



Mata Kuliah : Pemrograman Web Lanjut (PWL)
Program Studi : D4 – Teknik Informatika / D4 – Sistem Informasi Bisnis
Semester : 4 (empat) / 5 (lima)
Pertemuan ke- : 6 (enam)

JOBSHEET 06

Ajax Form (AdminLTE) dan Client Validation

Proses pembuatan form CRUD (Create, Read, Update, Delete) dengan validasi di Laravel 10 menggunakan jQuery Validation melibatkan beberapa langkah penting yang mencakup pengaturan database, pembuatan model dan migrasi, pengembangan controller, penulisan view, dan penambahan validasi form di sisi klien. *Client side form validation* lebih dilakukan disisi browser dan bukan untuk tujuan keamanan, tetapi lebih ke kenyamanan pengguna. Sedangkan *server side validation* dilakukan di sisi server dengan tujuan keamanan dengan *filter* semua *request* yang masuk sebelum akhirnya diproses lanjutan.

Salah satu cara yang populer untuk melakukan validasi di sisi klien adalah dengan menggunakan plugin jQuery Validation. Plugin **jQuery Validation** digunakan untuk menambahkan validasi sisi klien pada form. Misalnya, Kita bisa mengatur agar suatu input wajib diisi dan tidak boleh lebih dari 255 karakter. Validasi ini membantu dalam memberikan umpan balik langsung kepada pengguna tentang kesalahan input tanpa perlu memuat ulang halaman ataupun mengirim *request* ke server.

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

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

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

A. **AJAX form**

AJAX (Asynchronous JavaScript and XML) adalah sebuah teknik atau metode dalam pengembangan web yang memungkinkan aplikasi web untuk mengirim dan menerima data dari server secara asinkron (tanpa memuat ulang seluruh halaman). Dengan AJAX, interaksi antara



klien dan server menjadi lebih dinamis dan responsif, karena pengguna dapat berinteraksi dengan halaman web dan melihat perubahan langsung tanpa harus melakukan refresh halaman.

Ajax form adalah teknik di mana sebuah form HTML dikirim ke server secara asinkron menggunakan AJAX, tanpa memuat ulang seluruh halaman web. Dengan AJAX form, Kita bisa mengirim data ke server dan menampilkan respons secara dinamis di halaman, sehingga meningkatkan pengalaman pengguna dengan membuat interaksi lebih cepat dan lebih responsif.

Mengapa Menggunakan AJAX Form?

1. **Response Instan:** AJAX memungkinkan Kita untuk mengirim data dan menerima respons dari server tanpa perlu memuat ulang halaman.
2. **Pengalaman Pengguna yang Lebih Baik:** Karena tidak ada pemuat ulang halaman, aplikasi terasa lebih cepat dan lebih interaktif, mirip dengan aplikasi desktop.

Pengurangan Beban Server: Dengan mengirim hanya data yang diperlukan, AJAX dapat mengurangi penggunaan bandwidth dan beban di server.

B. Validasi Sisi Client

Validasi di sisi klien adalah proses pemeriksaan data yang dimasukkan oleh pengguna pada form web sebelum data tersebut dikirim ke server. Validasi ini dilakukan menggunakan kode yang berjalan di browser pengguna, seperti JavaScript, dan bertujuan untuk memastikan bahwa data yang dimasukkan sesuai dengan aturan tertentu, seperti format email yang benar, panjang karakter yang sesuai, atau tidak adanya kolom kosong yang wajib diisi.

Tujuan dan Manfaat Validasi di Sisi Klien

1. Umpulan Balik Instan

Pengguna mendapatkan umpan balik segera setelah mereka memasukkan data yang tidak valid, seperti kesalahan format email atau kolom yang tidak diisi. Ini meningkatkan pengalaman pengguna (*user experience*) karena mereka tidak perlu menunggu respon dari server untuk mengetahui apakah input mereka benar atau salah.

2. Mengurangi Beban Server

Dengan melakukan validasi di sisi klien, kesalahan dapat diidentifikasi dan diperbaiki sebelum data dikirim ke server, sehingga server tidak perlu memproses permintaan yang tidak valid. Ini dapat mengurangi beban kerja server dan meningkatkan kinerja aplikasi.

3. Meningkatkan Efisiensi



Validasi di sisi klien membantu mencegah pengiriman data yang tidak valid, sehingga mengurangi jumlah permintaan HTTP yang perlu diproses oleh server. Hal ini menghemat bandwidth dan waktu pemrosesan, membuat aplikasi lebih efisien.

4. Memastikan Integritas Data

Dengan validasi di sisi klien, banyak kesalahan input yang dapat dicegah sebelum data mencapai server. Misalnya, memastikan bahwa nomor telepon hanya berisi angka atau alamat email mengikuti format yang benar.

5. Menyederhanakan Proses Pengembangan

Dengan validasi di sisi klien, pengembang dapat menangani banyak potensi kesalahan input di awal, yang menyederhanakan logika pemrosesan di sisi server. Ini memungkinkan pengembang untuk fokus pada validasi yang lebih kompleks atau logika bisnis lainnya di server.

6. Meningkatkan Keamanan

Meskipun validasi di sisi klien tidak bisa mengantikan validasi di sisi server (karena dapat dengan mudah diabaikan atau dimanipulasi oleh pengguna yang berpengalaman), validasi ini tetap dapat membantu dalam mengurangi jumlah data yang tidak valid yang mencapai server. Ini berfungsi sebagai lapisan pertama pertahanan, mencegah beberapa jenis input yang tidak diinginkan.

7. Memberikan Panduan Pengguna

Validasi di sisi klien memungkinkan pengembang untuk memberikan panduan dan instruksi yang lebih baik kepada pengguna tentang cara memasukkan data dengan benar. Misalnya, pesan kesalahan bisa ditampilkan di bawah kolom yang salah, memberikan petunjuk spesifik kepada pengguna

Bagaimana Validasi di Sisi Klien Bekerja?

Validasi di sisi klien biasanya dilakukan menggunakan JavaScript atau framework JavaScript seperti jQuery. Berikut adalah contoh sederhana validasi form di sisi klien:

```
<!DOCTYPE html>
<html>
<head>
    <title>Client-Side Validation Example</title>
</head>
<body>
    <form name="myForm" onsubmit="return validateForm()" method="post">
        Name: <input type="text" name="name"><br><br>
        Email: <input type="text" name="email"><br><br>
        <input type="submit" value="Submit">
    </form>
    <script>
        function validateForm() {
            var email = document.forms["myForm"]["email"].value;
            var name = document.forms["myForm"]["name"].value;
        }
    </script>
</body>
</html>
```



```
if (name == "") {  
    alert("Name must be filled out");  
    return false;  
}  
  
var emailPattern = /^[a-zA-Z0-9._-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,6}$/;  
if (!emailPattern.test(email)) {  
    alert("Please enter a valid email address");  
    return false;  
}  
return true;  
}  
</script>  
</body>  
</html>
```

Keuntungan Validasi di Sisi Klien

1. Responsif

Pengguna mendapatkan respons cepat terhadap input mereka tanpa perlu menunggu interaksi dengan server.

2. Interaktif

Dapat memberikan instruksi tambahan dan lebih kontekstual kepada pengguna untuk memperbaiki kesalahan.

3. Penghematan Sumber Daya

Mengurangi jumlah permintaan ke server yang tidak perlu, sehingga menghemat bandwidth dan sumber daya server.

Keterbatasan Validasi di Sisi Klien

1. Tidak Mengganti Validasi di Sisi Server

Validasi di sisi klien dapat dilewati oleh pengguna berpengalaman atau perangkat otomatis. Oleh karena itu, validasi di sisi klien harus selalu dilengkapi dengan validasi di sisi server untuk memastikan keamanan dan integritas data.

2. Ketergantungan pada JavaScript

Jika pengguna menonaktifkan JavaScript di browser mereka, validasi di sisi klien tidak akan berfungsi.

Validasi di sisi klien merupakan komponen penting dalam pengembangan aplikasi web modern, karena meningkatkan pengalaman pengguna dan efisiensi aplikasi. Namun, ini harus selalu digunakan bersama dengan validasi di sisi server untuk menjaga keamanan dan memastikan data yang diterima oleh aplikasi adalah valid dan sesuai dengan aturan bisnis.



C. jQuery Validation

Salah satu cara yang populer untuk melakukan validasi di sisi klien adalah dengan menggunakan plugin jQuery Validation. **jQuery Validation** adalah plugin jQuery yang digunakan untuk memvalidasi form HTML di sisi klien secara efisien dan interaktif. Plugin ini memudahkan pengembang untuk menambahkan logika validasi pada form dengan cara yang mudah dan dapat disesuaikan, memberikan umpan balik langsung kepada pengguna mengenai kesalahan input mereka sebelum data dikirim ke server.

Fitur Utama jQuery Validation

1. Kemudahan Penggunaan

jQuery Validation dirancang untuk memudahkan integrasi dan penggunaan. Dengan beberapa baris kode, Kita dapat menambahkan validasi ke form HTML tanpa perlu menulis logika validasi dari awal.

2. Validasi Real-Time

Plugin ini memvalidasi input form secara real-time saat pengguna mengetik atau setelah mereka pindah dari satu field ke field lainnya. Ini memberikan umpan balik langsung kepada pengguna mengenai kesalahan input mereka.

3. Aturan Validasi yang Siap Pakai:

jQuery Validation menyediakan berbagai aturan validasi yang siap digunakan, seperti:

- *required*: Memastikan bahwa field tidak kosong.
- *email*: Memastikan bahwa input berformat alamat email yang valid.
- *url*: Memastikan bahwa input berformat URL yang valid.
- *minlength* dan *maxlength*: Membatasi jumlah karakter minimum atau maksimum dalam input.
- *number*: Memastikan bahwa input hanya berisi angka.

4. Pesan Kesalahan Kustom

Kita dapat menyesuaikan pesan kesalahan yang ditampilkan kepada pengguna. Misalnya, Kita dapat mengubah pesan default seperti "This field is required" menjadi sesuatu yang lebih spesifik atau sesuai dengan konteks aplikasi Kita.

5. Integrasi dengan jQuery UI

jQuery Validation dapat dengan mudah diintegrasikan dengan jQuery UI untuk menampilkan pesan kesalahan dalam format yang lebih menarik, seperti menggunakan tooltip atau dialog box.

6. Validasi Multi-Field



Plugin ini mendukung validasi yang melibatkan lebih dari satu field. Misalnya, Kita bisa memastikan bahwa dua field password dan konfirmasi password memiliki nilai yang sama.

7. Plugin dan Ekstensi

jQuery Validation memiliki ekosistem plugin dan ekstensi yang memungkinkan Kita menambahkan aturan validasi kustom atau mengubah perilaku default.

Cara Menggunakan jQuery Validation

Berikut adalah contoh sederhana bagaimana jQuery Validation digunakan untuk memvalidasi form:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>jQuery Validation Example</title>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/jqueryui/1.12.1/jquery-ui.css">
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery-validate/1.19.3/jquery.validate.min.js"></script>
</head>
<body>
    <form id="myForm">
        <label for="name">Name:</label><input type="text" name="name" id="name"><br>
        <label for="email">Email:</label><input type="text" name="email" id="email"><br>
        <input type="submit" value="Submit">
    </form>

    <script>
        $(document).ready(function() {
            $("#myForm").validate({
                rules: {
                    name: "required",
                    email: {
                        required: true,
                        email: true
                    }
                },
                messages: {
                    name: "Please enter your name",
                    email: "Please enter a valid email address"
                }
            });
        });
    </script>
</body>
</html>
```

Penjelasan:

- **rules**: mendefinisikan aturan validasi untuk setiap field. Dalam contoh di atas:
 - *Field name* harus diisi (required).
 - *Field email* harus diisi dan harus berformat email yang valid (email).



- **messages:** mendefinisikan pesan kesalahan yang akan ditampilkan jika aturan validasi tidak terpenuhi.

Keuntungan Menggunakan jQuery Validation

1. Pengalaman Pengguna yang Lebih Baik

Pengguna mendapatkan umpan balik langsung, yang membantu mereka memperbaiki kesalahan input dengan cepat.

2. Pengurangan Beban Server

Validasi di sisi klien mengurangi jumlah permintaan yang tidak valid yang dikirim ke server, menghemat sumber daya server.

3. Fleksibilitas dan Kustomisasi

Plugin ini sangat fleksibel dan dapat dikustomisasi sesuai kebutuhan aplikasi, dari aturan validasi hingga pesan kesalahan yang ditampilkan.

4. Kompatibilitas dengan Semua Browser Modern

jQuery Validation kompatibel dengan hampir semua browser modern, sehingga dapat digunakan di berbagai lingkungan pengguna.

jQuery Validation memiliki berbagai metode bawaan yang sangat berguna untuk memvalidasi form di sisi klien. Selain metode standar seperti required, email, dan number, Kita juga dapat menambahkan metode validasi kustom menggunakan addMethod. Ini memungkinkan Kita untuk membuat aturan validasi yang lebih spesifik sesuai kebutuhan aplikasi Kita.

D. Method jQuery Validation

jQuery Validation menyediakan beberapa metode bawaan (built-in methods) yang dapat digunakan untuk memvalidasi form dengan berbagai jenis aturan. Selain itu, jQuery Validation juga memungkinkan pengembang untuk menambahkan metode kustom dengan addMethod, seperti yang telah dijelaskan sebelumnya. Berikut adalah beberapa metode tambahan yang tersedia dalam jQuery Validation

No	Method	Deskripsi
1	<i>required</i>	<ul style="list-style-type: none">• Memastikan bahwa field tidak kosong.• Contoh: <code>required: true</code>
2	<i>email</i>	<ul style="list-style-type: none">• Memastikan bahwa input berformat alamat email yang valid.• Contoh: <code>email: true</code>
3	<i>url</i>	<ul style="list-style-type: none">• Memastikan bahwa input berformat URL yang valid.• Contoh: <code>url: true</code>



4	<i>date</i>	<ul style="list-style-type: none">Memastikan bahwa input berformat tanggal yang valid (berdasarkan pengaturan regional)Contoh: <code>date: true</code>
5	<i>dateISO</i>	<ul style="list-style-type: none">Memastikan bahwa input berformat tanggal yang valid dalam format ISO (YYYY-MM-DD)Contoh: <code>dateISO: true</code>
6	<i>number</i>	<ul style="list-style-type: none">Memastikan bahwa input hanya berisi angka (integer atau desimal).Contoh: <code>number: true</code>
7	<i>digits</i>	<ul style="list-style-type: none">Memastikan bahwa input hanya berisi angka (tanpa desimal).Contoh: <code>digits: true</code>
8	<i>creditcard</i>	<ul style="list-style-type: none">Memastikan bahwa input berformat nomor kartu kredit yang valid.Contoh: <code>creditcard: true</code>
9	<i>equalTo</i>	<ul style="list-style-type: none">Memastikan bahwa nilai elemen form sama dengan elemen lain (misalnya, untuk konfirmasi password).Contoh: <code>equalTo: "#password"</code>
10	<i>maxLength</i>	<ul style="list-style-type: none">Memastikan bahwa input tidak melebihi jumlah karakter tertentu.Contoh: <code>maxlength: 10</code>
11	<i>minLength</i>	<ul style="list-style-type: none">Memastikan bahwa input memiliki minimal jumlah karakter tertentu.Contoh: <code> minlength: 5</code>
12	<i>rangeLength</i>	<ul style="list-style-type: none">Memastikan bahwa panjang input berada dalam rentang karakter tertentu.Contoh: <code>rangelength: [5, 10]</code>
13	<i>range</i>	<ul style="list-style-type: none">Memastikan bahwa nilai input berada dalam rentang tertentu (misalnya, angka 1 sampai 100)Contoh: <code>range: [1, 100]</code>
14	<i>max</i>	<ul style="list-style-type: none">Memastikan bahwa nilai input tidak melebihi angka maksimum tertentu.Contoh: <code>max: 100</code>
15	<i>min</i>	<ul style="list-style-type: none">Memastikan bahwa nilai input tidak kurang dari angka minimum tertentu.Contoh: <code>min: 1</code>
16	<i>remote</i>	<ul style="list-style-type: none">Memvalidasi nilai dengan mengirimkan permintaan ke server untuk memeriksa apakah nilai tersebut valid atau tersedia (misalnya, memeriksa ketersediaan username)Contoh<pre>remote: { url: "/check-username", type: "post" }</pre>
17	<i>step</i>	<ul style="list-style-type: none">Memastikan bahwa nilai input adalah kelipatan dari angka tertentu (berguna untuk validasi angka desimal).Contoh: <code>step: 10</code>
18	<i>phoneUS</i>	<ul style="list-style-type: none">Memastikan bahwa input berformat nomor telepon yang valid di AS.Contoh: <code>phoneUS: true</code>



19	<i>extension</i>	<ul style="list-style-type: none">Memastikan bahwa file yang diupload memiliki ekstensi tertentu.Contoh: <code>extension: "jpg png gif"</code>
20	<i>accept</i>	<ul style="list-style-type: none">Memastikan bahwa file yang diupload memiliki jenis MIME tertentu.Contoh: <code>accept: "image/*"</code>
21	<i>exactlength</i>	<ul style="list-style-type: none">Memastikan bahwa input hanya berisi karakter yang panjangnya sama persis dengan ketentuan.Contoh: <code>exactlength: 10</code>

jQuery Validation adalah alat yang sangat berguna untuk memastikan data yang dimasukkan ke dalam form web valid dan sesuai dengan aturan yang ditetapkan sebelum data tersebut dikirim ke server. Ini meningkatkan pengalaman pengguna, mengurangi kesalahan, dan mempermudah pengelolaan validasi form di sisi klien dalam pengembangan aplikasi web.

E. Praktikum Jobsheet

Langsung saja kita praktikkan untuk menggunakan Ajax form dan validasi disisi client.

Praktikum 1. Modal Ajax Tambah Data (Data User)

1. Kita buat form tambah data baru dengan menerapkan modal dan proses ajax.
2. Pertama yang kita siapkan adalah menambahkan **library jQuery Validation** dan **Sweetalert** ke aplikasi web kita. Caranya kita tambahkan link kedua **library** tersebut ke `template.blade.php`, library sudah disediakan oleh adminLTE.

```
template.blade.php X
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      <meta name="csrf-token" content="{{ csrf_token() }}>
9
10     <!-- Font Awesome -->
11     <link rel="stylesheet" href="{{ asset('plugins/fontawesome-free/css/all.min.css') }}>
12     <!-- Datatables -->
13     <link rel="stylesheet" href="{{ asset('plugins/datatables-bs4/css/dataTables.bootstrap4.min.css') }}>
14     <link rel="stylesheet" href="{{ asset('plugins/datatables-responsive/css/responsive.bootstrap4.min.css') }}>
15     <link rel="stylesheet" href="{{ asset('plugins/datatables-buttons/css/buttons.bootstrap4.min.css') }}>
16     <!-- SweetAlert2 -->
17     <link rel="stylesheet" href="{{ asset('plugins/sweetalert2-theme-bootstrap-4/bootstrap-4.min.css') }}>
18     <!-- Theme style -->
19     <link rel="stylesheet" href="{{ asset('dist/css/adminlte.min.css') }}>
20
21     @stack('css')
22 </head>
```



```
template.blade.php ×  
65 <script src="{{ asset('plugins/pdfmake/pdfmake.min.js') }}"></script>  
66 <script src="{{ asset('plugins/pdfmake/vfs_fonts.js') }}"></script>  
67 <script src="{{ asset('plugins/datatables-buttons/js/buttons.html5.min.js') }}"></script>  
68 <script src="{{ asset('plugins/datatables-buttons/js/buttons.print.min.js') }}"></script>  
69 <script src="{{ asset('plugins/datatables-buttons/js/buttons.colVis.min.js') }}"></script>  
70  
71 <!-- jquery-validation -->  
72 <script src="{{ asset('plugins/jquery-validation/jquery.validate.min.js') }}"></script>  
73 <script src="{{ asset('plugins/jquery-validation/additional-methods.min.js') }}"></script>  
74  
75 <!-- SweetAlert2 -->  
76 <script src="{{ asset('plugins/sweetalert2/sweetalert2.min.js') }}"></script>  
77  
78 <!-- AdminLTE App -->  
79 <script src="{{ asset('dist/js/adminlte.min.js') }}"></script>  
80 <script>  
81     $.ajaxSetup({  
82         headers: {  
83             'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('content')  
84         }  
85     });  
86 </script>  
87  
88 @stack('js')  
89 </body>  
90 </html>
```

3. Selanjutnya Kita modifikasi view `user/index.blade.php`, kita tambahkan tombol untuk membuat form popup ajax

```
@extends('layouts.template')  
  
@section('content')  
    <div class="card card-outline card-primary">  
        <div class="card-header">  
            <h3 class="card-title">{{ $page->title }}</h3>  
            <div class="card-tools">  
                <a class="btn btn-sm btn-primary mt-1" href="{{ url('user/create') }}">Tambah</a>  
                <button onclick="modalAction('{{ url('user/create_ajax') }}')" class="btn btn-sm btn-success mt-1">Tambah Ajax</button>  
            </div>  
        </div>  
        <div class="card-body">  
            @if (@session('success'))  
                <div class="alert alert-success">{{ session('success') }}</div>  
            @endif  
            @if (session('error'))  
                <div class="alert alert-danger">{{ session('error') }}</div>  
            @endif  
            <div class="row">  
                <div class="col-md-12">  
                    <div class="form-group row">  
                        <label class="col-1 control-label col-form-label">Filter:</label>  
                        <div class="col-3">  
                            <select class="form-control" id="level_id" name="level_id" required>  
                                <option value="">- Semua -</option>  
                                @foreach ($level as $item)  
                                    <option value="{{ $item->level_id }}>{{ $item->level_nama }}</option>  
                                @endforeach  
                            </select>  
                        </div>  
                    </div>  
                </div>  
            </div>  
        </div>  
    </div>
```

Kita tambahkan kode berikut, untuk membuat form modal tambah data user dengan ajax



```
<button onclick="modalAction('{{ url('/user/create_ajax') }}')" class="btn btn-sm btn-success mt-1">Tambah Ajax</button>
```

4. Selanjutnya kita tambahkan kode berikut pada **akhir @section('content')** pada view `user/index.blade.php`

```
<div id="myModal" class="modal fade animate shake" tabindex="-1" role="dialog" data-backdrop="static" data-keyboard="false" data-width="75%" aria-hidden="true"></div>
```

5. Kemudian kita tambahkan kode berikut pada **awal @push('js')** pada view `user/index.blade.php`

```
function modalAction(url = ''){
    $('#myModal').load(url, function(){
        $('#myModal').modal('show');
    });
}
```

Sehingga tampilan kode program akan seperti berikut

```
<div id="myModal" class="modal fade animate shake" tabindex="-1" role="dialog" data-backdrop="static"
    data-keyboard="false" data-width="75%" aria-hidden="true"></div>
@endsection

@push('css')
@endpush

@push('js')
<script>
    function modalAction(url = ''){
        $('#myModal').load(url, function(){
            $('#myModal').modal('show');
        });
    }

    var dataUser;
    $(document).ready(function() {
        dataUser = $('#table_user').DataTable({
            // serverSide: true, jika ingin menggunakan server side processing
            serverSide: true,
            ajax: {
                "url": "{{ url('user/list') }}",
                "dataType": "json",
                "type": "POST",
                "data": function (d){
                    d.level_id = $('#level_id').val();
                }
            },
            columns: [

```

Ubah seperti ini

6. Selanjutkan Kita modifikasi `route/web.php` untuk mengakomodir operasi ajax



```
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('/create_ajax', [UserController::class, 'create_ajax']); // Menampilkan halaman form tambah user Ajax
    Route::post('/ajax', [UserController::class, 'store_ajax']); // Menyimpan data user baru Ajax
    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
});
```

7. Kemudian Kita tambahkan fungsi `create_ajax()` pada file `UserController.php`

```
public function create_ajax()
{
    $level = LevelModel::select('level_id', 'level_nama')->get();

    return view('user.create_ajax')
        ->with('level', $level);
}
```

8. Setelah itu, kita buat **view baru** `user/create_ajax.blade.php` untuk menampilkan form dengan ajax

```
<form action="{{ url('/user/ajax') }}" method="POST" id="form-tambah">
@csrf
<div id="modal-master" class="modal-dialog modal-lg" role="document">
    <div class="modal-content">
        <div class="modal-header">
            <h5 class="modal-title" id="exampleModalLabel">Tambah Data User</h5>
            <button type="button" class="close" data-dismiss="modal" aria-label="Close"><span aria-hidden="true">&times;</span></button>
        </div>
        <div class="modal-body">
            <div class="form-group">
                <label>Level Pengguna</label>
                <select name="level_id" id="level_id" class="form-control" required>
                    <option value="">- Pilih Level -</option>
                    @foreach($level as $l)
                        <option value="{{ $l->level_id }}>{{ $l->level_nama }}</option>
                    @endforeach
                </select>
                <small id="error-level_id" class="error-text form-text text-danger"></small>
            </div>
            <div class="form-group">
                <label>Username</label>
                <input value="" type="text" name="username" id="username" class="form-control" required>
                <small id="error-username" class="error-text form-text text-danger"></small>
            </div>
            <div class="form-group">
                <label>>Nama</label>
                <input value="" type="text" name="nama" id="nama" class="form-control" required>
                <small id="error-nama" class="error-text form-text text-danger"></small>
            </div>
            <div class="form-group">
                <label>Password</label>
                <input value="" type="password" name="password" id="password" class="form-control" required>
                <small id="error-password" class="error-text form-text text-danger"></small>
            </div>
        </div>
    </div>
</div>
```



```
<div class="modal-footer">
    <button type="button" data-dismiss="modal" class="btn btn-warning">Batal</button>
    <button type="submit" class="btn btn-primary">Simpan</button>
</div>
</div>
</form>
<script>
$(document).ready(function() {
    $("#form-tambah").validate({
        rules: {
            level_id: {required: true, number: true},
            username: {required: true, minlength: 3, maxlength: 20},
            nama: {required: true, minlength: 3, maxlength: 100},
            password: {required: true, minlength: 6, maxlength: 20}
        },
        submitHandler: function(form) {
            $.ajax({
                url: form.action,
                type: form.method,
                data: $(form).serialize(),
                success: function(response) {
                    if(response.status){
                        $('#myModal').modal('hide');
                        Swal.fire({
                            icon: 'success',
                            title: 'Berhasil',
                            text: response.message
                        });
                        dataUser.ajax.reload();
                    }else{
                        $('.error-text').text('');
                        $.each(response.msgField, function(prefix, val) {
                            $('#error-' +prefix).text(val[0]);
                        });
                        Swal.fire({
                            icon: 'error',
                            title: 'Terjadi Kesalahan',
                            text: response.message
                        });
                    }
                }
            });
            return false;
        },
        errorElement: 'span',
        errorPlacement: function (error, element) {
            error.addClass('invalid-feedback');
            element.closest('.form-group').append(error);
        },
        highlight: function (element, errorClass, validClass) {
            $(element).addClass('is-invalid');
        },
        unhighlight: function (element, errorClass, validClass) {
            $(element).removeClass('is-invalid');
        }
    });
}</script>
```

9. Kemudian untuk mengakomodir proses simpan data melalui ajax, kita buat fungsi `store_ajax()` pada `UserController.php`



```
public function store_ajax(Request $request) {
    // cek apakah request berupa ajax
    if($request->ajax() || $request->wantsJson()){
        $rules = [
            'level_id' => 'required|integer',
            'username' => 'required|string|min:3|unique:m_user,username',
            'nama'      => 'required|string|max:100',
            'password'  => 'required|min:6'
        ];

        // use Illuminate\Support\Facades\Validator;
        $validator = Validator::make($request->all(), $rules);

        if($validator->fails()){
            return response()->json([
                'status' => false, // response status, false: error/gagal, true: berhasil
                'message' => 'Validasi Gagal',
                'msgField' => $validator->errors(), // pesan error validasi
            ]);
        }

        UserModel::create($request->all());
        return response()->json([
            'status' => true,
            'message' => 'Data user berhasil disimpan'
        ]);
    }
    redirect('/');
}
```

10. OK, sekarang kita coba melakukan proses tambah data user menggunakan form ajax. Amati apa yang terjadi dan laporankan pada laporan *jobsheet* dan *commit* di github kalian!!!

ID	Username	Nama	Level Pengguna	Aksi
1	admin	Administrator	Administrator	[Detail] [Edit] [Hapus]
2	manager	Manager	Manager	[Detail] [Edit] [Hapus]
3	staff	Staff/Kasir	Staff/Kasir	[Detail] [Edit] [Hapus]



The screenshots illustrate the process of managing users in a system:

- Screenshot 1: Tambah Data User**
A modal window titled "Tambah Data User" is open. It contains fields for "Level Pengguna" (selected as "Staff/Kasir"), "Username" (set to "Kasir1"), "Nama" (set to "Kasir1"), and "Password" (set to "*****"). There are "Batal" and "Simpan" buttons at the bottom.
- Screenshot 2: Confirmation Message**
A modal window titled "Berhasil" displays the message "Data user berhasil disimpan". It has an "OK" button.
- Screenshot 3: Daftar User**
The main page shows a table of users with columns: ID, Username, Nama, Level Pengguna, and Aksi (Actions). The data is:

ID	Username	Nama	Level Pengguna	Aksi
1	admin	Administrator	Administrator	[Detail, Edit, Hapus]
2	manager	Manager	Manager	[Detail, Edit, Hapus]
3	staff	Staff/Kasir	Staff/Kasir	[Detail, Edit, Hapus]
4	Kasir1	Kasir1	Staff/Kasir	[Detail, Edit, Hapus]

Praktikum 2. Modal Ajax Edit Data (Data User)

1. Pada Praktikum 2 ini, kita akan melakukan koding untuk proses edit menggunakan ajax.
2. Pertama-tama, kita **ubah** dulu fungsi `list()` pada `UserController.php` untuk mengganti **tombol edit** untuk bisa menggunakan modal



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

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

    return DataTables::of($users)
        ->addIndexColumn() // menambahkan kolom index / no urut (default nama kolom: DT_RowIndex)
        ->addColumn('aksi', function ($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>';*/
            $btn = '<button onclick="modalAction(\"'.url('/user/' . $user->user_id . '/show_ajax').'\") class="btn btn-info btn-sm">Detail</button> ';
            $btn .= '<button onclick="modalAction(\"'.url('/user/' . $user->user_id . '/edit_ajax').'\") class="btn btn-warning btn-sm">Edit</button> ';
            $btn .= '<button onclick="modalAction(\"'.url('/user/' . $user->user_id . '/delete_ajax').'\") class="btn btn-danger btn-sm">Hapus</button> ';

            return $btn;
        })
        ->rawColumns(['aksi']) // memberitahu bahwa kolom aksi adalah html
        ->make(true);
}
```

3. Selanjutnya kita modifikasi `routes/web.php` untuk mengakomodir request edit menggunakan ajax

```
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('/create_ajax', [UserController::class, 'create_ajax']); // Menampilkan halaman form tambah user Ajax
    Route::post('/ajax', [UserController::class, 'store_ajax']); // Menyimpan data user baru Ajax
    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::get('/{id}/edit_ajax', [UserController::class, 'edit_ajax']); // Menampilkan halaman form edit user Ajax
    Route::put('/{id}/update_ajax', [UserController::class, 'update_ajax']); // Menyimpan perubahan data user Ajax
    Route::delete('/{id}', [UserController::class, 'destroy']); // Menghapus data user
});
```

4. Kemudian, kita buat fungsi `edit_ajax()` pada `UserController.php`

```
// Menampilkan halaman form edit user ajax
public function edit_ajax(string $id)
{
    $user = UserModel::find($id);
    $level = LevelModel::select('level_id', 'level_nama')->get();

    return view('user.edit_ajax',[ 'user' => $user, 'level' => $level]);
}
```



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

-
5. Kita buat **view baru** pada `user/edit_ajax.blade.php` untuk menampilkan form view ajax

```
@empty($user)
    <div id="modal-master" class="modal-dialog modal-lg" role="document">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-title" id="exampleModalLabel">Kesalahan</h5>
                <button type="button" class="close" data-dismiss="modal" aria-
label="Close"><span aria-hidden="true">&times;</span></button>
```



```
</div>
<div class="modal-body">
    <div class="alert alert-danger">
        <h5><i class="icon fas fa-ban"></i> Kesalahan!!!</h5>
        Data yang anda cari tidak ditemukan</div>
        <a href="{{ url('/user') }}" class="btn btn-warning">Kembali</a>
    </div>
</div>
</div>
@else
    <form action="{{ url('/user/' . $user->user_id . '/update_ajax') }}" method="POST" id="form-edit">
        @csrf
        @method('PUT')
        <div id="modal-master" class="modal-dialog modal-lg" role="document">
            <div class="modal-content">
                <div class="modal-header">
                    <h5 class="modal-title" id="exampleModalLabel">Edit Data User</h5>
                    <button type="button" class="close" data-dismiss="modal" aria-
label="Close"><span aria-hidden="true">&times;</span></button>
                </div>
                <div class="modal-body">
                    <div class="form-group">
                        <label>Level Pengguna</label>
                        <select name="level_id" id="level_id" class="form-control" required>
                            <option value="">- Pilih Level -</option>
                            @foreach($level as $l)
                                <option {{ ($l->level_id == $user->level_id)? 'selected' : '' }}>{{ $l->level_nama }}</option>
                            @endforeach
                        </select>
                        <small id="error-level_id" class="error-text form-text text-
danger"></small>
                    </div>
                    <div class="form-group">
                        <label>Username</label>
                        <input value="{{ $user->username }}" type="text" name="username" id="username" class="form-control" required>
                        <small id="error-username" class="error-text form-text text-
danger"></small>
                    </div>
                    <div class="form-group">
                        <label>>Nama</label>
                        <input value="{{ $user->nama }}" type="text" name="nama" id="nama" class="form-control" required>
                        <small id="error-nama" class="error-text form-text text-danger"></small>
                    </div>
                    <div class="form-group">
                        <label>Password</label>
                        <input value="" type="password" name="password" id="password" class="form-
control">
                        <small class="form-text text-muted">Abaikan jika tidak ingin ubah
password</small>
                        <small id="error-password" class="error-text form-text text-
danger"></small>
                    </div>
                </div>
                <div class="modal-footer">
                    <button type="button" data-dismiss="modal" class="btn btn-
warning">Batal</button>
                    <button type="submit" class="btn btn-primary">Simpan</button>
                </div>
            </div>
        </form>
        <script>
```



```
$(document).ready(function() {
    $("#form-edit").validate({
        rules: {
            level_id: {required: true, number: true},
            username: {required: true, minlength: 3, maxlength: 20},
            nama: {required: true, minlength: 3, maxlength: 100},
            password: {minlength: 6, maxlength: 20}
        },
        submitHandler: function(form) {
            $.ajax({
                url: form.action,
                type: form.method,
                data: $(form).serialize(),
                success: function(response) {
                    if(response.status){
                        $('#myModal').modal('hide');
                        Swal.fire({
                            icon: 'success',
                            title: 'Berhasil',
                            text: response.message
                        });
                        dataUser.ajax.reload();
                    }else{
                        $('.error-text').text('');
                        $.each(response.msgField, function(prefix, val) {
                            $('#error-'+prefix).text(val[0]);
                        });
                        Swal.fire({
                            icon: 'error',
                            title: 'Terjadi Kesalahan',
                            text: response.message
                        });
                    }
                }
            });
            return false;
        },
        errorElement: 'span',
        errorPlacement: function (error, element) {
            error.addClass('invalid-feedback');
            element.closest('.form-group').append(error);
        },
        highlight: function (element, errorClass, validClass) {
            $(element).addClass('is-invalid');
        },
        unhighlight: function (element, errorClass, validClass) {
            $(element).removeClass('is-invalid');
        }
    });
});
</script>
@endempty
```

6. Selanjutnya, kita buat fungsi `update_ajax()` pada `UserController.php` untuk mengakomodir request update data user melalui ajax

```
public function update_ajax(Request $request, $id){
    // cek apakah request dari ajax
    if ($request->ajax() || $request->wantsJson()) {
        $rules = [
            'level_id' => 'required|integer',
            'username' => 'required|max:20|unique:m_user,username,'.$id.',user_id',
            'nama'      => 'required|max:100',
            'password'  => 'nullable|min:6|max:20'
        ];
    }
}
```



```
// use Illuminate\Support\Facades\Validator;
$validator = Validator::make($request->all(), $rules);

if ($validator->fails()) {
    return response()->json([
        'status' => false, // respon json, true: berhasil, false: gagal
        'message' => 'Validasi gagal.',
        'msgField' => $validator->errors() // menunjukkan field mana yang error
    ]);
}

$check = UserModel::find($id);
if ($check) {
    if (!$request->filled('password')) { // jika password tidak diisi, maka hapus dari
request
        $request->request->remove('password');
    }

    $check->update($request->all());
    return response()->json([
        'status' => true,
        'message' => 'Data berhasil diupdate'
    ]);
} else{
    return response()->json([
        'status' => false,
        'message' => 'Data tidak ditemukan'
    ]);
}
}
return redirect('/');
}
```

7. Sekarang kita coba bagian edit user, amati proses nya. Jangan lupa laporkan dan *commit* ke *repository git* kalian !



The screenshots show a user interface for managing users in a system. The left sidebar includes links for Dashboard, Data Pengguna, Level User, and Data User (which is selected). The main area shows a table of users with columns for ID, Username, Name, Level Pengguna, and Aksi (Actions).

Screenshot 1: Edit Data User

A modal window titled "Edit Data User" is open. It contains fields for "Level Pengguna" (Staff/Kasir), "Username" (Kasir 1), "Nama" (Kasir Satu), and "Password". Below the password field is a note: "Abalikan jika tidak ingin ubah password". At the bottom are "Batal" and "Simpan" buttons.

Screenshot 2: Confirmation Message

A modal window titled "Berhasil" displays the message "Data berhasil diupdate". There is an "OK" button at the bottom.

Screenshot 3: User List

The main table shows the following user data:

ID	Username	Nama	Level Pengguna	Aksi
1	admin	Administrator	Administrator	[Detail, Edit, Hapus]
2	manager	Manager	Manager	[Detail, Edit, Hapus]
3	staff	Staff/Kasir	Staff/Kasir	[Detail, Edit, Hapus]
4	Kasir 1	Kasir Satu	Staff/Kasir	[Detail, Edit, Hapus]

Praktikum 3. Modal Ajax Hapus Data (Data User)

1. Pada Praktikum 3 ini, kita akan melakukan koding untuk proses hapus menggunakan ajax.
2. Pertama-tama, kita ubah dulu `routes/web.php` untuk mengakomodir request halaman



konfirmasi untuk menghapus data

```
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('/create_ajax', [UserController::class, 'create_ajax']); // Menampilkan halaman form tambah user Ajax
    Route::post('/ajax', [UserController::class, 'store_ajax']); // Menyimpan data user baru Ajax
    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::get('{id}/edit_ajax', [UserController::class, 'edit_ajax']); // Menampilkan halaman form edit user Ajax
    Route::put('{id}/update_ajax', [UserController::class, 'update_ajax']); // Menyimpan perubahan data user Ajax
    Route::get('{id}/delete_ajax', [UserController::class, 'confirm_ajax']); // Untuk tampilan form confirm delete user Ajax
    Route::delete('{id}/delete_ajax', [UserController::class, 'delete_ajax']); // Untuk hapus data user Ajax
    Route::delete('{id}', [UserController::class, 'destroy']); // Menghapus data user
});
```

3. Kemudian kita buat fungsi `confirm_ajax()` pada `UserController.php`

```
public function confirm_ajax(string $id){
    $user = UserModel::find($id);

    return view('user.confirm_ajax', ['user' => $user]);
}
```

4. Selanjutnya kita view untuk konfirmasi hapus data dengan nama `user/confirm_ajax.blade.php`



```
@empty($user)
    <div id="modal-master" class="modal-dialog modal-lg" role="document">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-title" id="exampleModalLabel">Kesalahan</h5>
                <button type="button" class="close" data-dismiss="modal" aria-
label="Close"><span aria-hidden="true">&times;</span></button>
            </div>
            <div class="modal-body">
                <div class="alert alert-danger">
                    <h5><i class="icon fas fa-ban"></i> Kesalahan!!!</h5>
                    Data yang anda cari tidak ditemukan</div>
                    <a href="{{ url('/user') }}" class="btn btn-warning">Kembali</a>
                </div>
            </div>
        </div>
    @else
        <form action="{{ url('/user/' . $user->user_id.'/delete_ajax') }}" method="POST" id="form-
delete">
            @csrf
            @method('DELETE')
            <div id="modal-master" class="modal-dialog modal-lg" role="document">
                <div class="modal-content">
                    <div class="modal-header">
                        <h5 class="modal-title" id="exampleModalLabel">Hapus Data User</h5>
                        <button type="button" class="close" data-dismiss="modal" aria-
label="Close"><span aria-hidden="true">&times;</span></button>
                    </div>
                    <div class="modal-body">
                        <div class="alert alert-warning">
                            <h5><i class="icon fas fa-ban"></i> Konfirmasi !!!</h5>
                            Apakah Anda ingin menghapus data seperti di bawah ini?
                        </div>
                        <table class="table table-sm table-bordered table-striped">
                            <tr><th class="text-right col-3">Level Pengguna :</th><td class="col-9">{{
                            $user->level->level_nama }}</td></tr>
                            <tr><th class="text-right col-3">Username :</th><td class="col-9">{{
                            $user->username }}</td></tr>
                            <tr><th class="text-right col-3">Nama :</th><td class="col-9">{{ $user-
                            >nama }}</td></tr>
                        </table>
                    </div>
                    <div class="modal-footer">
                        <button type="button" data-dismiss="modal" class="btn btn-
warning">Batal</button>
                        <button type="submit" class="btn btn-primary">Ya, Hapus</button>
                    </div>
                </div>
            </form>
            <script>
                $(document).ready(function() {
                    $("#form-delete").validate({
                        rules: {},
                        submitHandler: function(form) {
                            $.ajax({
```



```
        url: form.action,
        type: form.method,
        data: $(form).serialize(),
        success: function(response) {
            if(response.status){
                $('#myModal').modal('hide');
                Swal.fire({
                    icon: 'success',
                    title: 'Berhasil',
                    text: response.message
                });
                dataUser.ajax.reload();
            }else{
                $('.error-text').text('');
                $.each(response.msgField, function(prefix, val) {
                    $('#error-' + prefix).text(val[0]);
                });
                Swal.fire({
                    icon: 'error',
                    title: 'Terjadi Kesalahan',
                    text: response.message
                });
            }
        });
        return false;
    },
    errorElement: 'span',
    errorPlacement: function (error, element) {
        error.addClass('invalid-feedback');
        element.closest('.form-group').append(error);
    },
    highlight: function (element, errorClass, validClass) {
        $(element).addClass('is-invalid');
    },
    unhighlight: function (element, errorClass, validClass) {
        $(element).removeClass('is-invalid');
    }
});
});  

</script>  

@endempty
```

5. Kemudian kita buat fungsi `delete_ajax()` pada `UserController.php` untuk mengakomodir *request* hapus data user



```
public function delete_ajax(Request $request, $id)
{
    // cek apakah request dari ajax
    if ($request->ajax() || $request->wantsJson()) {
        $user = UserModel::find($id);
        if ($user) {
            $user->delete();
            return response()->json([
                'status' => true,
                'message' => 'Data berhasil dihapus'
            ]);
        } else {
            return response()->json([
                'status' => false,
                'message' => 'Data tidak ditemukan'
            ]);
        }
    }
    return redirect('/');
}
```

6. Setelah semua selesai, mari kita coba untuk melakukan percobaan dari koding yang telah kita lakukan.

ID	Username	Nama	Level Pengguna	Aksi
1	admin	Administrator	Administrator	[Detail] [Edit] [Hapus]
2	manager	Manager	Manager	[Detail] [Edit] [Hapus]
3	staff	Staff/Kasir	Staff/Kasir	[Detail] [Edit] [Hapus]
4	Kasir 1	Kasir Satu	Staff/Kasir	[Detail] [Edit] [Hapus]



The screenshots illustrate the steps of deleting a user account:

- Screenshot 1: Confirmation Dialog**
A modal dialog titled "Hapus Data User" asks for confirmation: "Apakah Anda ingin menghapus data seperti di bawah ini?". It shows the details of the user being deleted: Level Pengguna: Staff/Kasir; Username: Kasir1; Name: Kasir Satu. At the bottom are "Batal" and "Ya, Hapus" buttons.
- Screenshot 2: Success Message**
The application displays a success message: "Berhasil" and "Data berhasil dihapus". A green checkmark icon is shown. An "OK" button is at the bottom right.
- Screenshot 3: User List After Deletion**
The user list table now shows only three entries: admin, manager, and staff. The "Aksi" column contains "Detail", "Edit", and "Hapus" buttons for each user.

7. Jangan lupa laporkan ke laporan jobsheet dan lakukan *commit* pada *repository git* kalian!!!

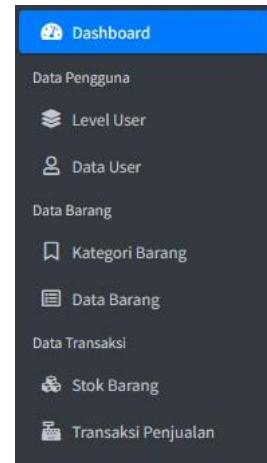


F. Tugas Jobsheet

Implementasikan coding untuk Ajax Form dan Client Validation dengan jQuery Validation pada menu berikut ini

- ✓ Tabel m_level
- ✓ Tabel m_kategori
- ✓ Tabel m_supplier
- ✓ Tabel m_barang

Laporkan pada laporan jobsheet dan Jangan lupa di commit dan push pada repository git kalian.



1. M_level

ID	Kode Level	Nama Level	Aksi
1	ADM	Administrator	[Detail] [Edit] [Hapus]
2	MNG	Manager	[Detail] [Edit] [Hapus]
3	STF	Staff/Kasir	[Detail] [Edit] [Hapus]

• Create

Tambah Data Level

Kode Level	AST
Nama Level	Assistance

[Batal] [Simpan]



The screenshot shows a modal dialog box in the center of the screen with a green checkmark icon and the text "Berhasil" (Successful). Below it, a message says "Data level berhasil disimpan" (Level data has been saved successfully). At the bottom right of the modal is a blue "OK" button. The background shows a table of level data with four entries: ADM, MNG, STF, and AST. The left sidebar shows navigation links for Dashboard, Data Pengguna, Level User, Data Barang, Kategori Barang, Data Supplier, Data Barang, Data Transaksi, Stok Barang, and Transaksi Penjualan.

The screenshot shows the "Daftar Level" (List of Levels) page. It displays a table with four rows of level data. The columns are labeled "ID", "Kode Level", "Nama Level", and "Aksi". The data is as follows:

ID	Kode Level	Nama Level	Aksi
1	ADM	Administrator	[Detail] [Edit] [Hapus]
2	MNG	Manager	[Detail] [Edit] [Hapus]
3	STF	Staff/Kasir	[Detail] [Edit] [Hapus]
4	AST	Assistance	[Detail] [Edit] [Hapus]

Below the table, a message says "Showing 1 to 4 of 4 entries". The left sidebar is identical to the first screenshot.

• Edit

The screenshot shows an "Edit Data Level" dialog box. Inside the dialog, there are two input fields: "Kode Level" with the value "ASM" and "Nama Level" with the value "Assistance Manager". At the bottom right of the dialog are two buttons: "Batal" (Cancel) and "Simpan" (Save). The background shows the same list of levels as the previous screenshot. The left sidebar is also the same.



The screenshot shows a modal dialog box in the center of the screen with a green checkmark icon and the text "Berhasil" (Successful). Below it, a message says "Data berhasil diupdate" (Data has been updated successfully). At the bottom right of the modal is an "OK" button. The background shows a table of "Level User" data with four entries: ADM, MNG, STF, and ASM. The left sidebar menu is visible, showing "Level User" as the active tab.

This screenshot shows the same "Daftar Level" page as the previous one, but without the modal dialog. It displays the same table of four level entries. The left sidebar menu is identical to the first screenshot.

- Delete

A confirmation dialog box titled "Hapus Data Level" is displayed. It contains a yellow header with the text "Konfirmasi !!!" and "Apakah Anda ingin menghapus data seperti di bawah ini?". Below this is a table with two rows, showing "Kode Level: ASM" and "Nama Level: Assistance Manager". At the bottom right of the dialog are "Batal" and "Ya, Hapus" buttons. The background shows the same "Daftar Level" table and sidebar as the previous screenshots.



The screenshot shows a confirmation dialog box in the center of a web-based application interface. The dialog box contains a green checkmark icon, the text "Berhasil" (Successful), and the message "Data berhasil dihapus" (Data deleted successfully). Below the dialog box is a blue "OK" button. The background shows a table of data with three entries: ADM, MNG, and STF. The left sidebar menu includes options like Dashboard, Data Pengguna, Level User, Data User, Data Barang, Data Supplier, Data Transaksi, Stok Barang, and Transaksi Penjualan.

This screenshot shows the same application interface as the previous one, but without the confirmation dialog. It displays a table of three level entries: ADM (Administrator), MNG (Manager), and STF (Staff/Kasir). The table has columns for ID, Kode Level, Nama Level, and Aksi (Actions). The Aksi column contains buttons for Detail, Edit, and Hapus (Delete). The background sidebar and footer are identical to the first screenshot.

2. M_kategori

This screenshot shows the application interface for managing categories. It displays a table of five category entries: MKN (Makanan), MNM (Minuman), PKN (Pakaian), ATK (Alat Tulis), and MIN (Mainan). The table has columns for ID, Kode Kategori, Nama Kategori, and Aksi. The Aksi column contains buttons for Detail, Edit, and Hapus. The background sidebar and footer are identical to the previous screenshots.

- Create



Tambah Data Kategori

Kode Kategori: ELT

Nama Kategori: Elektronik

Batal Simpan

ID	Kode Kategori	Nama Kategori	Aksi
1	MKN		[Detail] [Edit] [Hapus]
2	MNM		[Detail] [Edit] [Hapus]
3	PKN	Pakaian	[Detail] [Edit] [Hapus]
4	ATK	Alat Tulis	[Detail] [Edit] [Hapus]
5	MIN	Mainan	[Detail] [Edit] [Hapus]

Showing 1 to 5 of 5 entries

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

Version 3.2.0

Berhasil

Data kategori berhasil disimpan

OK

Daftar Kategori

Daftar kategori yang terdaftar dalam sistem

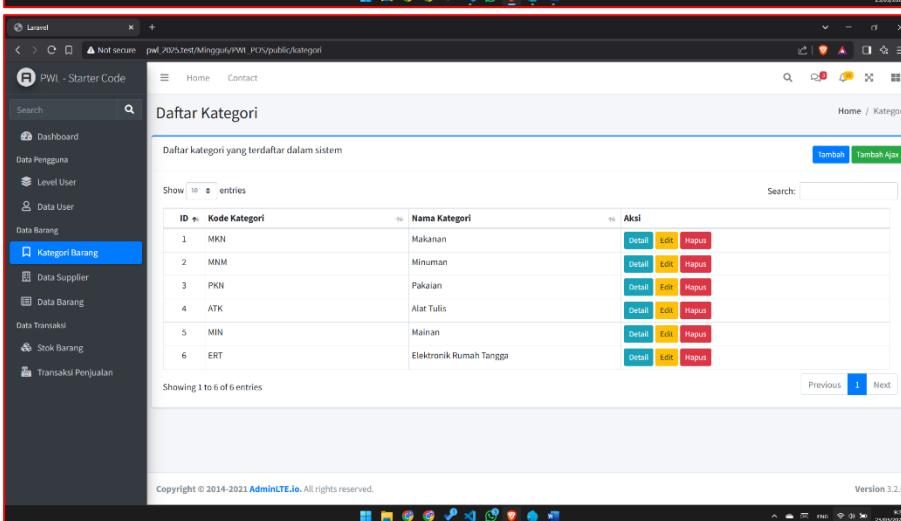
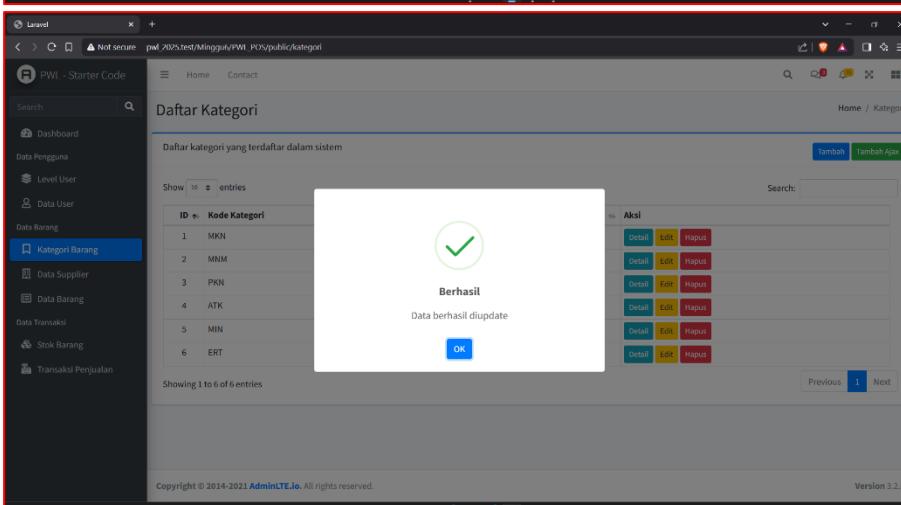
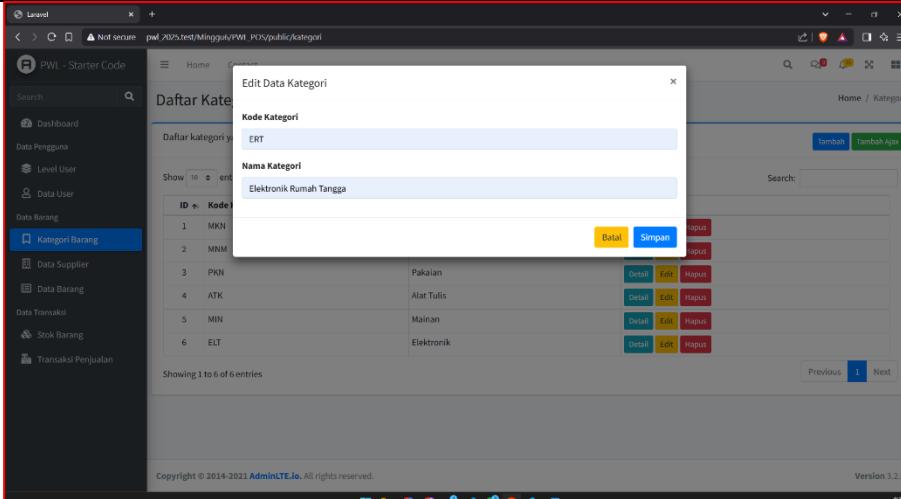
ID	Kode Kategori	Nama Kategori	Aksi
1	MKN	Makanan	[Detail] [Edit] [Hapus]
2	MNM	Minuman	[Detail] [Edit] [Hapus]
3	PKN	Pakaian	[Detail] [Edit] [Hapus]
4	ATK	Alat Tulis	[Detail] [Edit] [Hapus]
5	MIN	Mainan	[Detail] [Edit] [Hapus]
6	ELT	Elektronik	[Detail] [Edit] [Hapus]

Showing 1 to 6 of 6 entries

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

Version 3.2.0

- Edit



- Delete



The screenshots illustrate the process of deleting a category named "Elektronik Rumah Tangga" (ERT) from the system.

Screenshot 1: Confirmation Dialog
A modal dialog box titled "Hapus Data Kategori" asks for confirmation: "Apakah Anda ingin menghapus data seperti di bawah ini?". It displays the category details: "Kode Kategori: ERT" and "Nama Kategori: Elektronik Rumah Tangga". Below the dialog is a table listing categories, with "ERT" highlighted. At the bottom right of the dialog are "Batal" and "Ya, Hapus" buttons.

ID	Kode Kategori	Nama Kategori
1	MKN	
2	MNM	
3	PKN	Pakaian
4	ATK	Alat Tulis
5	MIN	Mainan
6	ERT	Elektronik Rumah Tangga

Screenshot 2: Success Message
A modal dialog box titled "Berhasil" confirms that "Data berhasil dihapus" (Data has been deleted successfully). It features a green checkmark icon and an "OK" button.

Screenshot 3: Updated Category List
The final screenshot shows the "Daftar Kategori" page after the deletion. The table now lists five categories: MKN, MNM, PKN, ATK, and MIN. The "Aksi" column for each row contains three buttons: "Detail", "Edit", and "Hapus".

3. M_supplier



ID	Kode Supplier	Nama Supplier	Alamat Supplier	Aksi
1	S001	PT. ABC	Jl. Raya No. 1	[Detail] [Edit] [Hapus]
2	S002	PT. DEF	Jl. Raya No. 2	[Detail] [Edit] [Hapus]
3	S003	PT. GHI	Jl. Raya No. 3	[Detail] [Edit] [Hapus]
4	S004	PT. JKL	Jl. Raya No. 4	[Detail] [Edit] [Hapus]
5	S005	PT. MNO	Jl. Raya No. 5	[Detail] [Edit] [Hapus]
6	S006	PT. PQR	Jl. Raya No. 6	[Detail] [Edit] [Hapus]
7	S007	PT. STU	Jl. Raya No. 7	[Detail] [Edit] [Hapus]
8	S008	PT. VWX	Jl. Raya No. 8	[Detail] [Edit] [Hapus]
9	S009	PT. YZA	Jl. Raya No. 9	[Detail] [Edit] [Hapus]
10	S010	PT. BCD	Jl. Raya No. 10	[Detail] [Edit] [Hapus]

• Create

Tambah Data Supplier

Kode Supplier
S011

Nama Supplier
PT. EFG

Alamat Supplier
Jl. Raya No. 11

[Batal] [Simpan]

Berhasil

Data supplier berhasil disimpan

[OK]



ID#	Kode Supplier	Nama Supplier	Alamat Supplier	Aksi
1	S001	PT.ABC	Jl. Raya No. 1	[Detail] [Edit] [Hapus]
2	S002	PT.DEF	Jl. Raya No. 2	[Detail] [Edit] [Hapus]
3	S003	PT.GHI	Jl. Raya No. 3	[Detail] [Edit] [Hapus]
4	S004	PT.JKL	Jl. Raya No. 4	[Detail] [Edit] [Hapus]
5	S005	PT.MNO	Jl. Raya No. 5	[Detail] [Edit] [Hapus]
6	S006	PT.PQR	Jl. Raya No. 6	[Detail] [Edit] [Hapus]
7	S007	PT.STU	Jl. Raya No. 7	[Detail] [Edit] [Hapus]
8	S008	PT.VWX	Jl. Raya No. 8	[Detail] [Edit] [Hapus]
9	S009	PT.YZA	Jl. Raya No. 9	[Detail] [Edit] [Hapus]
10	S010	PT.BCD	Jl. Raya No. 10	[Detail] [Edit] [Hapus]
11	S011	PT.EFG	Jl. Raya No. 11	[Detail] [Edit] [Hapus]

- Edit

Edit Data Supplier

Kode Supplier
S011

Nama Supplier
PT.EFG

Alamat Supplier
Jl.Raya No.12

Batal Simpan

Berhasil

Data berhasil diupdate

OK



ID	Kode Supplier	Nama Supplier	Alamat Supplier	Aksi
1	S001	PT.ABC	Jl. Raya No. 1	[Detail] [Edit] [Hapus]
2	S002	PT.DEF	Jl. Raya No. 2	[Detail] [Edit] [Hapus]
3	S003	PT.GHI	Jl. Raya No. 3	[Detail] [Edit] [Hapus]
4	S004	PT.JKL	Jl. Raya No. 4	[Detail] [Edit] [Hapus]
5	S005	PT.MNO	Jl. Raya No. 5	[Detail] [Edit] [Hapus]
6	S006	PT.PQR	Jl. Raya No. 6	[Detail] [Edit] [Hapus]
7	S007	PT.STU	Jl. Raya No. 7	[Detail] [Edit] [Hapus]
8	S008	PT.VWX	Jl. Raya No. 8	[Detail] [Edit] [Hapus]
9	S009	PT.YZA	Jl. Raya No. 9	[Detail] [Edit] [Hapus]
10	S010	PT.BCD	Jl. Raya No. 10	[Detail] [Edit] [Hapus]
11	S011	PT.EFG	Jl. Raya No. 12	[Detail] [Edit] [Hapus]

- Delete

Konfirmasi !!!
Apakah Anda ingin menghapus data seperti di bawah ini?

Kode Supplier : S011
Nama Supplier : PT.EFG
Alamat Supplier : Jl. Raya No. 12

Batal Ya, Hapus

Berhasil
Data berhasil dihapus

OK



ID	Kode Supplier	Nama Supplier	Alamat Supplier	Aksi
1	S001	PT.ABC	Jl. Raya No. 1	[Detail] [Edit] [Hapus]
2	S002	PT.DEF	Jl. Raya No. 2	[Detail] [Edit] [Hapus]
3	S003	PT.GHI	Jl. Raya No. 3	[Detail] [Edit] [Hapus]
4	S004	PT.JKL	Jl. Raya No. 4	[Detail] [Edit] [Hapus]
5	S005	PT.MNO	Jl. Raya No. 5	[Detail] [Edit] [Hapus]
6	S006	PT.PQR	Jl. Raya No. 6	[Detail] [Edit] [Hapus]
7	S007	PT.STU	Jl. Raya No. 7	[Detail] [Edit] [Hapus]
8	S008	PT.VWX	Jl. Raya No. 8	[Detail] [Edit] [Hapus]
9	S009	PT.YZA	Jl. Raya No. 9	[Detail] [Edit] [Hapus]
10	S010	PT.BCD	Jl. Raya No. 10	[Detail] [Edit] [Hapus]

4. M_barang

ID	Kode Barang	Nama Barang	Harga Beli	Harga Jual	Kategori Barang	Aksi
1	MKN001	Nasi Goreng	10000	15000	Makanan	[Detail] [Edit] [Hapus]
2	MKN002	Mie Goreng	5000	10000	Makanan	[Detail] [Edit] [Hapus]
3	MNM001	Teh Botol	2000	4000	Minuman	[Detail] [Edit] [Hapus]
4	MNM002	Aqua	1000	2000	Minuman	[Detail] [Edit] [Hapus]
5	PKN001	Kemeja	50000	70000	Pakaian	[Detail] [Edit] [Hapus]
6	PKN002	Celana	70000	100000	Pakaian	[Detail] [Edit] [Hapus]
7	ATK001	Pensil	1000	2000	Alat Tulis	[Detail] [Edit] [Hapus]
8	ATK002	Buku Tulis	5000	10000	Alat Tulis	[Detail] [Edit] [Hapus]
9	MIN001	Lego	20000	25000	Mainan	[Detail] [Edit] [Hapus]
10	MIN002	Boneka	30000	40000	Mainan	[Detail] [Edit] [Hapus]

- Create

Tambah Data Barang

Kategori Barang: Minuman

Kode Barang: MNM003

Nama Barang: Jus

Harga Beli: 6000

Harga Jual: 10000

Aksi:

ID	Kode Barang	Nama Barang	Harga Beli	Harga Jual	Kategori Barang	Aksi
1	MKN001	Nasi Goreng	10000	15000	Makanan	[Detail] [Edit] [Hapus]
2	MKN002	Mie Goreng	5000	10000	Makanan	[Detail] [Edit] [Hapus]
3	MNM001	Teh Botol	2000	4000	Minuman	[Detail] [Edit] [Hapus]
4	MNM002	Aqua	1000	2000	Minuman	[Detail] [Edit] [Hapus]
5	PKN001	Kemeja	50000	70000	Pakaian	[Detail] [Edit] [Hapus]
6	PKN002	Celana	70000	100000	Pakaian	[Detail] [Edit] [Hapus]
7	ATK001	Pensil	1000	2000	Alat Tulis	[Detail] [Edit] [Hapus]
8	ATK002	Buku Tulis	5000	10000	Alat Tulis	[Detail] [Edit] [Hapus]
9	MIN001	Lego	20000	25000	Mainan	[Detail] [Edit] [Hapus]
10	MIN002	Boneka	30000	40000	Mainan	[Detail] [Edit] [Hapus]



Laravel x + Not secure pwl.2025/test/Minggu4/PWI_PoS/public/barang

Daftar Barang

Daftar barang yang terdaftar dalam sistem

Filter: - Semua - Kategori Barang

Show 25 entries

ID Kode Barang % Nama Barang % Harga Beli % Harga Jual % Kategori Barang Aksi

1	MKN001	Nasi Goreng	10000	15000	Makanan	[Detail] [Edit] [Hapus]
2	MKN002	Mie Goreng	5000	10000	Makanan	[Detail] [Edit] [Hapus]
3	MNM001	Teh Botol	2000	4000	Minuman	[Detail] [Edit] [Hapus]
4	MNM002	Aqua	1000	2000	Minuman	[Detail] [Edit] [Hapus]
5	PKN001	Kemeja	50000	70000	Pakaian	[Detail] [Edit] [Hapus]
6	PKN002	Celana	70000	100000	Pakaian	[Detail] [Edit] [Hapus]
7	ATK001	Pensil	1000	2000	Alat Tulis	[Detail] [Edit] [Hapus]
8	ATK002	Buku Tulis	5000	10000	Alat Tulis	[Detail] [Edit] [Hapus]
9	MIN001	Lego	20000	25000	Mainan	[Detail] [Edit] [Hapus]
10	MIN002	Boneka	30000	40000	Mainan	[Detail] [Edit] [Hapus]
11	MNM003	Jus	6000	10000	Minuman	[Detail] [Edit] [Hapus]

Showing 1 to 10 of 11 entries

OK

Berhasil

Search:

Home / Barang

Tambah Tambah/Ajax

Laravel x + Not secure pwl.2025/test/Minggu4/PWI_PoS/public/barang

Daftar Barang

Daftar barang yang terdaftar dalam sistem

Filter: - Semua - Kategori Barang

Show 25 entries

ID Kode Barang % Nama Barang % Harga Beli % Harga Jual % Kategori Barang Aksi

1	MKN001	Nasi Goreng	10000	15000	Makanan	[Detail] [Edit] [Hapus]
2	MKN002	Mie Goreng	5000	10000	Makanan	[Detail] [Edit] [Hapus]
3	MNM001	Teh Botol	2000	4000	Minuman	[Detail] [Edit] [Hapus]
4	MNM002	Aqua	1000	2000	Minuman	[Detail] [Edit] [Hapus]
5	PKN001	Kemeja	50000	70000	Pakaian	[Detail] [Edit] [Hapus]
6	PKN002	Celana	70000	100000	Pakaian	[Detail] [Edit] [Hapus]
7	ATK001	Pensil	1000	2000	Alat Tulis	[Detail] [Edit] [Hapus]
8	ATK002	Buku Tulis	5000	10000	Alat Tulis	[Detail] [Edit] [Hapus]
9	MIN001	Lego	20000	25000	Mainan	[Detail] [Edit] [Hapus]
10	MIN002	Boneka	30000	40000	Mainan	[Detail] [Edit] [Hapus]
11	MNM003	Jus	6000	10000	Minuman	[Detail] [Edit] [Hapus]

Showing 1 to 11 of 11 entries

OK

Berhasil

Search:

Home / Barang

Tambah Tambah/Ajax

• Edit

Laravel x + Not secure pwl.2025/test/Minggu4/PWI_PoS/public/barang

Daftar Barang

Edit Data Barang

Kategori Barang

Minuman

ID Kode Barang % Nama Barang % Harga Beli % Harga Jual % Kategori Barang Aksi

1	MKN001	Jus Buah	6000	10000	Makanan	[Detail] [Edit] [Hapus]
2	MKN002					[Detail] [Edit] [Hapus]
3	MNM001					[Detail] [Edit] [Hapus]
4	MNM002					[Detail] [Edit] [Hapus]
5	PKN001					[Detail] [Edit] [Hapus]
6	PKN002					[Detail] [Edit] [Hapus]
7	ATK001					[Detail] [Edit] [Hapus]
8	ATK002					[Detail] [Edit] [Hapus]
9	MIN001					[Detail] [Edit] [Hapus]
10	MIN002					[Detail] [Edit] [Hapus]
11	MNM003					[Detail] [Edit] [Hapus]

Showing 1 to 11 of 11 entries

OK

Berhasil

Search:

Home / Barang

Tambah Tambah/Ajax



The screenshot shows a list of items with columns for ID, Kode Barang, Nama Barang, Harga Beli, Harga Jual, Kategori Barang, and Aksi (Actions). A modal window in the center displays a green checkmark icon and the text 'Berhasil' (Successful) with 'Data berhasil diupdate' (Data has been updated successfully). A blue 'OK' button is at the bottom of the modal.

ID	Kode Barang	Nama Barang	Harga Beli	Harga Jual	Kategori Barang	Aksi
1	MKN001	Nasi Goreng	10000	15000	Makanan	[Detail] [Edit] [Hapus]
2	MKN002	Mie Goreng	5000	10000	Makanan	[Detail] [Edit] [Hapus]
3	MNM001	Teh Botol	2000	4000	Minuman	[Detail] [Edit] [Hapus]
4	MNM002	Aqua	1000	2000	Minuman	[Detail] [Edit] [Hapus]
5	PKN001	Kemeja	50000	70000	Pakaian	[Detail] [Edit] [Hapus]
6	PKN002	Celana	70000	100000	Pakaian	[Detail] [Edit] [Hapus]
7	ATK001	Pensil	1000	2000	Alat Tulis	[Detail] [Edit] [Hapus]
8	ATK002	Buku Tulis	5000	10000	Alat Tulis	[Detail] [Edit] [Hapus]
9	MIN001	Lego	20000	25000	Mainan	[Detail] [Edit] [Hapus]
10	MIN002	Boneka	30000	40000	Mainan	[Detail] [Edit] [Hapus]
11	MNM003	Jus Buah	6000	10000	Minuman	[Detail] [Edit] [Hapus]

The screenshot shows the same list of items. A modal window in the center asks 'Apakah Anda ingin menghapus data seperti di bawah ini?' (Do you want to delete the data like the one below?). It lists the item details: Kategori Barang: Minuman, Kode Barang: MNM003, Nama Barang: Jus Buah, Harga Beli: 6000, and Harga Jual: 10000. At the bottom are 'Batal' (Cancel) and 'Ya, Hapus' (Yes, Delete) buttons.

ID	Kode Barang	Nama Barang	Harga Beli	Harga Jual	Kategori Barang	Aksi
1	MKN001	Nasi Goreng	10000	15000	Makanan	[Detail] [Edit] [Hapus]
2	MKN002	Mie Goreng	5000	10000	Makanan	[Detail] [Edit] [Hapus]
3	MNM001	Teh Botol	2000	4000	Minuman	[Detail] [Edit] [Hapus]
4	MNM002	Aqua	1000	2000	Minuman	[Detail] [Edit] [Hapus]
5	PKN001	Kemeja	50000	70000	Pakaian	[Detail] [Edit] [Hapus]
6	PKN002	Celana	70000	100000	Pakaian	[Detail] [Edit] [Hapus]
7	ATK001	Pensil	1000	2000	Alat Tulis	[Detail] [Edit] [Hapus]
8	ATK002	Buku Tulis	5000	10000	Alat Tulis	[Detail] [Edit] [Hapus]
9	MIN001	Lego	20000	25000	Mainan	[Detail] [Edit] [Hapus]
10	MIN002	Boneka	30000	40000	Mainan	[Detail] [Edit] [Hapus]
11	MNM003	Jus Buah	6000	10000	Minuman	[Detail] [Edit] [Hapus]

- **Delete**

The screenshot shows the same list of items. A modal window in the center asks 'Apakah Anda ingin menghapus data seperti di bawah ini?' (Do you want to delete the data like the one below?). It lists the item details: Kategori Barang: Minuman, Kode Barang: MNM003, Nama Barang: Jus Buah, Harga Beli: 6000, and Harga Jual: 10000. At the bottom are 'Batal' (Cancel) and 'Ya, Hapus' (Yes, Delete) buttons.

ID	Kode Barang	Nama Barang	Harga Beli	Harga Jual	Kategori Barang	Aksi
1	MKN001	Nasi Goreng	10000	15000	Makanan	[Detail] [Edit] [Hapus]
2	MKN002	Mie Goreng	5000	10000	Makanan	[Detail] [Edit] [Hapus]
3	MNM001	Teh Botol	2000	4000	Minuman	[Detail] [Edit] [Hapus]
4	MNM002	Aqua	1000	2000	Minuman	[Detail] [Edit] [Hapus]
5	PKN001	Kemeja	50000	70000	Pakaian	[Detail] [Edit] [Hapus]
6	PKN002	Celana	70000	100000	Pakaian	[Detail] [Edit] [Hapus]
7	ATK001	Pensil	1000	2000	Alat Tulis	[Detail] [Edit] [Hapus]
8	ATK002	Buku Tulis	5000	10000	Alat Tulis	[Detail] [Edit] [Hapus]
9	MIN001	Lego	20000	25000	Mainan	[Detail] [Edit] [Hapus]
10	MIN002	Boneka	30000	40000	Mainan	[Detail] [Edit] [Hapus]
11	MNM003	Jus Buah	6000	10000	Minuman	[Detail] [Edit] [Hapus]



Laravel x + Not secure pwl.2025/test/Minggu/PWI_PoS/public/barang

Daftar Barang

Daftar barang yang terdaftar dalam sistem

Filter: - Semua - Kategori Barang

Show 25 entries

ID Kode Barang Nama Barang Harga Beli Harga Jual Kategori Barang Aksi

1	MKN001	Nasi Goreng	10000	15000	Makanan	[Detail] [Edit] [Hapus]
2	MKN002	Mie Goreng	5000	10000	Makanan	[Detail] [Edit] [Hapus]
3	MNM001	Teh Botol	2000	4000	Minuman	[Detail] [Edit] [Hapus]
4	MNM002	Aqua	1000	2000	Minuman	[Detail] [Edit] [Hapus]
5	PKN001	Kemeja	50000	70000	Pakaian	[Detail] [Edit] [Hapus]
6	PKN002	Celana	70000	100000	Pakaian	[Detail] [Edit] [Hapus]
7	ATK001	Pensil	1000	2000	Alat Tulis	[Detail] [Edit] [Hapus]
8	ATK002	Buku Tulis	5000	10000	Alat Tulis	[Detail] [Edit] [Hapus]
9	MIN001	Lego	20000	25000	Mainan	[Detail] [Edit] [Hapus]
10	MIN002	Boneka	30000	40000	Mainan	[Detail] [Edit] [Hapus]

Berhasil Data berhasil dihapus

OK

Showing 1 to 10 of 10 entries

Previous 1 Next

Home / Barang

Tambah Tambah-Ajax

Laravel x + Not secure pwl.2025/test/Minggu/PWI_PoS/public/barang

Daftar Barang

Daftar barang yang terdaftar dalam sistem

Filter: - Semua - Kategori Barang

Show 25 entries

ID Kode Barang Nama Barang Harga Beli Harga Jual Kategori Barang Aksi

1	MKN001	Nasi Goreng	10000	15000	Makanan	[Detail] [Edit] [Hapus]
2	MKN002	Mie Goreng	5000	10000	Makanan	[Detail] [Edit] [Hapus]
3	MNM001	Teh Botol	2000	4000	Minuman	[Detail] [Edit] [Hapus]
4	MNM002	Aqua	1000	2000	Minuman	[Detail] [Edit] [Hapus]
5	PKN001	Kemeja	50000	70000	Pakaian	[Detail] [Edit] [Hapus]
6	PKN002	Celana	70000	100000	Pakaian	[Detail] [Edit] [Hapus]
7	ATK001	Pensil	1000	2000	Alat Tulis	[Detail] [Edit] [Hapus]
8	ATK002	Buku Tulis	5000	10000	Alat Tulis	[Detail] [Edit] [Hapus]
9	MIN001	Lego	20000	25000	Mainan	[Detail] [Edit] [Hapus]
10	MIN002	Boneka	30000	40000	Mainan	[Detail] [Edit] [Hapus]

Showing 1 to 10 of 10 entries

Previous 1 Next

Home / Barang

Tambah Tambah-Ajax

*** Sekian, dan selamat belajar ***