	Manager	[Icons]
5	manager22	Manager Dua Dua	Manager	[Icons]
6	manager33	Manager Tiga Tiga	Manager	[Icons]
7	manager55	Manager55	Manager	[Icons]

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

Praktikum 4 – Implementasi *Filtering* Datatables:

Kita akan menerapkan filtering pada datatable yang sudah kita buat. Hal ini akan mempermudah kita dalam mengelompokkan suatu data sesuai kategori tertentu. Langkah-langkah yang kita kerjakan sebagai berikut

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

```
public function index()
{
    $breadcrumb = (object)[
        'title' => 'Daftar User',
        'list' => ['Home', 'User']
    ];
    $page = (object)[
        'title' => 'Daftar user yang terdaftar dalam sistem'
    ];
    $activeMenu = 'user'; //set menu yg sdg aktif
    $level = LevelModel::all();
    return view('user.index', ['breadcrumb' => $breadcrumb, 'page' => $page, 'activeMenu' => $activeMenu]);
}
```

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

```
<div class="row">
    <div class="col-md-12">
        <div class="form-group">
            <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

```
52 @push('js')
53     <script>
54         $(document).ready(function() {
55             var dataUser = $('#table_user').DataTable({
56                 serverSide: true,
57                 ajax: {
58                     "url": "{{ url('user/list') }}",
59                     "dataType": "json",
60                     "type": "POST"
61                     "data": function (d) {
62                         d.level_id = $('#level_id').val();
63                     }
64                 },
65             });
66         });
67     </script>
68 @endpush
```

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

```

92
93         $('#level_id').on('change', function() {
94             dataUser.ajax.reload();
95         });
96     });
97 </script>
98 @endpush

```

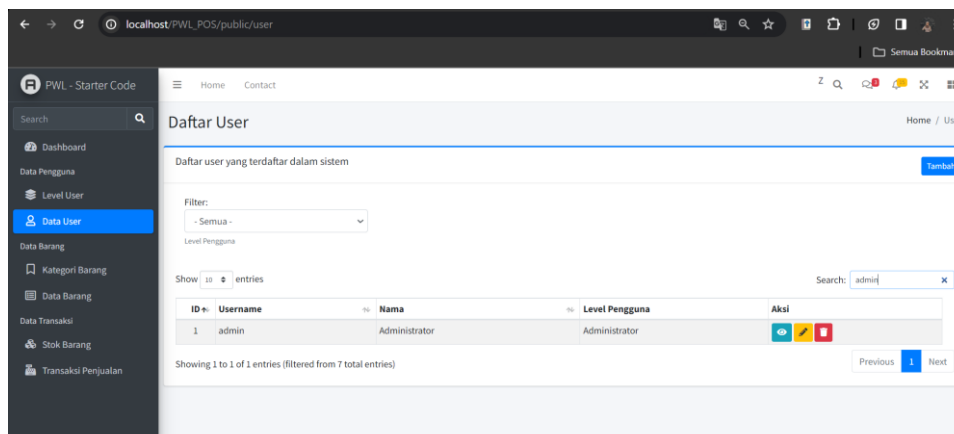
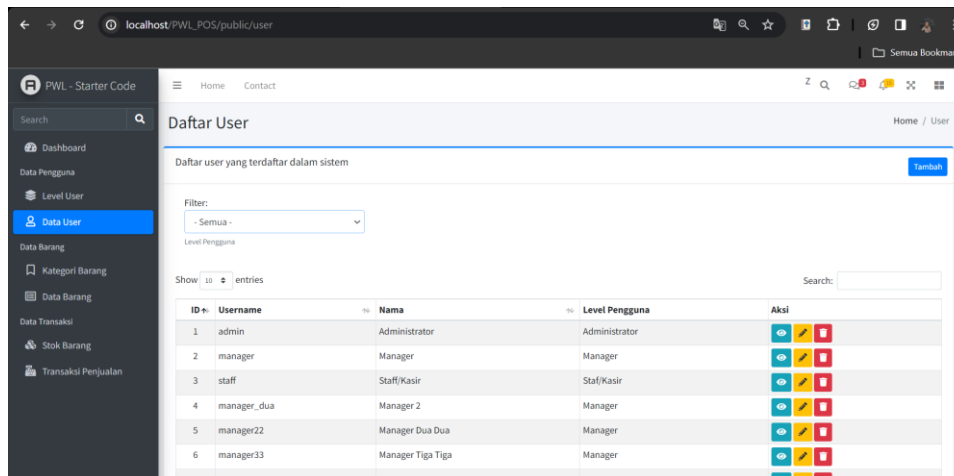
5. Tahapan akhir adalah memodifikasi fungsi **list()** pada **UserController.php** yang digunakan untuk menampilkan data pada datatable

```

78 public function list(Request $request)
79 {
80     $users = UserModel::select('user_id', 'username', 'nama', 'level_id')
81         ->with('level');
82
83     //Filter data berdasarkan level_id
84     if ($request->level_id) {
85         $users->where('level_id', $request->level_id);
86     }
87
88     return DataTables::of($users)
89         ->addIndexColumn()
90         ->addColumn('aksi', function ($user) {
91             $btn = '<a href="'.url('/user/'. $user->user_id).'" class="btn btn-info btn-sm"><i class="fas fa-eye"></i></a> ';
92             $btn .= '<a href="'.url('/user/'. $user->user_id . '/edit').'" class="btn btn-warning btn-sm"><i class="fas fa-pencil-alt"></i></a> ';
93             $btn .= '<form class="d-inline-block" method="POST" action="'.url('/user/'. $user->user_id).'">';
94                 . csrf_field() . method_field('DELETE')
95                 . '<button type="submit" class="btn btn-danger btn-sm" onclick="return confirm('\<\/a>');"><i class="fas fa-trash"></i></button>';
96             return $btn;
97         })
98         ->rawColumns(['aksi'])
99         ->make(true);
100 }
101 }

```

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.

A. PERTANYAAN

Jawablah pertanyaan berikut sesuai pemahaman materi di atas

1. Apa perbedaan *frontend template* dengan *backend template*?

Jawab:

Frontend template mengatur tampilan pengguna, seperti halaman web dan interaksi pengguna, menggunakan HTML, CSS, dan JavaScript. Backend template mengelola logika dan data di balik layar, seperti pengolahan data, pengaturan, dan hak akses, menggunakan bahasa pemrograman seperti PHP, Python, atau JavaScript. Jadi, frontend berfokus pada tampilan dan interaksi, sedangkan backend berfokus pada manajemen data dan proses sistem.

2. Apakah *layouting* itu penting dalam membangun sebuah website?

Jawab:

Layouting sangat penting dalam membangun sebuah website karena menentukan tata letak elemen-elemen seperti teks, gambar, dan tombol. Dengan layout yang baik, pengguna dapat dengan mudah menavigasi dan berinteraksi dengan konten website. Layout yang responsif juga penting agar halaman web dapat tampil dengan baik di berbagai perangkat. Keseluruhan, layouting mempengaruhi usability, pengalaman pengguna, dan kesuksesan website secara keseluruhan.

3. Jelaskan fungsi dari komponen laravel blade berikut : `@include()`, `@extend()`, `@section()`, `@push()`, `@yield()`, dan `@stack()`

Jawab:

Dalam Laravel Blade, `@include('view.name')` digunakan untuk menyertakan konten dari file blade lain ke dalam file blade saat ini saat rendering, memungkinkan penggunaan kembali kode. `@extends('layout.name')` memungkinkan penerapan tata letak induk pada file blade yang memerlukannya, membantu dalam membuat tampilan konsisten di seluruh situs. `@section('name')` dan `@endsection` menentukan bagian dari konten yang akan dimasukkan ke dalam layout induk. `@push('name')` dan `@endpush` memungkinkan penambahan konten seperti skrip atau gaya tambahan ke dalam stack untuk penggunaan di halaman tertentu. `@yield('name')` digunakan untuk menampilkan konten dari bagian yang ditentukan dalam layout induk, sementara `@stack('name')` digunakan untuk menampilkan konten yang telah ditambahkan ke dalam stack pada waktu tertentu. Keseluruhan, komponen-komponen ini mempermudah pengaturan dan penambahan konten di aplikasi Laravel Blade, memungkinkan pengembangan yang lebih terstruktur dan efisien.

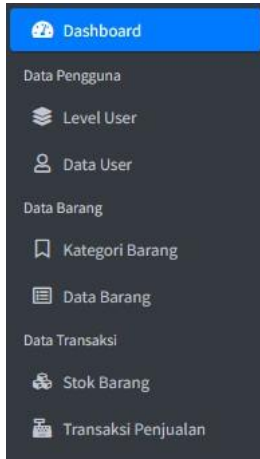
4. Apa fungsi dan tujuan dari variable `$activeMenu` ?

Jawab:

Variabel `$activeMenu` dalam konteks Laravel atau pemrograman web umumnya digunakan untuk menentukan atau menandai bagian mana dari antarmuka pengguna yang sedang aktif atau dipilih oleh pengguna. Fungsinya adalah untuk mengatur tampilan atau perilaku yang sesuai dengan menu atau bagian yang sedang digunakan atau diakses pengguna saat itu. Hal ini membantu dalam memberikan umpan balik visual kepada pengguna terkait posisi atau status mereka di dalam aplikasi atau situs web, serta memudahkan navigasi dan pemahaman pengguna terhadap struktur antarmuka yang ada.

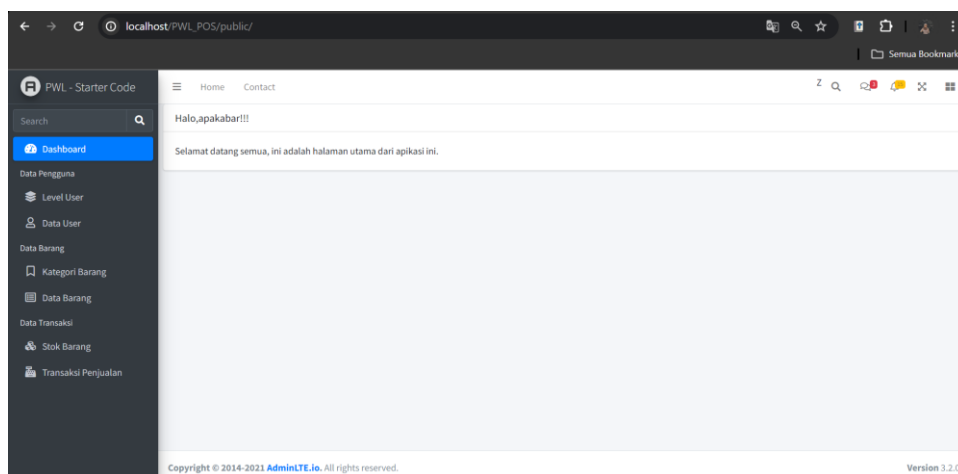
B. TUGAS

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.

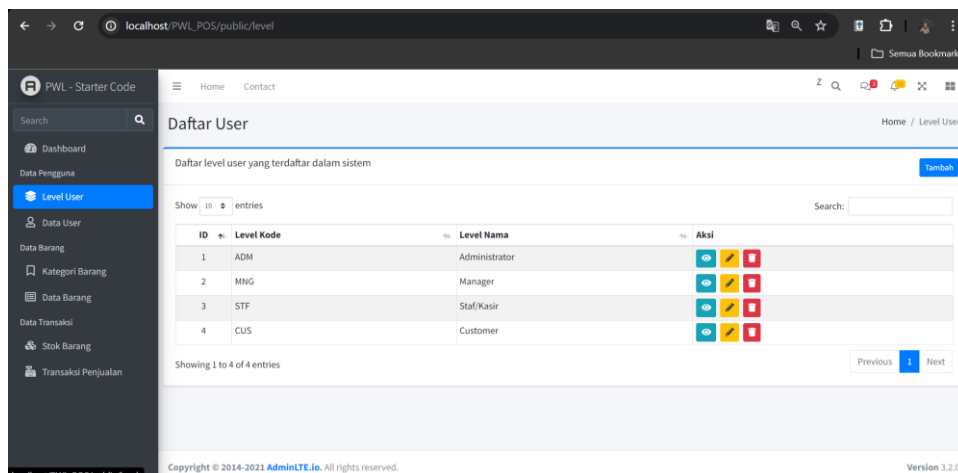


JAWAB

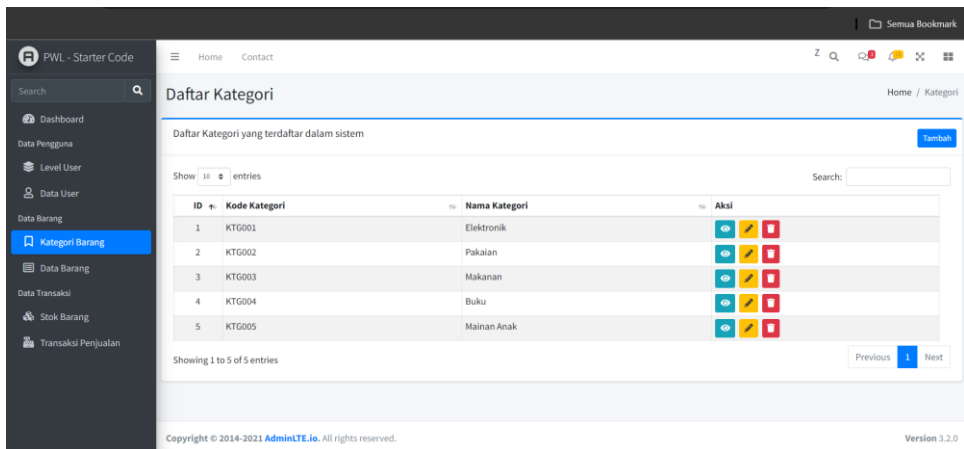
- **Halaman Dashboard**



- **Halaman Level User**



- **Halaman Kategori Barang**



Daftar Kategori

Daftar Kategori yang terdaftar dalam sistem

Show 10 entries

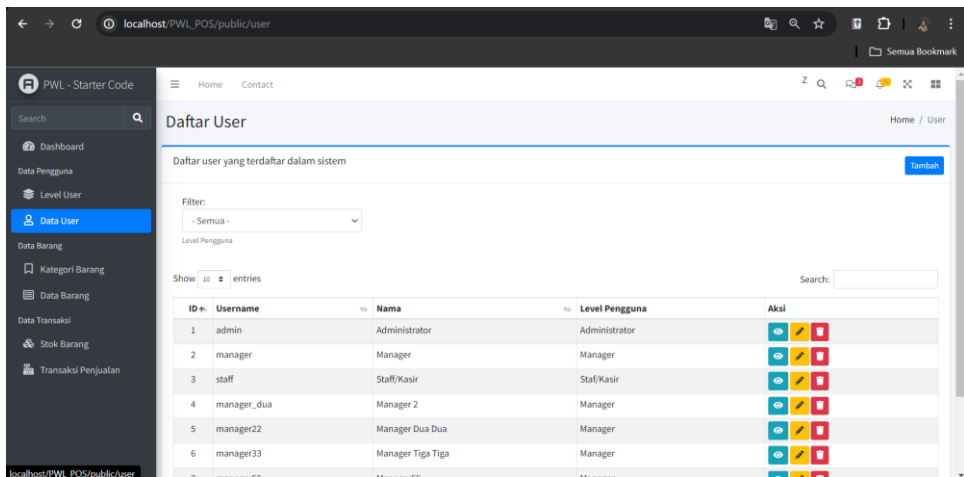
ID	Kode Kategori	Nama Kategori	Aksi
1	KTG001	Elektronik	[Edit] [Delete] [Add]
2	KTG002	Pakaian	[Edit] [Delete] [Add]
3	KTG003	Makanan	[Edit] [Delete] [Add]
4	KTG004	Buku	[Edit] [Delete] [Add]
5	KTG005	Mainan Anak	[Edit] [Delete] [Add]

Showing 1 to 5 of 5 entries

Previous 1 Next

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

- **Halaman Data User**



Daftar User

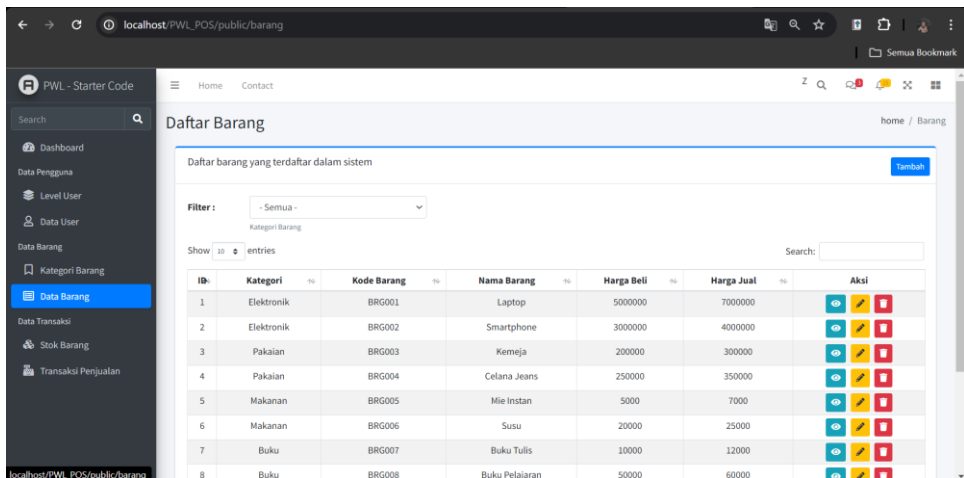
Daftar user yang terdaftar dalam sistem

Filter: - Semua -
Level Pengguna

Show 10 entries

ID	Username	Nama	Level Pengguna	Aksi
1	admin	Administrator	Administrator	[Edit] [Delete] [Add]
2	manager	Manager	Manager	[Edit] [Delete] [Add]
3	staff	Staff/Kasir	Staff/Kasir	[Edit] [Delete] [Add]
4	manager_dua	Manager 2	Manager	[Edit] [Delete] [Add]
5	manager22	Manager Dua Dua	Manager	[Edit] [Delete] [Add]
6	manager33	Manager Tiga Tiga	Manager	[Edit] [Delete] [Add]
7	manager44	Manager Empat Empat	Manager	[Edit] [Delete] [Add]

- **Halaman Data Barang**



Daftar Barang

Daftar barang yang terdaftar dalam sistem

Filter: - Semua -
Kategori Barang

Show 10 entries

ID	Kategori	Kode Barang	Nama Barang	Harga Beli	Harga Jual	Aksi
1	Elektronik	BRG001	Laptop	5000000	7000000	[Edit] [Delete] [Add]
2	Elektronik	BRG002	Smartphone	3000000	4000000	[Edit] [Delete] [Add]
3	Pakaian	BRG003	Kemeja	200000	300000	[Edit] [Delete] [Add]
4	Pakaian	BRG004	Celana Jeans	250000	350000	[Edit] [Delete] [Add]
5	Makanan	BRG005	Mie Instan	5000	7000	[Edit] [Delete] [Add]
6	Makanan	BRG006	Susu	20000	25000	[Edit] [Delete] [Add]
7	Buku	BRG007	Buku Tulis	10000	12000	[Edit] [Delete] [Add]
8	Buku	BRG008	Buku Pelajaran	50000	60000	[Edit] [Delete] [Add]

- **Halaman Stok Barang**

Daftar Stok

Daftar stok yang terdaftar dalam sistem

Filter : - Semua - - Semua -

Barang User

Show 10 entries Search:

ID	Barang	User	Stok Tanggal	Stok Jumlah	Aksi
1	Mie Instan	Administrator	2024-02-27	36	
2	Boneka	Administrator	2024-02-07	66	
3	Kemeja	Administrator	2024-02-18	60	
4	Smartphone	Administrator	2024-02-08	60	
5	Buku Tulis	Administrator	2024-02-15	82	
6	Laptop	Administrator	2024-04-16	10	
7	Televisi	Staff/Kasir	2024-04-14	80	
8	Spaghetti	Administrator	2024-04-14	50	

- **Halaman Transaksi Penjualan**

Daftar Transaksi Penjualan

Daftar transaksi penjualan yang terdaftar dalam sistem

Filter: - Semua -

Username Staff

Show 10 entries Search:

ID	Username Staff	Pembeli	Kode Penjualan	Tanggal	Aksi
1	admin	Pelanggan 1	PJ00001	2024-02-18	
2	admin	Pelanggan 2	PJ00002	2024-02-07	
3	admin	Pelanggan 3	PJ00003	2024-02-09	
4	admin	Pelanggan 4	PJ00004	2024-02-08	
5	admin	Pelanggan 5	PJ00005	2024-02-11	
6	admin	Pelanggan 6	PJ00006	2024-02-25	
7	admin	Pelanggan 7	PJ00007	2024-03-01	