Aplikasi POS (Point Of Sales) menggunakan Framework Laravel 10



Prepared By : Safrizal

[SMK Negeri 1 Karang Baru](https://www.google.com/maps/place/SMK+Negeri+1+Karang+Baru/data=!4m2!3m1!19sChIJWePn2SF0NzARd2BOgu-Rsw0)

Daftar Isi

[**I. Penting untuk dibaca 3**](#_9bhie42lzkfn)

[A. Pengetahuan Awal 3](#_qokz19f0okg)

[B. Petunjuk Pengumpulan Tugas 3](#_sbtv1v6wwfwo)

[**II. Persiapan 4**](#_fone15bo4363)

[A. Instalasi Project Laravel 4](#_hilnalftwvxo)

[B. Membuat Database 7](#_m1ymy82s0ll3)

[1. Konfigurasi Database 7](#_kosi6om5sukh)

[2. Membuat Migration Database 8](#_lc0imr46g1fd)

[**III. Membuat Project 14**](#_sqlhrk5dbf41)

[A. Tampilan Utama 14](#_i4av96457gpo)

[B. Manajemen Kategori 22](#_swkmahbkgztt)

[1. Membuat CategoryController 22](#_hzaor7ra131n)

[2. Membuat Route Kategor 27](#_kaexopr7h3f3)

[3. Membuat Method store() 28](#_jhsclf6cr5hi)

[4. Membuat Method edit() dan update() 30](#_dup3sklxhll2)

[C. Manajemen Customer 32](#_s3434sm14c71)

[1. Membuat Controller Customer 32](#_6xemoi4dcabr)

[2. Membuat Route 36](#_gqm5wns3jagx)

[3. Membuat method create() 37](#_3hmh7u13qw52)

[4. Membuat method edit() dan update() 39](#_kau9yjgoyprg)

[5. Membuat method show() 42](#_544f7mujgtn3)

[D. Manajemen Product 46](#_pz9dq4vpmxmb)

[1. Membuat ProductController dan Modifikasi Model Product dan Model Category 46](#_9vgwllvc3uf5)

[2. Membuat Route Product dan Method index() 47](#_44uusfk88ux)

[3. Membuat Method create() dan store() 53](#_y2ocicrjy9of)

[4. Membuat Format Uang 56](#_sothprxarayn)

[5. Membuat Method edit() dan Method update() 58](#_4fii9pa4i27h)

[6. Membuat Method show() 60](#_c1aef4i3a2)

[7. Membuat Method destroy() 62](#_s0225xruf7td)

[E. Manajemen User 63](#_d6uqqo91vjd5)

[1. Membuat controller UserController 63](#_jiayco5bu5ea)

[2. Membuat Route 63](#_hss5inpzvff9)

[3. Membuat Tampilan Manajemen User 63](#_x46g40t88hmq)

[4. Membuat method store() 67](#_okje4oejo2u)

[F. Autentikasi User 69](#_1yfvftnfbt7k)

[1. Menyiapkan Form Login 69](#_spdtpns98fbl)

[2. Menambahkan Controller LoginController 72](#_mqn8fad9mc7s)

[3. Menambahkan Route Loginu 73](#_8hvfupdiqk4k)

[4. Memperbaiki Route 74](#_oimm98ljgxsm)

[5. Menambahkan Method Logout 76](#_s8aa6g2msiik)

[G. Penjualan 78](#_4b82llhh6xvf)

[1. Membuat Komponen 78](#_vmx80sa23tsf)

[2. Membuat seluruh relasi 86](#_jdasergocgyh)

[3. Memperbaiki komponen livewire 89](#_cgsbfxoujnnc)

[4. Menambahkan route 92](#_xdt53zwozkm8)

[5. Update Stok dan Cetak Receipt 94](#_t462lwmokvxm)

[H. Laporan Stok Barang 97](#_r61pmsk6121g)

[I. Laporan Harian 97](#_ag79hkac81h2)

[J. Laporan Bulanan 97](#_27jr33tbofar)

[K. Laporan Tahunan 97](#_af3haeufg9m0)

[**IV. Perbaikan Project 98**](#_cob48ze4dzqt)

[A. Memperbaiki Tabel Product 98](#_1gulo2ajcyc4)

[B. Mencetak Barcode 98](#_u86yp8hqymsl)

[C. Menambahkan Discount bagi Customer 98](#_9yw180po2qmh)

[D. Memperbaiki Tampilan Dashboard 98](#_e596jv8x0o4b)

# 

# Penting untuk dibaca

## Pengetahuan Awal

Jangan memulai menggunakan atau mengerjakan Modul ini jika belum memiliki pengetahuan di bawah berikut!

1. Dasar pemrograman php
2. Pemrograman Berorientasi Objek
3. Konsep MVC pada Framework Laravel
4. Membuat CRUD menggunakan Framework Laravel
5. Menggunakan Version Control System

Silahkan mengulangi kembali materi pelajaran di atas jika anda merasa belum menguasainya!

## Petunjuk Pengumpulan Tugas

1. Setelah melakukan instalasi project pada Bab II Persiapan, segera jadikan folder tersebut menjadi repository menggunakan Git. (Lakukan git init)
2. Lakukan git add . dan git commit -m “Awal” sebagai penanda tahap awal, agar tugas mudah di tracking perkembangannya
3. Hubungkan git dengan github **atau** VCS Online lainnya.
4. Lakukan git push.
5. Buatlah **Bab II Sub Bab Membuat Database**, setelah selesai lakukan commit kembali, dengan pesan Judul Sub-Bab. (Contoh : git commit -am “Membuat Database”)
6. Cek pada phpmyadmin.
7. Screenshot hasil lalu tempelkan (paste) pada dokumen laporan yang terletak didalam folder project post. (Buat dokumen dengan judul “Laporan”, lalu tempelkan hasil screenshot)
8. Lakukan commit kembali, dengan pesan “Membuat Laporan”
9. **Lakukan proses tersebut setiap selesai mengerjakan 1 Sub Bab**
10. Sebelum pulang atau mengakhiri pelajaran setiap harinya, lakukan git push, agar pekerjaan yang telah dilakukan akan tersimpan di Github atau VCS online lainnya dan mudah untuk di cek perkembangannya.
11. Kumpulkan link repository pada classroom.

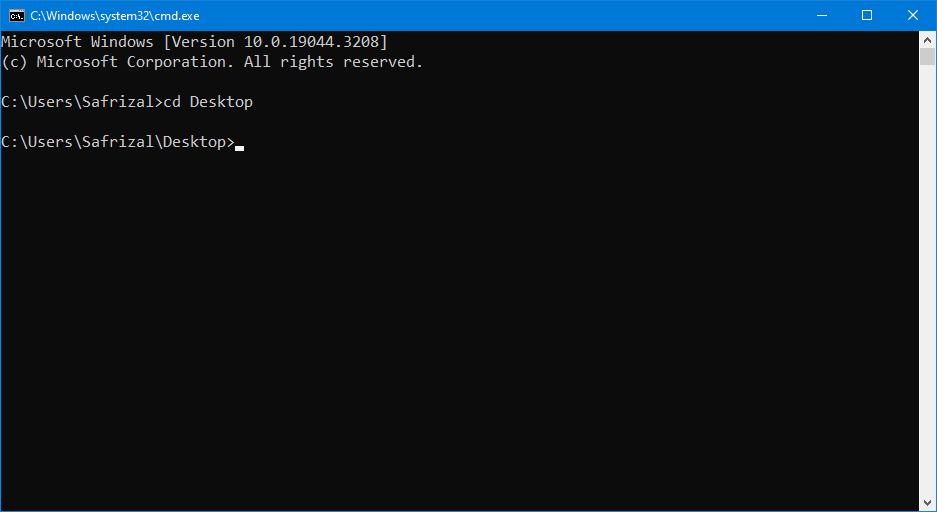
# 

# Persiapan

## Instalasi Project Laravel

1. Pastikan Aplikasi yang dibutuhkan sudah terinstal dengan baik. Aplikasi yang dibutuhkan yaitu, XAMPP dengan PHP versi minimal 8.1, composer, text editor
2. Pastikan pada saat proses instalasi laravel, sudah terkoneksi ke Internet
3. Buka command prompt/shell/git bash kemudian arahkan ke folder/lokasi instalasi project

Contoh : Saya akan membuat projectPOS pada Desktop

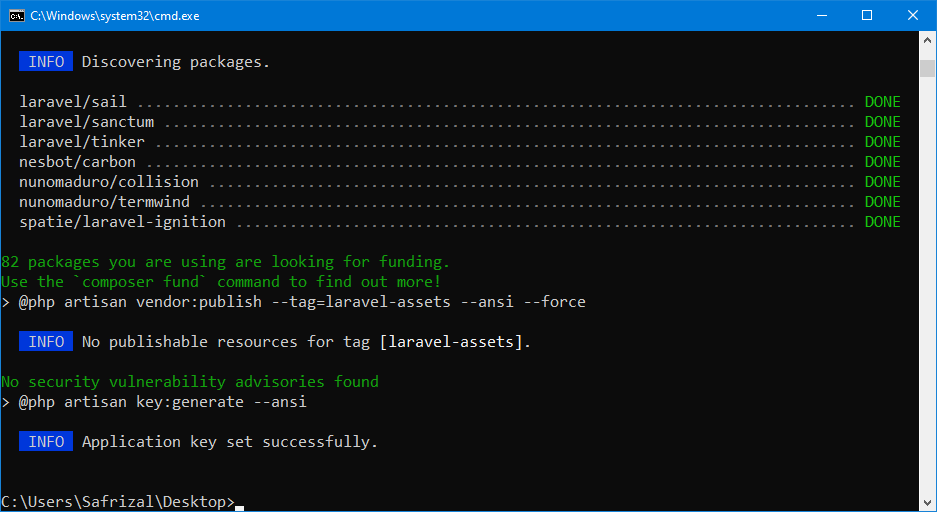


1. Lakukan instalasi laravel pada folder yang diinginkan dengan cara mengetikkan perintah

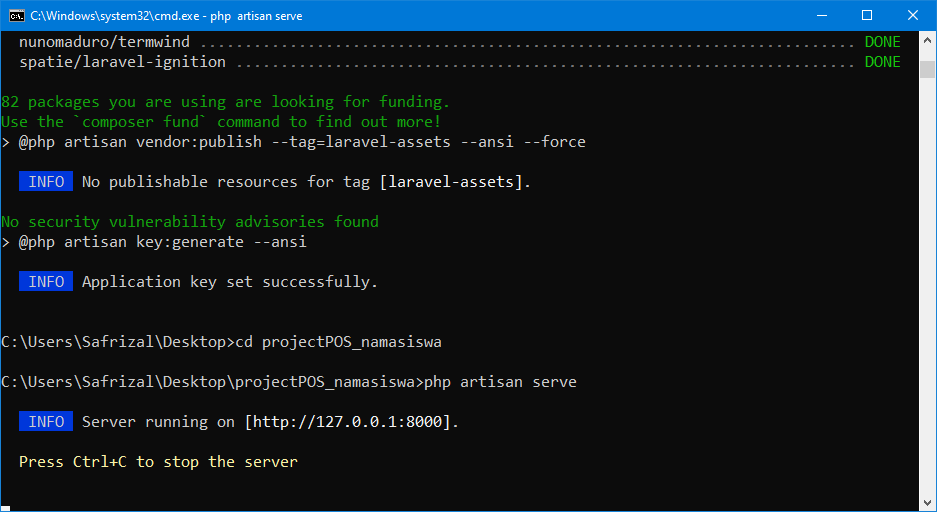
composer create-project laravel/laravel projectPOS\_namasiswa

Catatan:namasiswa diganti sesuai dengan nama masing-masing, hindari penggunaan spasi, ganti spasi dengan underscore atau menggunakan camelCase.

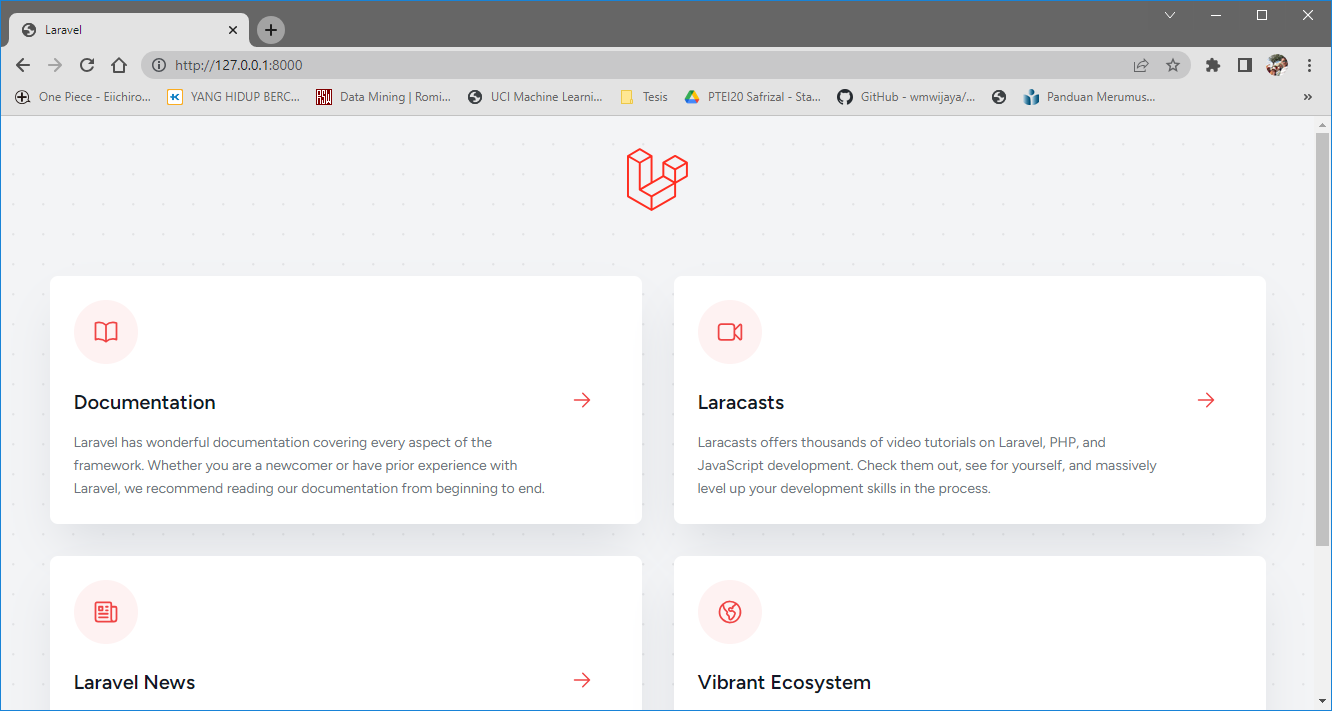
1. Tunggu proses instalasi hingga selesai, jika sudah selesai maka tampilannya akan seperti berikut!



1. Buka aplikasi text editor kesayangan kita semua, yaitu VS Code. Buka folder **projectPOS**
2. Jalankan php artisan serve pada project tersebut, bisa dari terminal yang ada di VS Code maupun dari Command Prompt



1. Buka browser kesayangan kita semua, dan buka alamat 127.0.0.1:8000 pada web address. Berikut tampilannya ➕

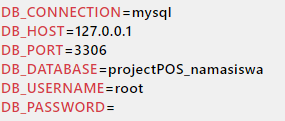


1. Horay, selanjutnya kita masuk ke tahap membuat database….

## Membuat Database

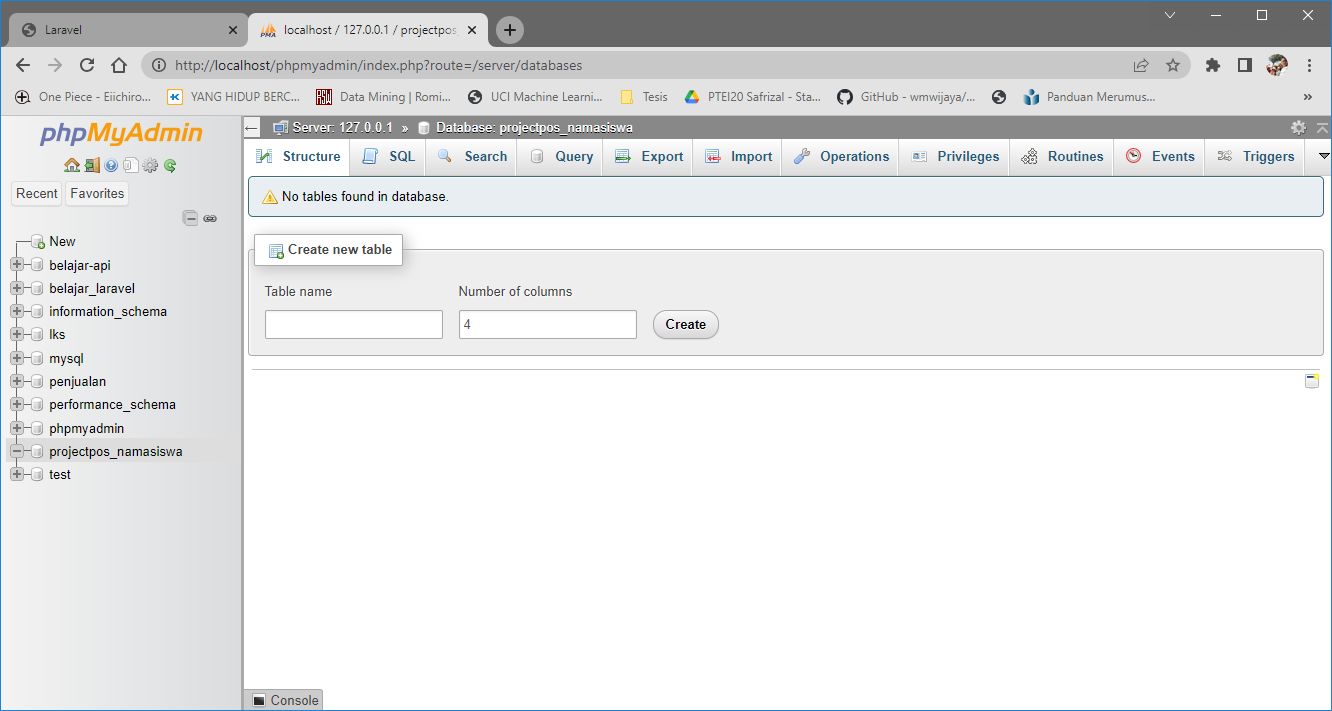
### Konfigurasi Database

1. Ubah file **.env** seperti berikut!



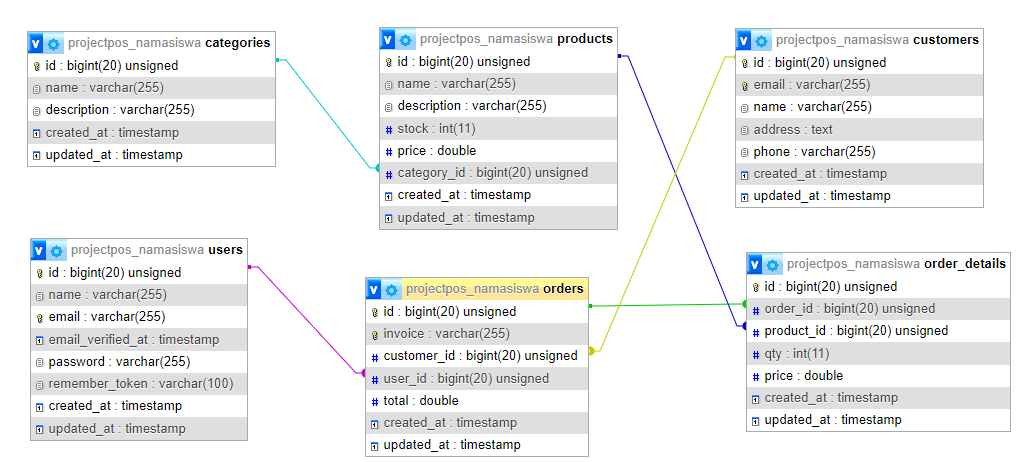
1. Buka aplikasi XAMPP, jalankan Apache dan MySQL
2. Buka localhost/phpmyadmin pada browser dan buat sebuah database dengan nama **projectPOS\_namasiswa**

Catatan : namasiswa diganti sesuai dengan nama masing-masing, hindari penggunaan spasi, ganti spasi dengan underscore atau menggunakan camelCase.



### Membuat Migration Database

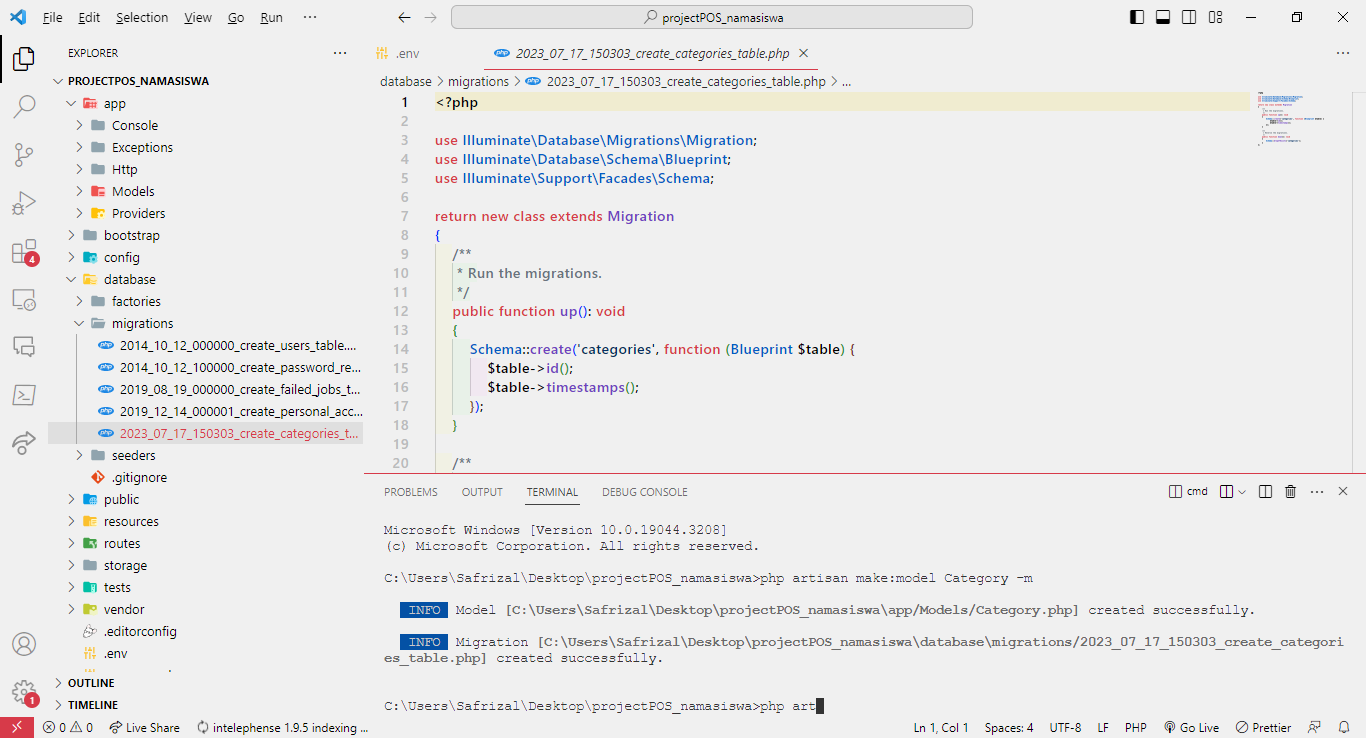
1. Schema database yang akan kita buat akan seperti berikut!



1. Buka terminal pada VS Code, lalu ketikkan perintah berikut!

php artisan make:model Category -m

Perintah tersebut akan membuat sebuah model dengan nama Category, dan juga akan membuat sebuah migration secara otomatis.



1. Buka file migration pada folder database/migration di vscode kemudian temukan file **create\_categories\_table.php**
2. Ubah public function up() menjadi seperti berikut:

public function up(): void

{

Schema::create('categories', function (Blueprint $table) {

$table->id();

$table->string('name');

$table->string('description')->nullable();

$table->timestamps();

});

}

1. Selanjutnya kita akan membuat migration untuk **product**. Tulis perintah berikut pada terminal VS Code:

php artisan make:model Product -m

1. Buka file create\_products\_table.php yang terdapat dalam folder migration.
2. Kemudian ubah seperti berikut:

public function up(): void

{

Schema::create('products', function (Blueprint $table) {

$table->id();

$table->string('name');

$table->string('description')->nullable();

$table->integer('stock');

$table->double('price');

$table->unsignedBigInteger('category\_id');

$table->timestamps();

});

Schema::table('products', function (Blueprint $table) {

$table->foreign('category\_id')->references('id')->on('categories')->onUpdate('cascade')->onDelete('cascade');

});

}

1. Kemudian untuk function down() rubah seperti berikut:

public function down(): void

{

Schema::table('products', function (Blueprint $table) {

$table->dropForeign('products\_category\_id\_foreign');

});

Schema::table('products', function (Blueprint $table) {

$table->dropIndex('products\_category\_id\_foreign');

});

Schema::dropIfExists('products');

}

1. Selanjutnya kita akan membuat migration untuk **customer**. Tulis perintah berikut pada terminal VS Code:

php artisan make:model Customer -m

1. Ubah file create\_customer\_table.php menjadi seperti berikut:

public function up(): void

{

Schema::create('customers', function (Blueprint $table) {

$table->id();

$table->string('email')->unique();

$table->string('name');

$table->text('address');

$table->string('phone');

$table->timestamps();

});

}

1. Selanjutnya buat migration untuk **order**. Tulis perintah berikut pada terminal VS Code:

php artisan make:model Order -m

1. Ubah file migration order menjadi seperti berikut:

public function up(): void

{

Schema::create('orders', function (Blueprint $table) {

$table->id();

$table->string('invoice')->unique();

$table->bigInteger('customer\_id')->unsigned();

$table->bigInteger('user\_id')->unsigned();

$table->double('total');

$table->timestamps();

});

Schema::table('orders',function(Blueprint $table){

$table->foreign('customer\_id')->references('id')->on('customers')

->onUpdate('cascade')->onDelete('cascade');

$table->foreign('user\_id')->references('id')->on('users')

->onUpdate('cascade')->onDelete('cascade');

});

}

1. Untuk function down() menjadi seperti berikut:

public function down(): void

{

Schema::table('orders', function(Blueprint $table) {

$table->dropForeign('orders\_customer\_id\_foreign');

});

Schema::table('orders', function(Blueprint $table) {

$table->dropIndex('orders\_customer\_id\_foreign');

});

Schema::table('orders', function(Blueprint $table) {

$table->dropForeign('orders\_user\_id\_foreign');

});

Schema::table('orders', function(Blueprint $table) {

$table->dropIndex('orders\_user\_id\_foreign');

});

Schema::dropIfExists('orders');

}

1. Selanjutnya buat migration untuk **order\_detail**. Tulis perintah berikut pada terminal VS Code:

php artisan make:model Order\_detail -m

1. Lalu ubah file *migration*-nya menjadi seperti berikut:

public function up(): void

{

Schema::create('order\_details', function (Blueprint $table) {

$table->id();

$table->bigInteger('order\_id')->unsigned();

$table->bigInteger('product\_id')->unsigned();

$table->integer('qty');

$table->double('price');

$table->timestamps();

});

Schema::table('order\_details', function(Blueprint $table) {

$table->foreign('order\_id')->references('id')->on('orders')

->onUpdate('cascade')

->onDelete('cascade');

$table->foreign('product\_id')->references('id')->on('products')

->onUpdate('cascade')

->onDelete('cascade');

});

}

1. Kemudian untuk function down() menjadi seperti berikut:

public function down(): void

{

Schema::table('order\_details', function(Blueprint $table) {

$table->dropForeign('order\_details\_order\_id\_foreign');

});

Schema::table('order\_details', function(Blueprint $table) {

$table->dropIndex('order\_details\_order\_id\_foreign');

});

Schema::table('order\_details', function(Blueprint $table) {

$table->dropForeign('order\_details\_product\_id\_foreign');

});

Schema::table('order\_details', function(Blueprint $table) {

$table->dropIndex('order\_details\_product\_id\_foreign');

});

Schema::dropIfExists('order\_details');

}

1. Untuk membuat database ketik perintah

php artisan migrate

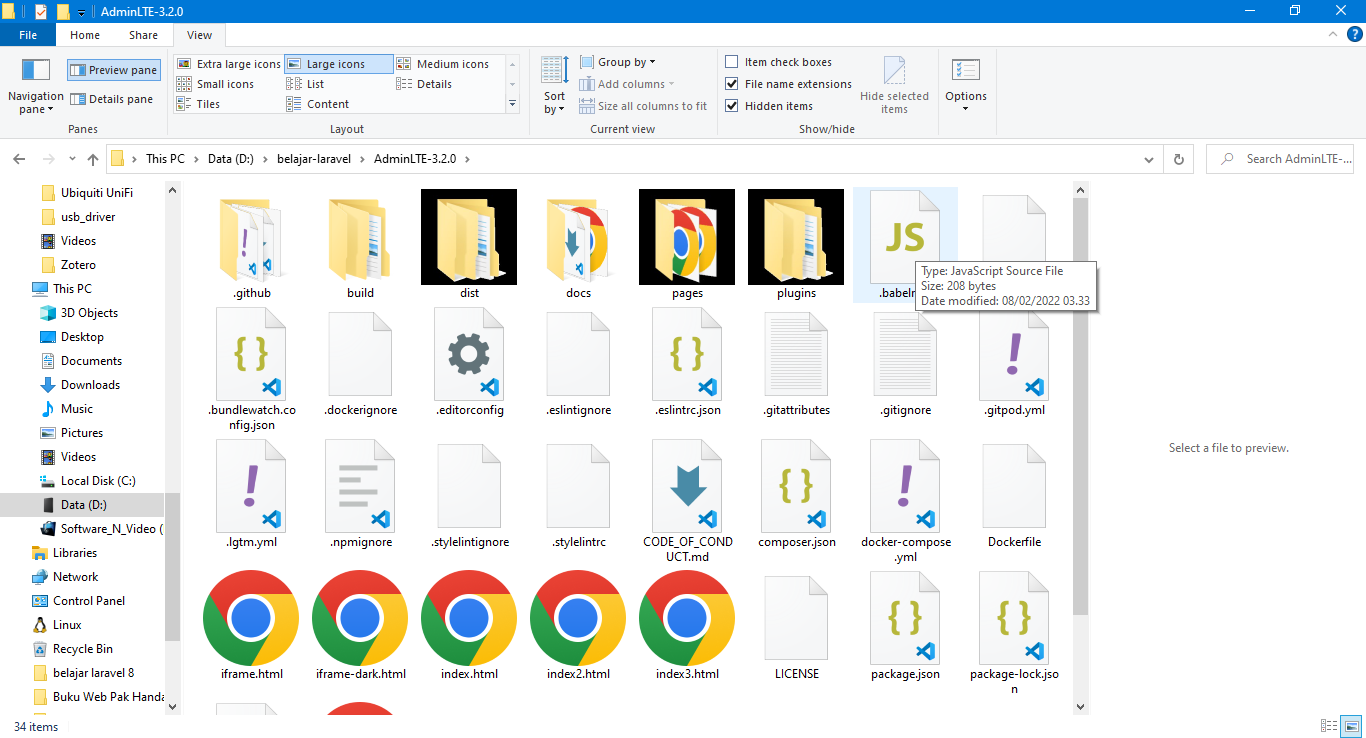
1. Cek hasilnya di localhost/phpmyadmin

# Membuat Project

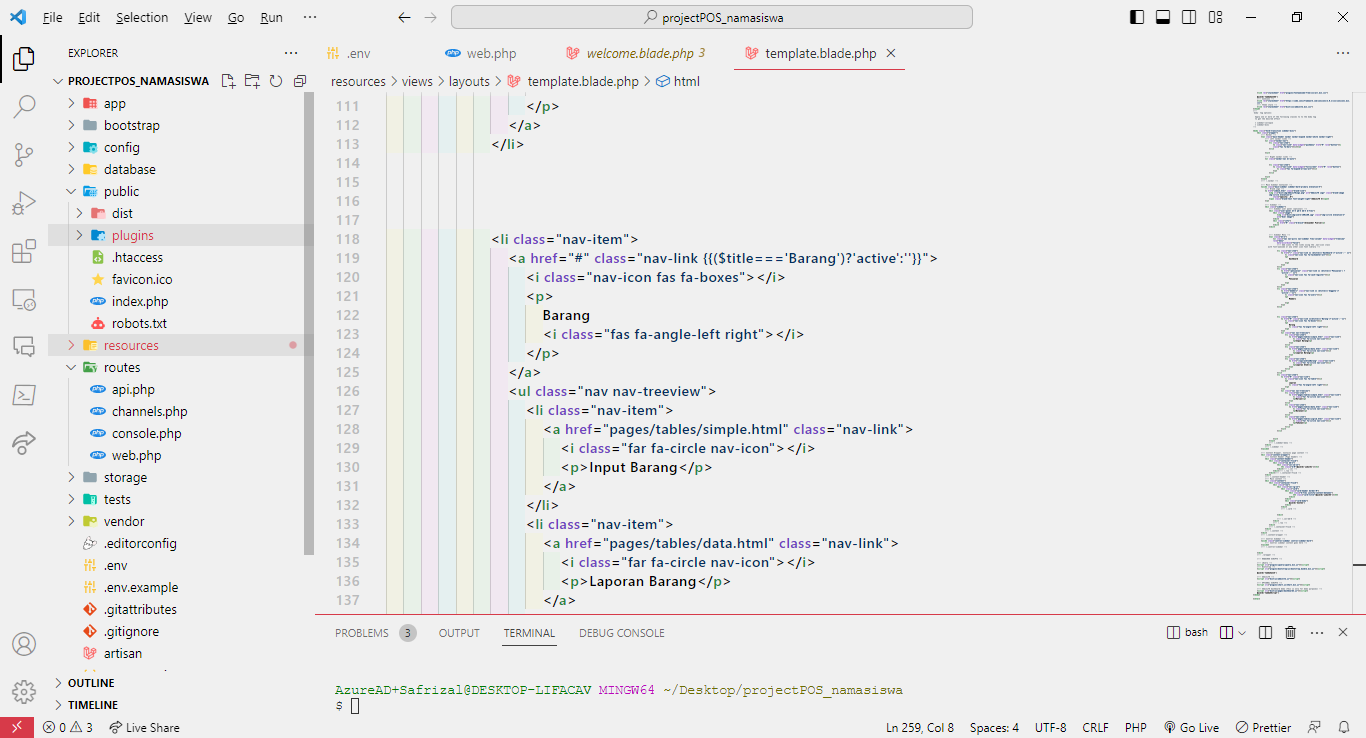
## Tampilan Utama

Untuk mempermudah kita dalam membuat tampilan aplikasi POS ini maka kita akan menggunakan Template Admin gratis. Pada project ini kita akan menggunakan Admin LTE sebagai Template utamanya.

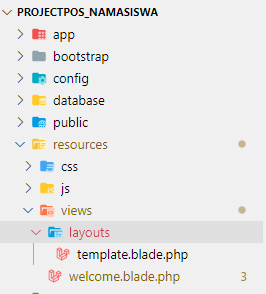
1. Download Admin LTE pada alamat berikut: <https://github.com/ColorlibHQ/AdminLTE/archive/refs/tags/v3.2.0.zip>
2. Ekstrak hasil download Admin LTE
3. Copy 2 buah folder dari Admin LTE yaitu **plugins** dan **dist** ke dalam folder **public** yang terdapat di project yang telah kita kerjakan.



1. Sehingga di dalam folder public project kita terdapat 2 buah folder tambahan yaitu plugins dan dist.



1. Selanjutnya buat folder baru dengan nama **layouts** di dalam folder **resources/views**.
2. Lalu buatlah file baru dengan nama **template.blade.php**.



1. Tuliskan script berikut pada file **template.blade.php**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>{{$title}}</title>

<!-- Google Font: Source Sans Pro -->

<link rel="stylesheet"

href="https://fonts.googleapis.com/css?family=Source+Sans+Pro:300,400,400i,700&display=fallback">

<!-- Font Awesome Icons -->

<link rel="stylesheet" href="{{ asset('plugins/fontawesome-free/css/all.min.css') }}">

@yield('tambahanCSS')

<!-- IonIcons -->

<link rel="stylesheet" href="https://code.ionicframework.com/ionicons/2.0.1/css/ionicons.min.css">

<!-- Theme style -->

<link rel="stylesheet" href="{{ asset('dist/css/adminlte.min.css')}}">

</head>

<body class="hold-transition sidebar-mini">

<div class="wrapper">

<!-- Navbar -->

<nav class="main-header navbar navbar-expand navbar-white navbar-light">

<!-- Left navbar links -->

<ul class="navbar-nav">

<li class="nav-item">

<a class="nav-link" data-widget="pushmenu" href="#" role="button"><i class="fas fa-bars"></i></a>

</li>

</ul>

<!-- Right navbar links -->

<ul class="navbar-nav ml-auto">

<li class="nav-item">

<a class="nav-link" data-widget="fullscreen" href="#" role="button">

<i class="fas fa-expand-arrows-alt"></i>

</a>

</li>

</ul>

</nav>

<!-- /.navbar -->

<!-- Main Sidebar Container -->

<aside class="main-sidebar sidebar-dark-primary elevation-4">

<!-- Brand Logo -->

<a href="index3.html" class="brand-link">

<img src="{{ asset('dist/img/AdminLTELogo.png')}}" alt="AdminLTE Logo"

class="brand-image img-circle elevation-3" style="opacity: .8">

<span class="brand-text font-weight-light">POS</span>

</a>

<!-- Sidebar -->

<div class="sidebar">

<!-- Sidebar user panel (optional) -->

<div class="user-panel mt-3 pb-3 mb-3 d-flex">

<div class="image">

<img src="{{ asset('dist/img/user2-160x160.jpg')}}" class="img-circle elevation-2"

alt="User Image">

</div>

<div class="info">

<a href="#" class="d-block">Safrizal</a>

</div>

</div>

<!-- Sidebar Menu -->

<nav class="mt-2">

<ul class="nav nav-pills nav-sidebar flex-column" data-widget="treeview" role="menu"

data-accordion="false">

<!-- Add icons to the links using the .nav-icon class

with font-awesome or any other icon font library -->

<li class="nav-item">

<a href="/" class="nav-link {{ ($title==='Dashboard')?'active':'' }}">

<i class="nav-icon fas fa-tachometer-alt"></i>

<p>

Dashboard

</p>

</a>

</li>

<li class="nav-item">

<a href="{{ route('kategori.index')}}"

class="nav-link {{ ($title==='Kategori') ? 'active':'' }}">

<i class="nav-icon fas fa-th-large"></i>

<p>

Kategori

</p>

</a>

</li>

<li class="nav-item">

<a href="#" class="nav-link {{($title==='Produk')?'active':''}}">

<i class="nav-icon fas fa-boxes"></i>

<p>

Produk

<i class="fas fa-angle-left right"></i>

</p>

</a>

<ul class="nav nav-treeview">

<li class="nav-item">

<a href="#" class="nav-link">

<i class="far fa-circle nav-icon"></i>

<p>Input Produk</p>

</a>

</li>

<li class="nav-item">

<a href="#" class="nav-link">

<i class="far fa-circle nav-icon"></i>

<p>Laporan Produk</p>

</a>

</li>

</ul>

</li>

<li class="nav-item">

<a href="#" class="nav-link {{ ($title==='Customer')?'active':''}}">

<i class="nav-icon fas fa-users"></i>

<p>

Customer

</p>

</a>

</li>

<li class="nav-item">

<a href="#" class="nav-link {{ ($title==='Users')?'active':''}}">

<i class="nav-icon fas fa-users"></i>

<p>

Users

</p>

</a>

</li>

<li class="nav-item">

<a href="#" class="nav-link {{ ($title==='Pembelian') ? 'active':'' }}">

<i class="nav-icon fas fa-cash-register"></i>

<p>

Pembelian

</p>

</a>

</li>

<li class="nav-item">

<a href="#" class="nav-link">

<i class="nav-icon fas fa-table"></i>

<p>

Laporan

<i class="fas fa-angle-left right"></i>

</p>

</a>

<ul class="nav nav-treeview">

<li class="nav-item">

<a href="pages/tables/simple.html" class="nav-link">

<i class="far fa-circle nav-icon"></i>

<p>Harian</p>

</a>

</li>

<li class="nav-item">

<a href="pages/tables/data.html" class="nav-link">

<i class="far fa-circle nav-icon"></i>

<p>Bulanan</p>

</a>

</li>

<li class="nav-item">

<a href="pages/tables/jsgrid.html" class="nav-link">

<i class="far fa-circle nav-icon"></i>

<p>Tahunan</p>

</a>

</li>

</ul>

</li>

</ul>

</nav>

<!-- /.sidebar-menu -->

</div>

<!-- /.sidebar -->

</aside>

<!-- Content Wrapper. Contains page content -->

<div class="content-wrapper">

<!-- Content Header (Page header) -->

<div class="content-header">

<div class="container-fluid">

<div class="row mb-2">

<div class="col-sm-12">

<h1 class="m-0">@yield('judulh1')</h1>

</div><!-- /.col -->

</div><!-- /.row -->

</div><!-- /.container-fluid -->

</div>

<!-- /.content-header -->

<!-- Main content -->

<div class="content">

<div class="container-fluid">

<div class="row">

@yield('konten')

</div>

<!-- /.row -->

</div>

<!-- /.container-fluid -->

</div>

<!-- /.content -->

</div>

<!-- /.content-wrapper -->

<!-- Control Sidebar -->

<aside class="control-sidebar control-sidebar-dark">

<!-- Control sidebar content goes here -->

</aside>

<!-- /.control-sidebar -->

</div>

<!-- ./wrapper -->

<!-- REQUIRED SCRIPTS -->

<!-- jQuery -->

<script src="{{ asset('plugins/jquery/jquery.min.js')}}"></script>

<!-- Bootstrap -->

<script src="{{ asset('plugins/bootstrap/js/bootstrap.bundle.min.js')}}"></script>

@yield('tambahanJS')

<!-- AdminLTE -->

<script src="{{ asset('dist/js/adminlte.js')}}"></script>

<!-- OPTIONAL SCRIPTS -->

<script src="{{ asset('plugins/chart.js/Chart.min.js')}}"></script>

@yield('tambahScript')

</body>

</html>

Catatan : script di atas bisa didapatkan dari file index.php yang terdapat pada folder Admin LTE, selanjutnya dimodifikasi sehingga menghasilkan tampilan seperti di atas.

1. Untuk menguji template selanjutnya kita buat file baru dengan nama **dashboard.blade.php** pada folder **resources/views**.

@extends('layouts.template')

@section('judulh1','Admin - Dashboard')

@section('konten')

<div class="col-lg-12">

<div class="card">

<div class="card-header border-0">

<div class="d-flex justify-content-between">

<h3 class="card-title">Dashboard</h3>

</div>

</div>

<div class="card-body">

<h3>Halaman Dashboard</h3>

</div>

</div>

</div>

@endsection

1. Selanjutnya tambahkan script berikut pada routes/web.php.

Route::get('/', function () {

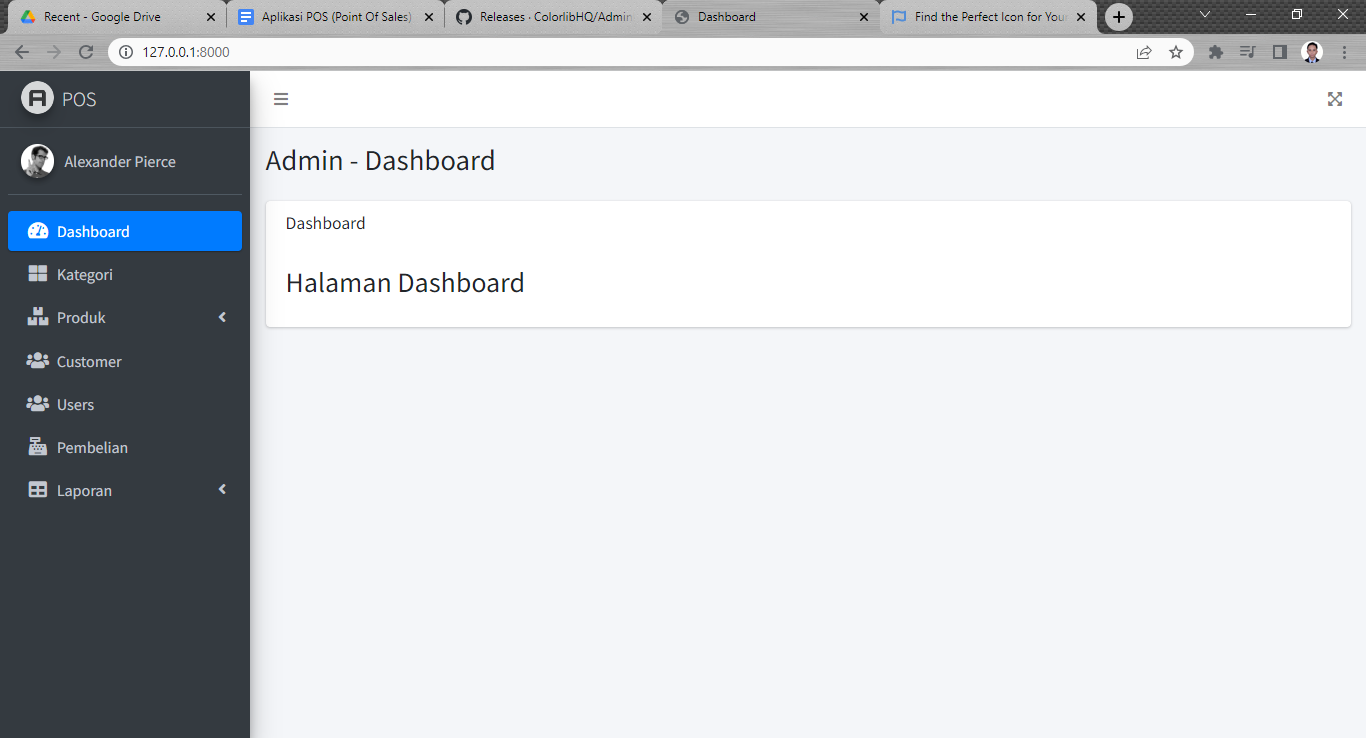
return view('dashboard',[

"title"=>"Dashboard"

]);

});

1. Jalankan server kemudian silahkan buka alamat http://127.0.0.1:8000/



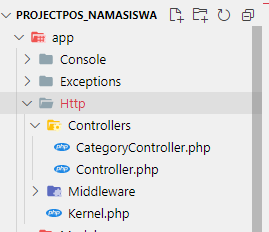
## Manajemen Kategori

### Membuat CategoryController

1. Jalankan perintah berikut pada terminal kesayangan Anda (pastikan terminal sudah berada di dalam folder project)

php artisan make:controller CategoryController

1. Kemudian cek Folder **App\Http\Controllers**. Apakah sudah terdapat controller baru seperti berikut!



1. Tambahkan method index pada CategoryController.php

use App\Models\Category;

class CategoryController extends Controller

{

//

public function index()

{

return view('category.category', [

"title" => "Kategori",

"data" => Category::all()

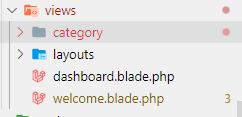
]);

}

}

Catatan: perhatikan script "data" => Category::all(). Category tersebut mengacu pada **Model Class Category**, dan fungsi all() yang sudah disediakan oleh laravel. Tujuan dari fungsi all() adalah untuk menampilkan seluruh data yang terdapat pada database. Kemudian perhatikan view yang digunakan 'category.category', makna dari script ini adalah terdapat file category.blade.php di dalam folder views/category.

1. Buat folder baru didalam folder views dengan nama category. Sehingga struktur folder pada views menjadi seperti berikut.



1. Di dalam folder category buat sebuah file dengan nama **category.blade.php**.
2. Lalu tambahkan script berikut pada file **category.blade.php**:

@extends('layouts.template')

@section('tambahanCSS')

<!-- DataTables -->

<link rel="stylesheet" href="plugins/datatables-bs4/css/dataTables.bootstrap4.min.css">

<link rel="stylesheet" href="plugins/datatables-responsive/css/responsive.bootstrap4.min.css">

<link rel="stylesheet" href="plugins/datatables-buttons/css/buttons.bootstrap4.min.css">

<!-- Toastr -->

<link rel="stylesheet" type="text/css" href="//cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/css/toastr.css">

@endsection

@section('judulh1','Admin - Category')

@section('judulh3','Category')

@section('konten')

<div class="col-md-4">

<div class="card card-success">

<div class="card-header">

<h3 class="card-title">Input Category</h3>

</div>

<!-- /.card-header -->

<!-- form start -->

<form action="{{ route('kategori.store') }}" method="POST">

@csrf

<div class=" card-body">

<div class="form-group">

<label for="name">Nama Kategori</label>

<input type="text" class="form-control" id="name" name="name" placeholder=" Nama Kategori">

</div>

<div class="form-group">

<label for="description">Deskripsi</label>

<textarea id="description" name="description" class=" form-control" rows="4"

placeholder="Deskripsi"></textarea>

</div>

</div>

<!-- /.card-body -->

<div class="card-footer">

<button type="submit" class="btn btn-success float-right">Simpan</button>

</div>

</form>

</div>

</div>

<div class="col-md-8">

<div class="card card-info">

<div class="card-header">

<h3 class="card-title">Data Category</h3>

</div>

<!-- /.card-header -->

<div class="card-body">

<table id="example1" class="table table-bordered table-striped ">

<thead>

<tr>

<th>No</th>

<th>Kategori</th>

<th>Deskripsi</th>

<th>Aksi</th>

</tr>

</thead>

<tbody>

@foreach($data as $dt)

<tr>

<td>{{ $loop->iteration }}</td>

<td>{{ $dt->name }}</td>

<td>{{ $dt->description }}</td>

<td>

<a type="button" class="btn btn-warning" href="{{ route('kategori.edit',$dt->id) }}">

<i class=" fas fa-edit"></i>

</a>

</td>

</tr>

@endforeach

</tbody>

</table>

</div>

</div>

</div>

@endsection

@section('tambahanJS')

<!-- DataTables & Plugins -->

<script src="plugins/datatables/jquery.dataTables.min.js"></script>

<script src="plugins/datatables-bs4/js/dataTables.bootstrap4.min.js"></script>

<script src="plugins/datatables-responsive/js/dataTables.responsive.min.js"></script>

<script src="plugins/datatables-responsive/js/responsive.bootstrap4.min.js"></script>

<script src="plugins/datatables-buttons/js/dataTables.buttons.min.js"></script>

<script src="plugins/datatables-buttons/js/buttons.bootstrap4.min.js"></script>

<script src="plugins/jszip/jszip.min.js"></script>

<script src="plugins/pdfmake/pdfmake.min.js"></script>

<script src="plugins/pdfmake/vfs\_fonts.js"></script>

<script src="plugins/datatables-buttons/js/buttons.html5.min.js"></script>

<script src="plugins/datatables-buttons/js/buttons.print.min.js"></script>

<script src="plugins/datatables-buttons/js/buttons.colVis.min.js"></script>

<!-- Toastr -->

<script src="https://cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/toastr.min.js"></script>

@endsection

@section('tambahScript')

<script>

$(function() {

$("#example1").DataTable({

"responsive": true,

"lengthChange": true,

"autoWidth": false,

"responsive": true,

}).buttons().container().appendTo('#example1\_wrapper .col-md-6:eq(0)');

});

@if($message = Session::get('success'))

toastr.success("{{ $message}}");

@elseif($message = Session::get('updated'))

toastr.warning("{{ $message}}");

@endif

</script>

@endsection

### Membuat Route Kategor

1. Tambahkan script berikut pada **routes/web.php**:

Route::resource('kategori',CategoryController::class);

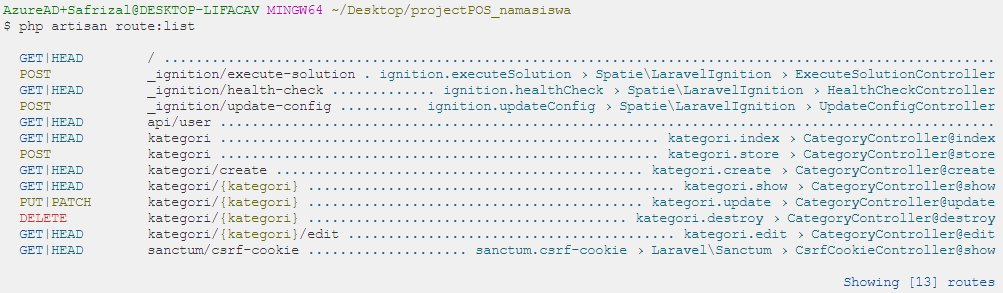
Jangan lupa untuk menambahkan script berikut pada bagian atas file route/web.php.

use App\Http\Controllers\CategoryController;

1. Lalu jalankan perintah berikut di terminal

php artisan route:list

Catatan: perintah tersebut digunakan untuk menampilkan seluruh daftar route yang ada pada project



**Perhatikan** untuk route yang dimiliki oleh kategori ada 7 method, yaitu index,store,create,show,update,destroy dan edit. Namun pada manajemen kategori tidak kita gunakan semua. Yang akan kita gunakan adalah index, store, update,edit. Method **index()** adalah method default, atau method yang secara otomatis akan diakses pertama kali. Method **store()** digunakan untuk insert data ke dalam database. Method **edit()** digunakan untuk mengambil data yang akan diedit. Sementara Method **update()** digunakan untuk merubah data didalam database.

1. Karena tidak semua method akan digunakan maka kita perlu merubah routes/web.php menjadi seperti berikut:

Route::resource('kategori',CategoryController::class)

->except('show','destroy','create');

Catatan: code except bertujuan agar method show,destroy dan create tidak kita masukkan kedalam list method yang digunakan.

1. Silahkan ulangi kembali perintah:

php artisan route:list

Maka kita tidak akan melihat method show, destroy dan create pada daftar method di dalam kategori.

1. Akses route kategori melalui browser, pastikan tidak muncul error

### **Membuat Method store()**

1. Buka kembali CategoryController tambahkan script berikut pada bagian atas file:

use Illuminate\Http\Request;

use Illuminate\Http\RedirectResponse;

1. Buat method store() pada CategoryController:

public function store(Request $request): RedirectResponse

{

$request->validate([

"name" => "required",

"description" => "nullable",

]);

Category::create($request->all());

return redirect()->route('kategori.index')->with('success','Kategori Berhasil Ditambahkan.');

}

1. Buka **Models\Category.php** lalu ubah sehingga script-nya menjadi seperti berikut:

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class Category extends Model

{

use HasFactory;

protected $fillable = [

'name',

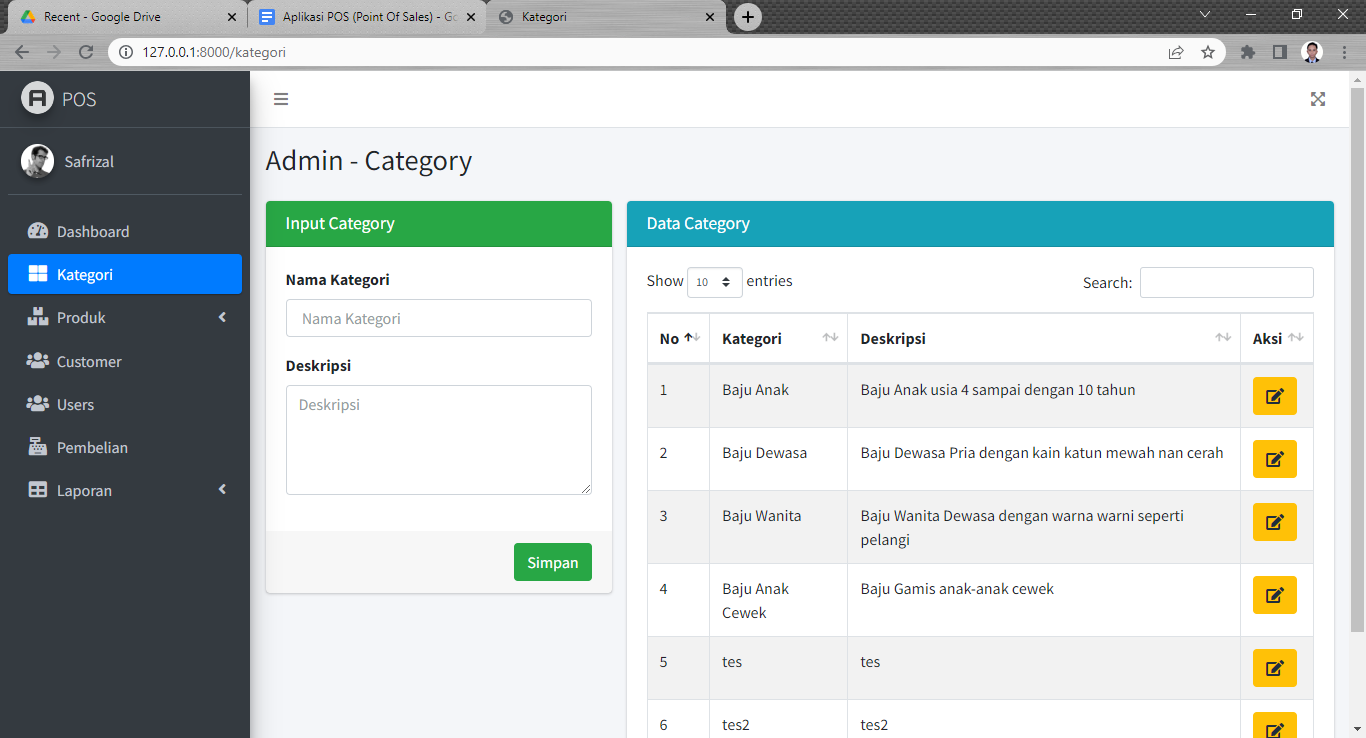
'description',

];

}

Catatan: variabel **$fillable** berarti kita mendeklarasikan bahwa field yang dapat diisi adalah name dan description, selain dari pada itu akan di handle oleh Eloquent ORM milik laravel.

1. Uji coba method store dengan menambahkan data.



### Membuat Method edit() dan update()

1. Buka kembali CategoryController tambahkan script berikut pada bagian atas file:

use Illuminate\View\View;

1. Buat Method edit()

public function edit(Category $kategori): View

{

return view('category.edit', compact('kategori'))->with(["title" => "Edit Kategori"]);

}

1. Buat Method update()

public function update(Request $request, Category $kategori):RedirectResponse

{

$request->validate([

"name" => "required",

"description" => "required",

]);

$kategori->update($request->all());

return redirect()->route('kategori.index')->with('updated','Kategori Berhasil Diubah.');

}

1. Membuat **edit.blade.php** di dalam folder **views/category**.

@extends('layouts.template')

@section('judulh1','Admin - Category')

@section('judulh3','Edit Category')

@section('konten')

<div class="col-md-6">

@if ($errors->any())

<div class="alert alert-danger">

<strong>Whoops!</strong> Ada Yang salah.<br><br>

<ul>

@foreach ($errors->all() as $error)

<li>{{ $error }}</li>

@endforeach

</ul>

</div>

@endif

<div class="card card-warning">

<div class="card-header">

<h3 class="card-title">Edit Category</h3>

</div>

<!-- /.card-header -->

<!-- form start -->

<form action="{{ route('kategori.update',$kategori->id) }}" method="POST">

@csrf

@method('PUT')

<div class=" card-body">

<div class="form-group">

<label for="name">Nama Kategori</label>

<input type="text" class="form-control" id="name" name="name" value="{{$kategori->name}}">

</div>

<div class="form-group">

<label for="description">Deskripsi</label>

<textarea id="description" name="description" class=" form-control"

rows="4"> {{ $kategori->description }} </textarea>

</div>

</div>

<!-- /.card-body -->

<div class="card-footer">

<button type="submit" class="btn btn-warning float-right">Simpan</button>

</div>

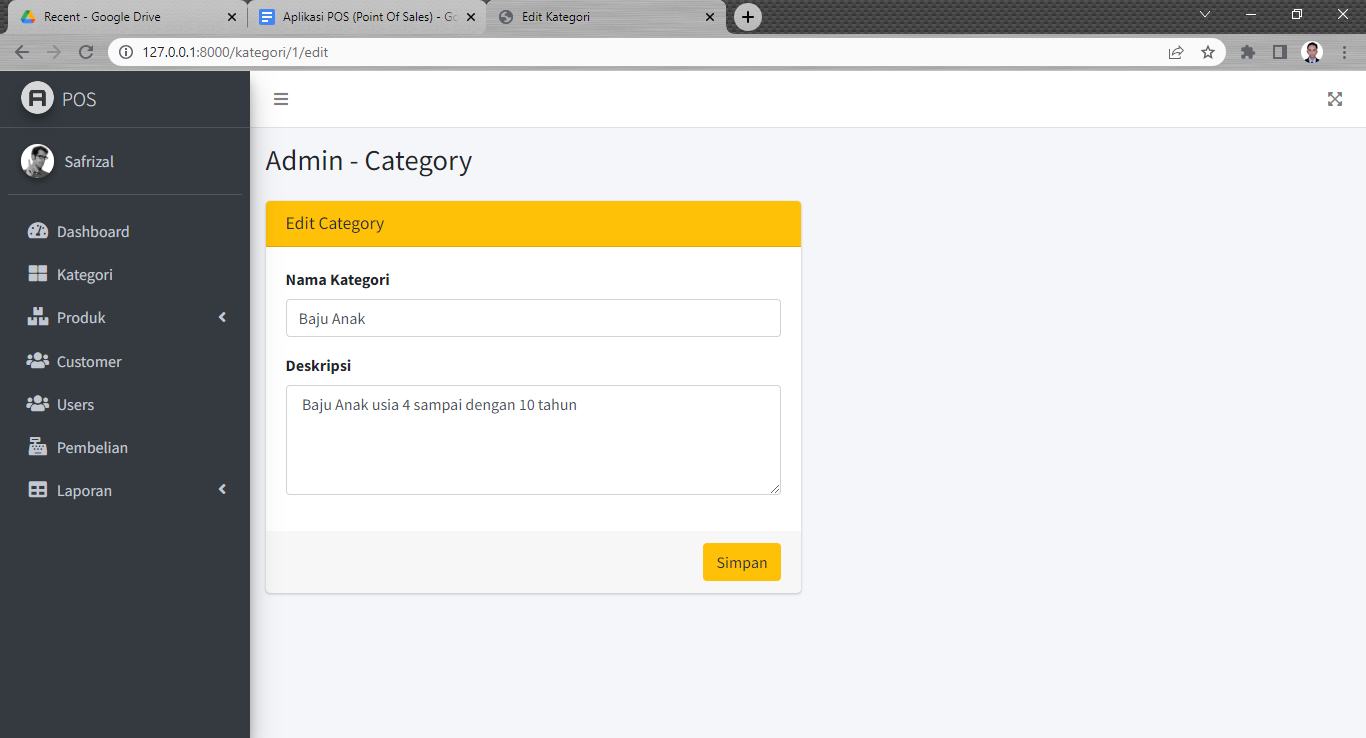
</form>

</div>

</div>

@endsection

1. Uji coba, untuk menambah data, menampilkan data yang akan diubah, dan merubah data. Pastikan seluruh proses dapat dikerjakan.



1. Jika ada yang bertanya, mengapa tidak ada tombol hapus, jawabannya adalah ketika kita memberikan method hapus, dan ternyata secara tak sengaja user nantinya menghapus sebuah record kategori maka seluruh produk yang memiliki kategori tersebut akan ikut terhapus

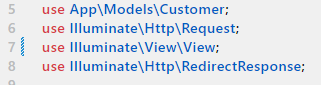
## Manajemen Customer

### Membuat Controller Customer

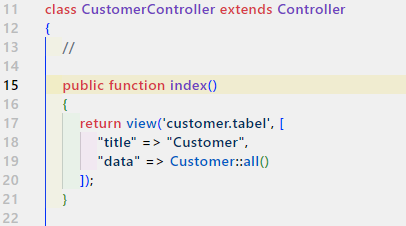
1. Buat controller dengan menggunakan php artisan

php artisan make:controller CustomerController

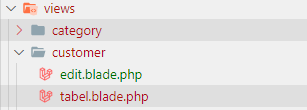
1. Tambahkan script berikut pada **CustomerController.php**

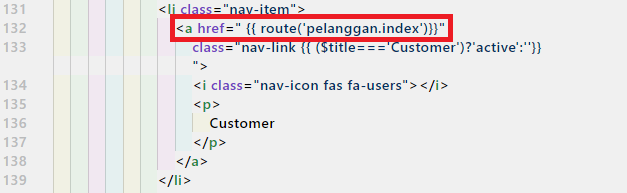


1. Buat function **index()** di dalam class CustomerController



1. Buat folder baru didalam views dengan nama customer
2. Buat file baru di dalam folder customer dengan nama **tabel.blade.php**

****

1. Buka file **views/layouts/template.blade.php**, lalu tambahkan script berikut pada menu customer.
2. Lalu tambahkan script berikut:

@extends('layouts.template')

@section('tambahanCSS')

<!-- DataTables -->

<link rel="stylesheet" href="plugins/datatables-bs4/css/dataTables.bootstrap4.min.css">

<link rel="stylesheet" href="plugins/datatables-responsive/css/responsive.bootstrap4.min.css">

<link rel="stylesheet" href="plugins/datatables-buttons/css/buttons.bootstrap4.min.css">

<!-- Toastr -->

<link rel="stylesheet" type="text/css" href="//cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/css/toastr.css">

@endsection

@section('judulh1','Admin - Customer')

@section('konten')

<div class="col-md-12">

<div class="card card-info">

<div class="card-header">

<h2 class="card-title">Data Customer</h2>

<a type="button" class="btn btn-success float-right" href="{{ route('pelanggan.create') }}">

<i class=" fas fa-plus"></i>Tambah Customer

</a>

</div>

<!-- /.card-header -->

<div class="card-body">

<table id="example1" class="table table-bordered table-striped ">

<thead>

<tr>

<th>No</th>

<th>Nama</th>

<th>No Telepon</th>

<th>Aksi</th>

</tr>

</thead>

<tbody>

@foreach($data as $dt)

<tr>

<td>{{ $loop->iteration }}</td>

<td>{{ $dt->name }}</td>

<td>{{ $dt->phone }}</td>

<td>

<div class="btn-group">

<a type="button" class="btn btn-warning" href="{{ route('pelanggan.edit',$dt->id) }}">

<i class=" fas fa-edit"></i>

</a>

<a type="button" class="btn btn-success" href="{{ route('pelanggan.show',$dt->id) }}">

<i class=" fas fa-eye"></i>

</a>

</div>

</td>

</tr>

@endforeach

</tbody>

</table>

</div>

</div>

</div>

@endsection

@section('tambahanJS')

<!-- DataTables & Plugins -->

<script src="plugins/datatables/jquery.dataTables.min.js"></script>

<script src="plugins/datatables-bs4/js/dataTables.bootstrap4.min.js"></script>

<script src="plugins/datatables-responsive/js/dataTables.responsive.min.js"></script>

<script src="plugins/datatables-responsive/js/responsive.bootstrap4.min.js"></script>

<script src="plugins/datatables-buttons/js/dataTables.buttons.min.js"></script>

<script src="plugins/datatables-buttons/js/buttons.bootstrap4.min.js"></script>

<script src="plugins/jszip/jszip.min.js"></script>

<script src="plugins/pdfmake/pdfmake.min.js"></script>

<script src="plugins/pdfmake/vfs\_fonts.js"></script>

<script src="plugins/datatables-buttons/js/buttons.html5.min.js"></script>

<script src="plugins/datatables-buttons/js/buttons.print.min.js"></script>

<script src="plugins/datatables-buttons/js/buttons.colVis.min.js"></script>

<!-- Toastr -->

<script src="https://cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/toastr.min.js"></script>

@endsection

@section('tambahScript')

<script>

$(function() {

$("#example1").DataTable({

"responsive": true,

"lengthChange": true,

"autoWidth": false,

"responsive": true,

}).buttons().container().appendTo('#example1\_wrapper .col-md-6:eq(0)');

});

@if($message = Session::get('success'))

toastr.success("{{ $message}}");

@endif

@if($message = Session::get('updated'))

toastr.warning("{{ $message}}");

@endif

</script>

@endsection

### Membuat Route

1. Tambahkan script berikut pada route/web.php



1. Lalu jalankan perintah php artisan route:list untuk melihat daftar method dari pelanggan.

### Membuat method create()

1. Buat file baru di dalam folder **views/customer** dengan nama **tambah.blade.php**
2. Tambahkan script berikut:

@extends('layouts.template')

@section('judulh1','Admin - Customer')

@section('konten')

<div class="col-md-6">

@if ($errors->any())

<div class="alert alert-danger">

<strong>Whoops!</strong> There were some problems with your input.<br><br>

<ul>

@foreach ($errors->all() as $error)

<li>{{ $error }}</li>

@endforeach

</ul>

</div>

@endif

<div class="card card-success">

<div class="card-header">

<h3 class="card-title">Tambah Data Customer</h3>

</div>

<!-- /.card-header -->

<!-- form start -->

<form action="{{ route('pelanggan.store') }}" method="POST">

@csrf

<div class=" card-body">

<div class="form-group">

<label for="name">Nama Pelanggan</label>

<input type="text" class="form-control" id="name" name="name" placeholder="">

</div>

<div class="form-group">

<label for="email">Email</label>

<input type="email" class="form-control" id="email" name="email">

</div>

<div class="form-group">

<label for="phone">No Telepon</label>

<input type="text" class="form-control" id="phone" name="phone">

</div>

<div class="form-group">

<label for="address">Alamat</label>

<textarea id="address" name="address" class=" form-control" rows="4"></textarea>

</div>

</div>

<!-- /.card-body -->

<div class="card-footer">

<button type="submit" class="btn btn-success float-right">Simpan</button>

</div>

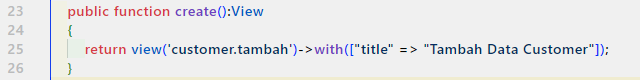
</form>

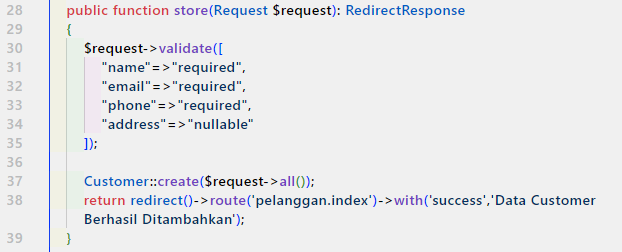
</div>

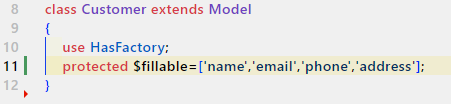
</div>

@endsection

1. Lalu ke file **CustomerController.php** tambahkan method create di dalam **class CustomerController.**

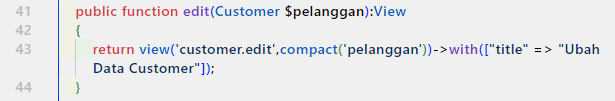


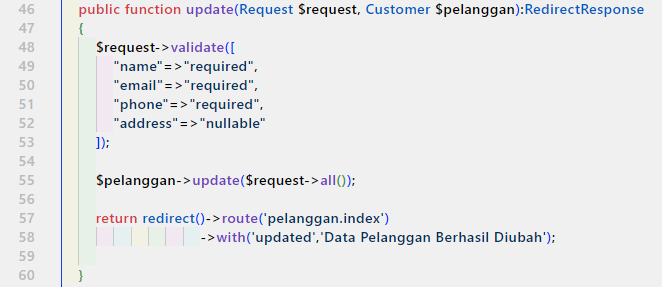
1. Lalu tambahkan method store()
2. Pada model Customer tambahkan variabel $fillable



### Membuat method edit() dan update()

1. Tambahkan script berikut pada file CustomerController.php didalam class CustomerController





1. Buat file baru pada Views/Customer dengan nama **edit.blade.php**, lalu tambahkan script berikut:

@extends('layouts.template')

@section('judulh1','Admin - Customer')

@section('konten')

<div class="col-md-6">

@if ($errors->any())

<div class="alert alert-danger">

<strong>Whoops!</strong> There were some problems with your input.<br><br>

<ul>

@foreach ($errors->all() as $error)

<li>{{ $error }}</li>

@endforeach

</ul>

</div>

@endif

<div class="card card-warning">

<div class="card-header">

<h3 class="card-title">Ubah Data Customer</h3>

</div>

<!-- /.card-header -->

<!-- form start -->

<form action="{{ route('pelanggan.update',$pelanggan->id) }}" method="POST">

@csrf

@method('PUT')

<div class=" card-body">

<div class="form-group">

<label for="name">Nama Pelanggan</label>

<input type="text" class="form-control" id="name" name="name" placeholder="" value="{{ $pelanggan->name }}">

</div>

<div class="form-group">

<label for="email">Email</label>

<input type="email" class="form-control" id="email" name="email" value="{{ $pelanggan->email }}">

</div>

<div class="form-group">

<label for="phone">No Telepon</label>

<input type="text" class="form-control" id="phone" name="phone" value="{{ $pelanggan->phone }}">

</div>

<div class="form-group">

<label for="address">Alamat</label>

<textarea id="address" name="address" class=" form-control"

rows="4">{{ $pelanggan->address }}</textarea>

</div>

</div>

<!-- /.card-body -->

<div class="card-footer">

<button type="submit" class="btn btn-warning float-right">Simpan</button>

</div>

</form>

</div>

</div>

@endsection

### **Membuat method show()**

1. Buat file baru di dalam **views/customer** dengan nama **tampil.blade.php**
2. Kemudian tambahkan script berikut:

@extends('layouts.template')

@section('judulh1','Admin - Customer')

@section('konten')

<div class="col-md-6">

@if ($errors->any())

<div class="alert alert-danger">

<strong>Whoops!</strong> There were some problems with your input.<br><br>

<ul>

@foreach ($errors->all() as $error)

<li>{{ $error }}</li>

@endforeach

</ul>

</div>

@endif

<div class="card card-secondary">

<div class="card-header">

<h3 class="card-title">Data Customer</h3>

</div>

<!-- /.card-header -->

<!-- form start -->

<form method=" POST">

@csrf

@method('PUT')

<div class=" card-body">

<div class="form-group">

<label for="name">Nama Pelanggan</label>

<input type="text" class="form-control" id="name" name="name" placeholder=""

value="{{ $pelanggan->name }}" disabled>

</div>

<div class="form-group">

<label for="email">Email</label>

<input type="email" class="form-control" id="email" name="email" value="{{ $pelanggan->email }}"

disabled>

</div>

<div class="form-group">

<label for="phone">No Telepon</label>

<input type="text" class="form-control" id="phone" name="phone" value="{{ $pelanggan->phone }}"

disabled>

</div>

<div class="form-group">

<label for="address">Alamat</label>

<textarea id="address" name="address" class=" form-control" rows="4"

disabled>{{ $pelanggan->address }}</textarea>

</div>

</div>

<!-- /.card-body -->

<div class="card-footer">

</div>

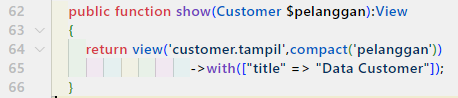
</form>

</div>

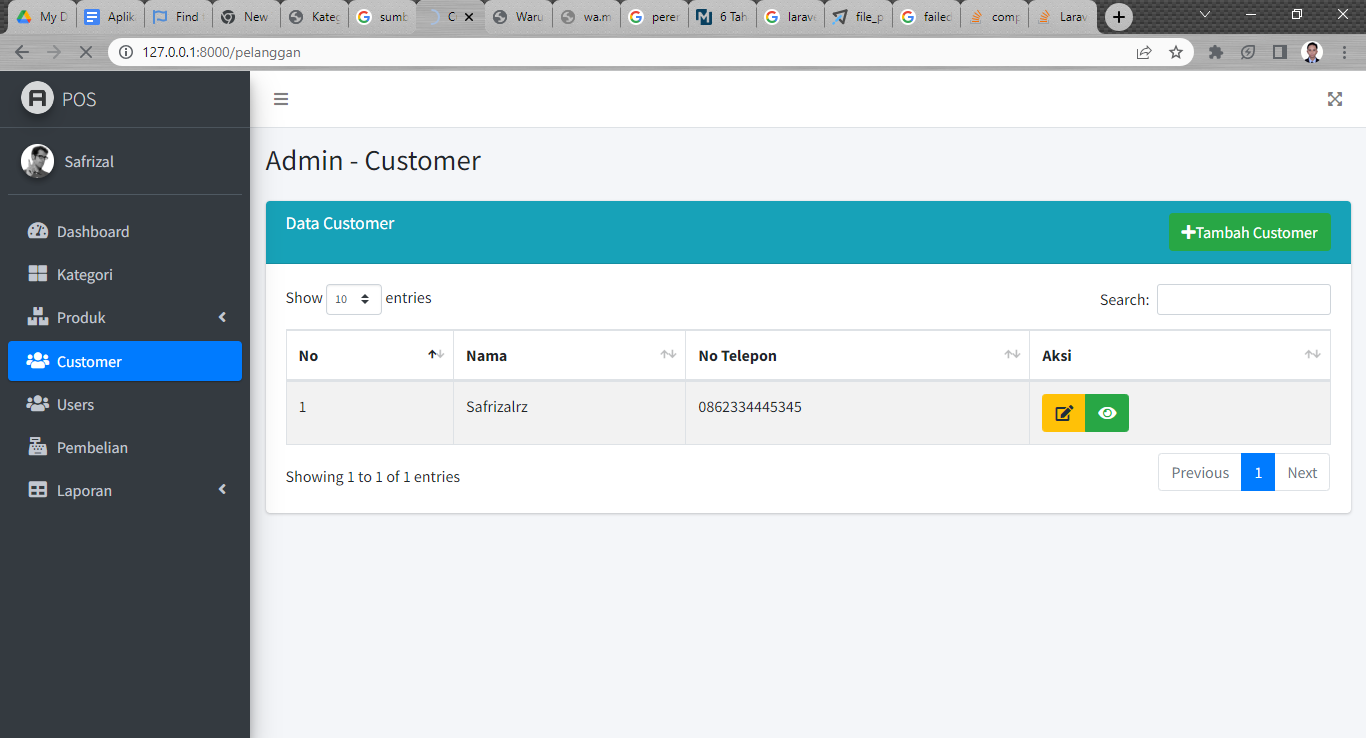
</div>

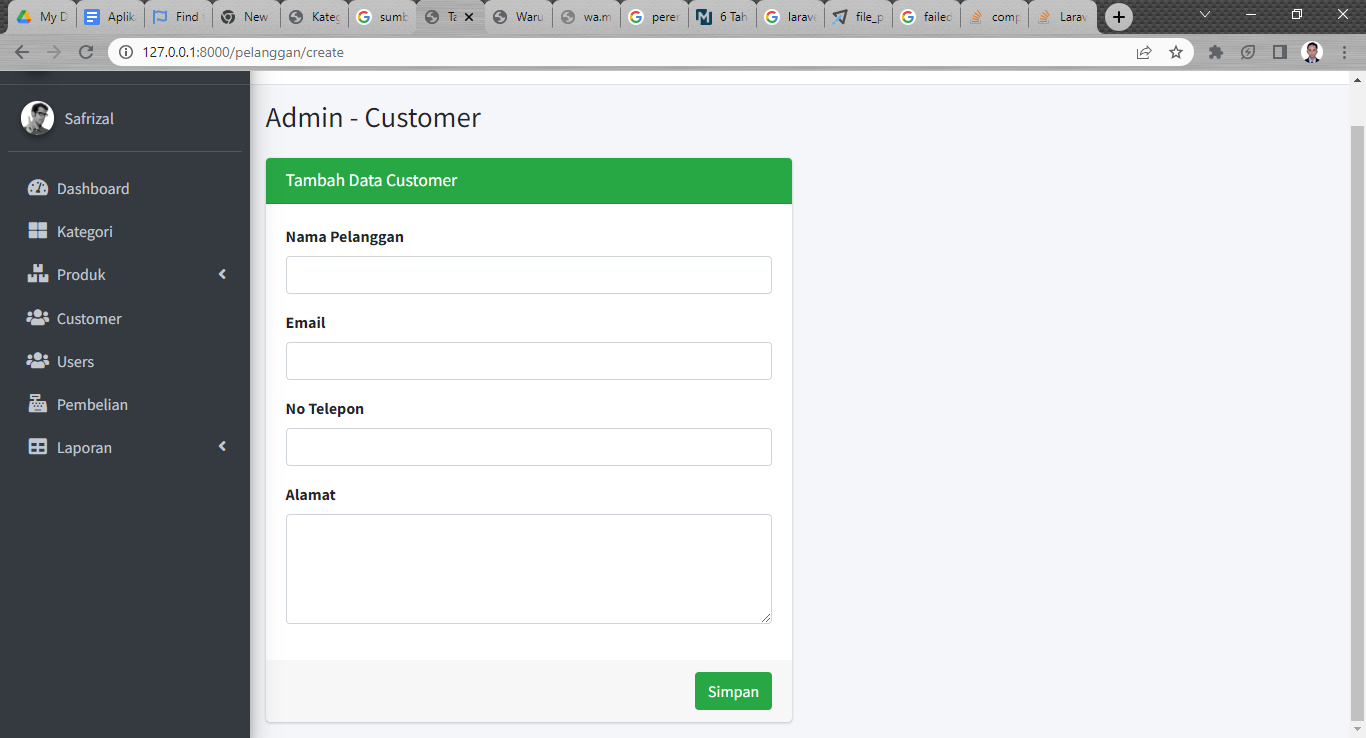
@endsection

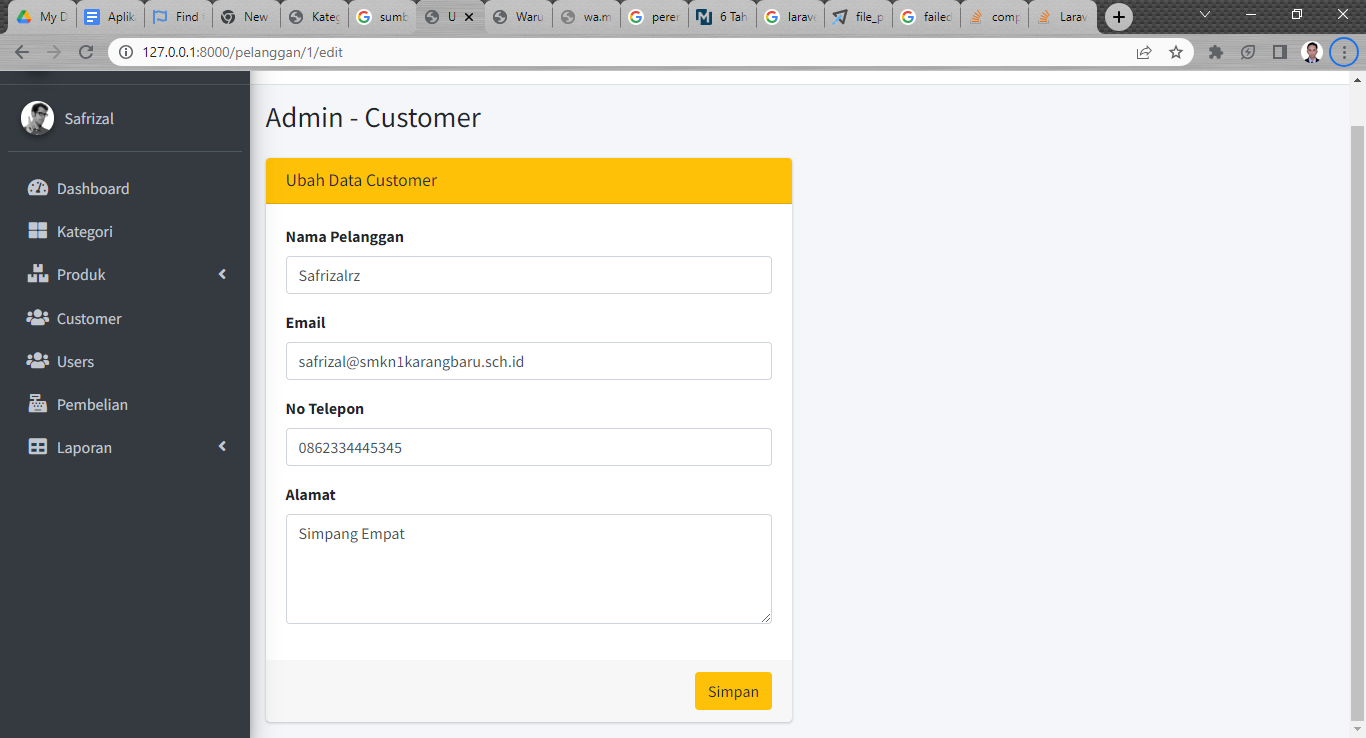
1. Buka CustomerController.php lalu tambahkan method show()

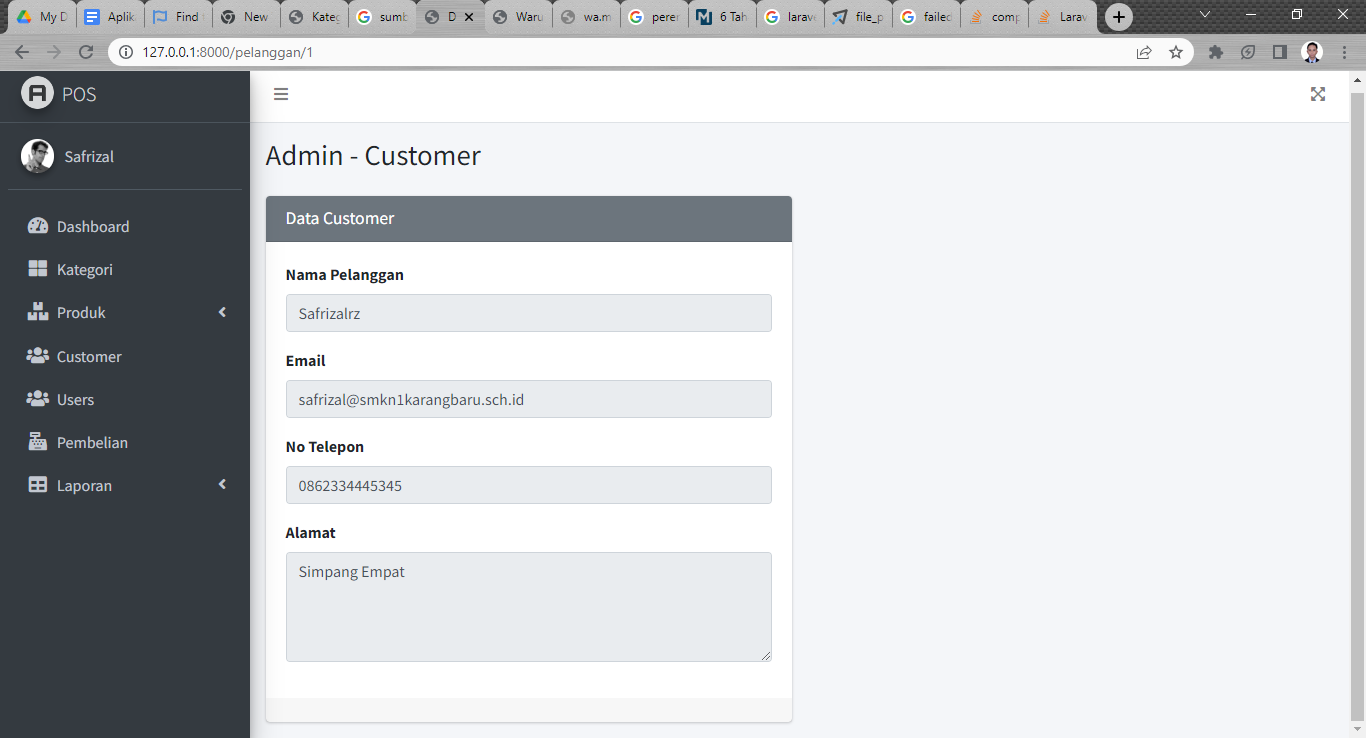


1. Uji coba keseluruhan method yang telah dibuat
2. Hasil: tampilan









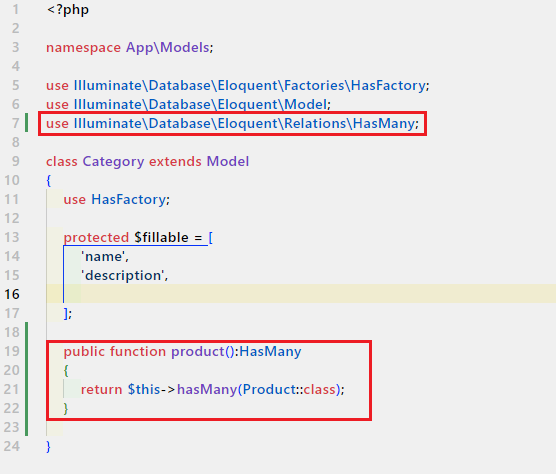
## Manajemen Product

### Membuat ProductController dan Modifikasi Model Product dan Model Category

1. Buatlah product controller dengan cara menjalankan perintah php artisan berikut:

php artisan make:controller ProductController

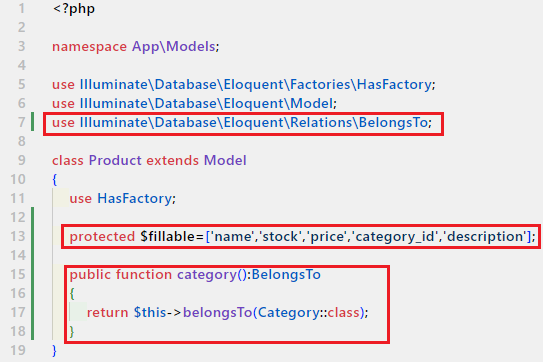
1. Jika sudah berhasil cek kembali **ProductController.php** pada **App\Http\Controller**
2. Sebelum kita memodifikasi **Model Product** dan **Model Category**, kita perlu memahami *relationship* antara **Category** dan **Product**, jika kita maknai bersama **Sebuah Category** dapat dimiliki oleh **Banyak Product**. Contohnya terdapat sebuah Category Baju didalam bisa terdapat berbagai jenis Product Baju contoh Baju Anak Laki-Laki, Baju Anak Perempuan, Baju Remaja, Kemeja, dan lain-lain. Maka dari itu relasi yang dimiliki antara Category dan Product adalah **One To Many**. Relasi ini akan mempengaruhi cara kita memodifikasi Model Product dan Model Category.
3. Buka File **Category.php** yang terdapat pada **app\Models**. Lalu tambahkan script yang ditandai berikut:



Untuk mempermudah pemahaman, kita bisa membaca script di atas seperti berikut:

**Category** **hasMany** (dimiliki oleh Banyak) **Product**

1. Buka file Product.php pada app\Models. Lalu tambahkan script yang ditandai berikut:



Untuk mempermudah pemahaman, kita bisa membaca script di atas seperti berikut:

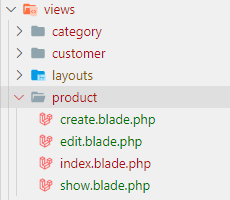
**Product** **belongsTo** (dimiliki oleh) **Category**

### Membuat Route Product dan Method index()

1. Buka file **routes\web.php** lalu tambahkan Route berikut:



1. Jangan lupa untuk meng-import class **ProductController** ke dalam **routes/web.php**.
2. Kemudian buat sebuah folder di dalam folder views dengan nama product, lalu buat 4 file dengan nama **index.blade.php**, **create.blade.php**, **edit.blade.php**, dan **show.blade.php**.



Kalau diperhatikan penamaan file dari ketika membuat category dan customer berbeda-beda, tujuannya adalah untuk membangun pemahaman dalam isi file dan tujuan dari file tersebut dibuat, seperti pada customer terdapat file tabel.blade.php yang isinya adalah tampilan tabel, dan seterusnya. Lalu pada bagian product ini kita mulai menggunakan penamaan standar yang biasanya digunakan dalam project laravel. File index, akan menjadi tampilan utama dari product, kemudian file create adalah tampilan untuk menambahkan produk, file edit untuk merubah/update produk, sedangkan file show untuk menampilkan detail product.

1. Buka file index.blade.php lalu tambahkan script berikut:

@extends('layouts.template')

@section('tambahanCSS')

<!-- DataTables -->

<link rel="stylesheet" href="plugins/datatables-bs4/css/dataTables.bootstrap4.min.css">

<link rel="stylesheet" href="plugins/datatables-responsive/css/responsive.bootstrap4.min.css">

<link rel="stylesheet" href="plugins/datatables-buttons/css/buttons.bootstrap4.min.css">

<!-- Toastr -->

<link rel="stylesheet" type="text/css" href="//cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/css/toastr.css">

@endsection

@section('judulh1','Admin - Product')

@section('konten')

<div class="col-md-12">

<div class="card card-info">

<div class="card-header">

<h2 class="card-title">Data Customer</h2>

<a type="button" class="btn btn-success float-right" href="{{ route('produk.create') }}">

<i class=" fas fa-plus"></i> Tambah Product

</a>

</div>

<!-- /.card-header -->

<div class="card-body">

<table id="example1" class="table table-bordered table-striped ">

<thead>

<tr>

<th>No</th>

<th>Nama</th>

<th>Kategori</th>

<th>Stok</th>

<th>Harga</th>

<th>Aksi</th>

</tr>

</thead>

<tbody>

@foreach($data as $dt)

<tr>

<td>{{ $loop->iteration }}</td>

<td>{{ $dt->name }}</td>

<td>{{ $dt->category->name }}</td>

<td>{{ $dt->stock }}</td>

<td>{{ $dt->price }}</td>

<td>

<div class="btn-group">

<form action="{{ route('produk.destroy',$dt->id)}}" method="POST">

@csrf

@method('DELETE')

<button type="submit" class="btn btn-danger">

<i class=" fas fa-trash"></i>

</button>

</form>

<a type="button" class="btn btn-warning" href="{{ route('produk.edit',$dt->id) }}">

<i class=" fas fa-edit"></i>

</a>

<a type="button" class="btn btn-success" href="{{ route('produk.show',$dt->id) }}">

<i class=" fas fa-eye"></i>

</a>

</div>

</td>

</tr>

@endforeach

</tbody>

</table>

</div>

</div>

</div>

@endsection

@section('tambahanJS')

<!-- DataTables & Plugins -->

<script src="plugins/datatables/jquery.dataTables.min.js"></script>

<script src="plugins/datatables-bs4/js/dataTables.bootstrap4.min.js"></script>

<script src="plugins/datatables-responsive/js/dataTables.responsive.min.js"></script>

<script src="plugins/datatables-responsive/js/responsive.bootstrap4.min.js"></script>

<script src="plugins/datatables-buttons/js/dataTables.buttons.min.js"></script>

<script src="plugins/datatables-buttons/js/buttons.bootstrap4.min.js"></script>

<script src="plugins/jszip/jszip.min.js"></script>

<script src="plugins/pdfmake/pdfmake.min.js"></script>

<script src="plugins/pdfmake/vfs\_fonts.js"></script>

<script src="plugins/datatables-buttons/js/buttons.html5.min.js"></script>

<script src="plugins/datatables-buttons/js/buttons.print.min.js"></script>

<script src="plugins/datatables-buttons/js/buttons.colVis.min.js"></script>

<!-- Toastr -->

<script src="https://cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/toastr.min.js"></script>

@endsection

@section('tambahScript')

<script>

$(function() {

$("#example1").DataTable({

"responsive": true,

"lengthChange": true,

"autoWidth": false,

"responsive": true,

}).buttons().container().appendTo('#example1\_wrapper .col-md-6:eq(0)');

});

@if($message = Session::get('success'))

toastr.success("{{ $message}}");

@elseif($message = Session::get('updated'))

toastr.warning("{{ $message}}");

@elseif($message = Session::get('deleted'))

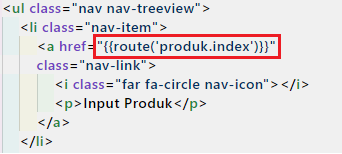
toastr.error("{{ $message}}");

@endif

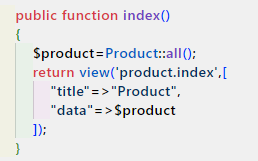
</script>

@endsection

1. Buka file **template.blade.php** pada folder **views\layouts**. Lalu tambahkan script yang ditandai berikut:



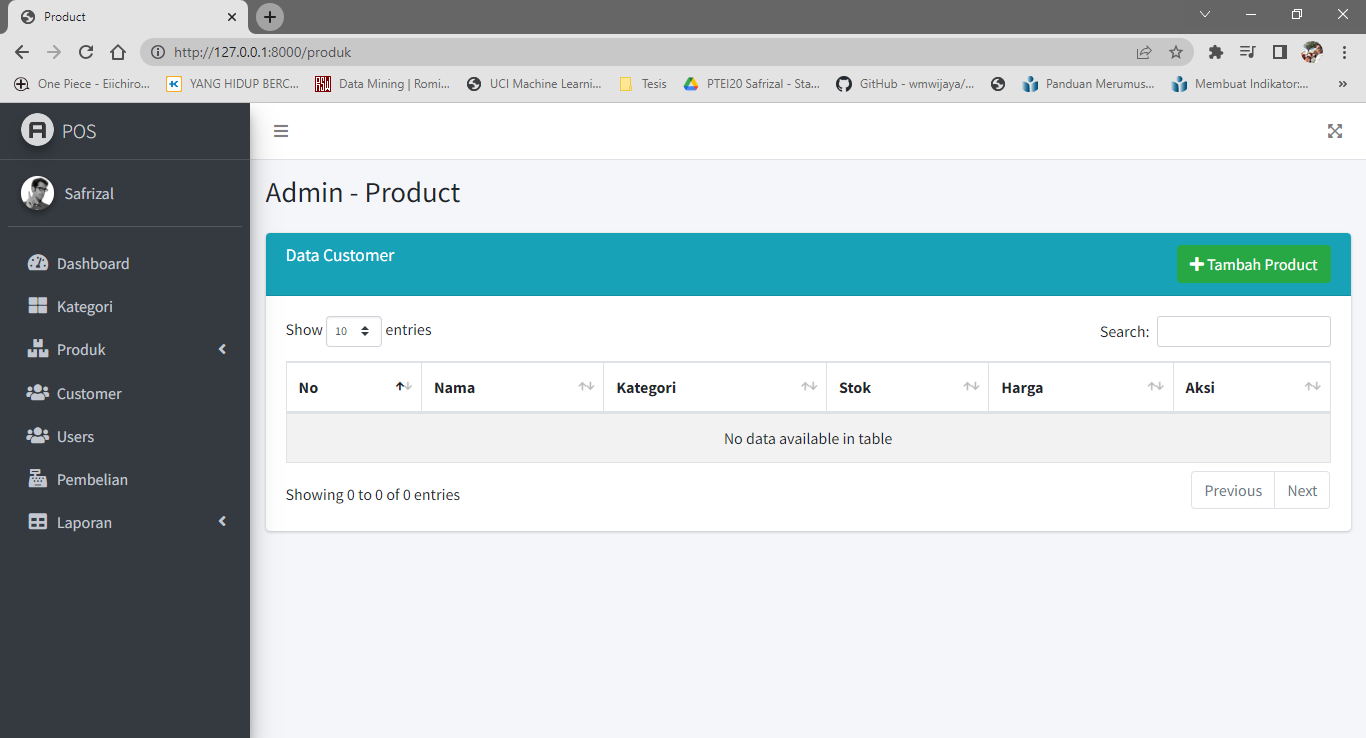
1. Buka file **ProductController.php**, lalu tambahkan Method index() di dalam class ProductController.



1. Jangan lupa untuk import model Product



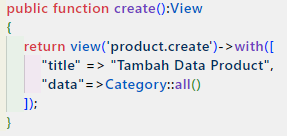
1. Uji coba yang telah dikerjakan.



### Membuat Method create() dan store()

Jika diperhatikan dari manajemen customer sebelumnya, method create() digunakan untuk menampilkan tampilan untuk menambah data sementara method store() digunakan untuk menyimpan data ke dalam database.

1. Tambahkan method create() di dalam class ProductController pada file ProductController.php



1. Jangan lupa untuk import model Category dan class View ke dalam file ProductController.php





1. Buka file create.blade.php pada folder views\product. Tambahkan script berikut:

@extends('layouts.template')

@section('judulh1','Admin - Product')

@section('konten')

<div class="col-md-6">

@if ($errors->any())

<div class="alert alert-danger">

<strong>Whoops!</strong> There were some problems with your input.<br><br>

<ul>

@foreach ($errors->all() as $error)

<li>{{ $error }}</li>

@endforeach

</ul>

</div>

@endif

<div class="card card-success">

<div class="card-header">

<h3 class="card-title">Tambah Data Product</h3>

</div>

<!-- /.card-header -->

<!-- form start -->

<form action="{{ route('produk.store') }}" method="POST">

@csrf

<div class=" card-body">

<div class="form-group">

<label for="name">Nama Produk</label>

<input type="text" class="form-control" id="name" name="name" placeholder="">

</div>

<div class="form-group">

<label for="stock">Stok</label>

<input type="number" class="form-control" id="stock" name="stock">

</div>

<div class="form-group">

<label>Kategori</label>

<select class="form-control" name="category\_id">

@foreach($data as $dt )

<option value="{{ $dt->id }}">{{ $dt->name }}</option>

@endforeach

</select>

</div>

<div class="form-group">

<label for="price">Harga</label>

<input type="number" class="form-control" id="price" name="price">

</div>

<div class="form-group">

<label for="description">Deskripsi</label>

<textarea id="description" name="description" class=" form-control" rows="4"></textarea>

</div>

</div>

<!-- /.card-body -->

<div class="card-footer">

<button type="submit" class="btn btn-success float-right">Simpan</button>

</div>

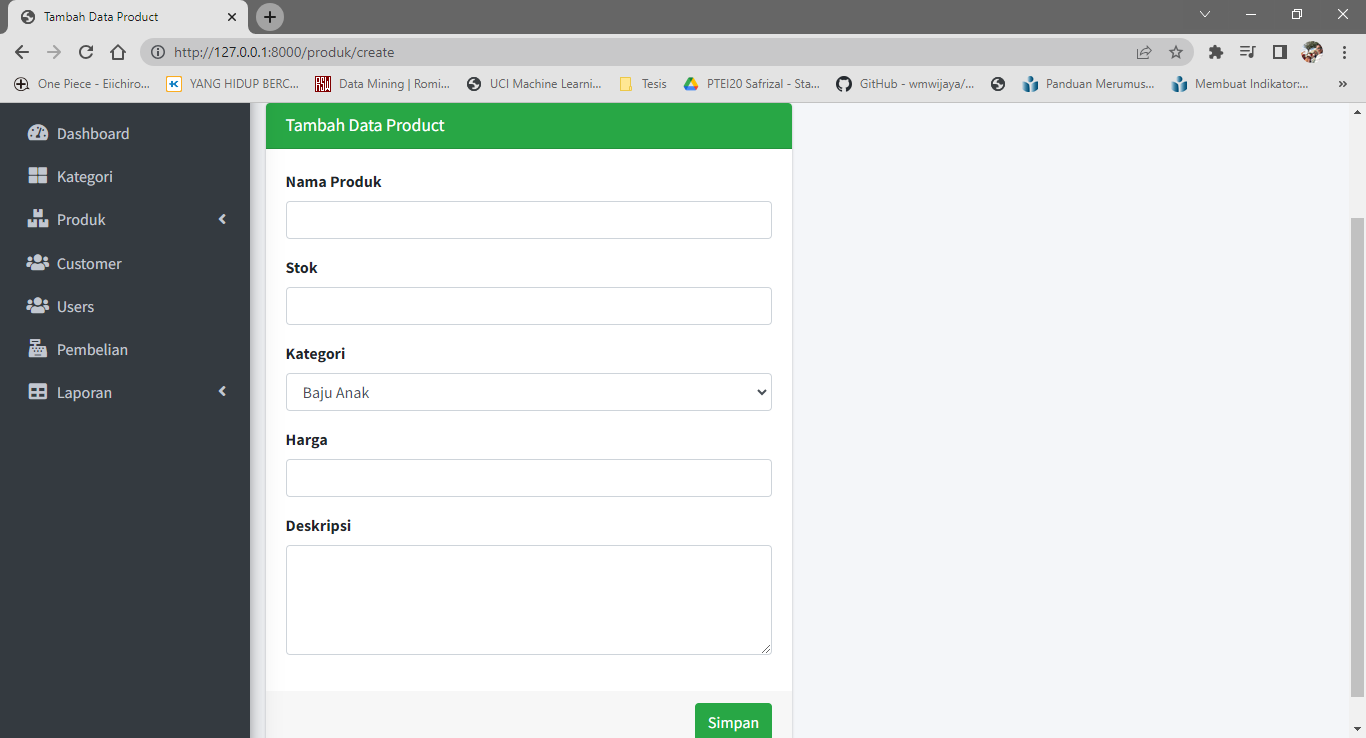
</form>

</div>

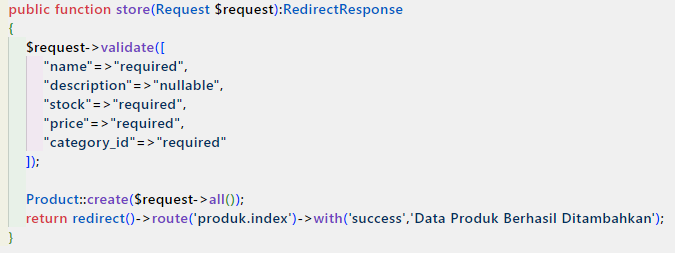
</div>

@endsection

1. Uji coba pekerjaan dengan. Apakah tampil file create.blade.php



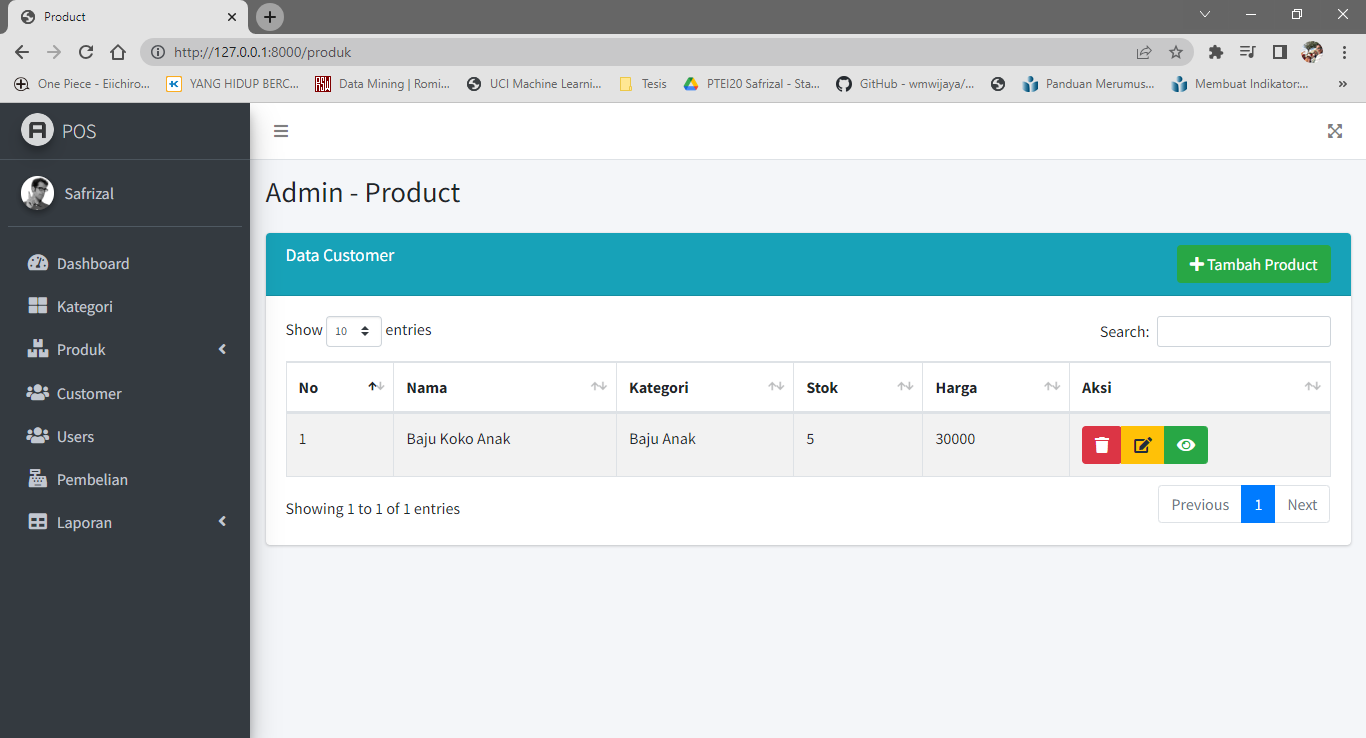
1. Jangan terlalu cepat mengisi data dan mengklik tombol simpan, karena kita belum membuat method store yang digunakan untuk menyimpan data. Buka ProductController.php lalu tambahkan method store().



1. Jangan lupa untuk import class Request dan RedirectResponse



1. Uji coba penambahan data



Apakah sudah berhasil ? Jika sudah perhatikan bagian Kategori, seharusnya berisi nama kategori yang terdapat pada tabel categories (di database). Selanjutnya perhatikan kolom harga, seharusnya yang sudah dikerjakan tidak muncul format mata uang. Lalu bagaimana caranya agar muncul format uangnya. Untuk melakukan hal tersebut kita harus membuat fungsi baru.

### Membuat Format Uang

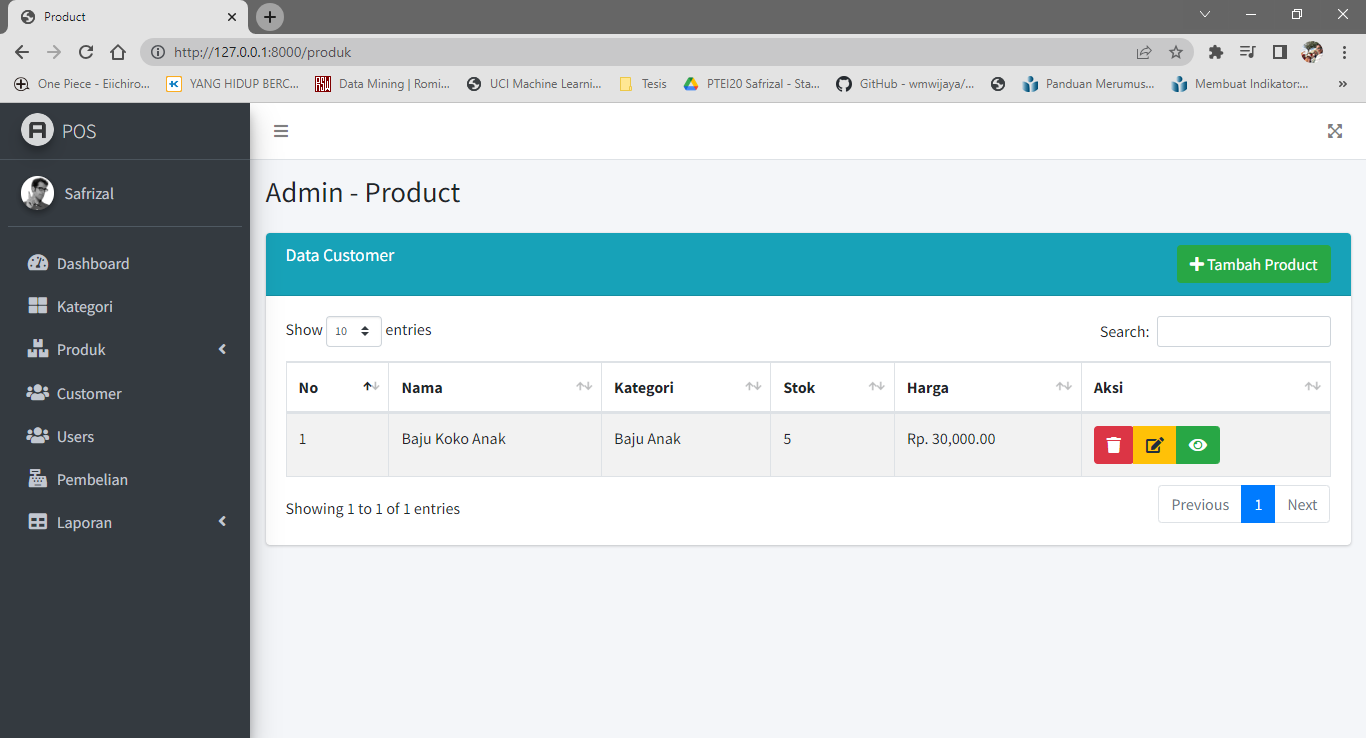
1. Buka file AppServiceProvider.php pada folder app\Providers.
2. Tambahkan script berikut:



1. Rubah file index.blade.php yang berada di dalam folder views\products



1. Buka kembali aplikasi pos di browser, silahkan refresh. Apakah sudah muncul format uang?



### Membuat Method edit() dan Method update()

Method edit digunakan untuk menampilkan tampilan edit product sedangkan method update digunakan untuk merubah record pada tabel products di database.

1. Tambahkan method edit() pada class ProductController.



1. Buka file edit.blade.php lalu tambahkan script berikut:

@extends('layouts.template')

@section('judulh1','Admin - Product')

@section('konten')

<div class="col-md-6">

@if ($errors->any())

<div class="alert alert-danger">

<strong>Whoops!</strong> There were some problems with your input.<br><br>

<ul>

@foreach ($errors->all() as $error)

<li>{{ $error }}</li>

@endforeach

</ul>

</div>

@endif

<div class="card card-warning">

<div class="card-header">

<h3 class="card-title">Ubah Data Product</h3>

</div>

<!-- /.card-header -->

<!-- form start -->

<form action="{{ route('produk.update',$produk->id) }}" method="POST">

@csrf

@method('PUT')

<div class=" card-body">

<div class="form-group">

<label for="name">Nama Produk</label>

<input type="text" class="form-control" id="name" name="name" placeholder=""

value="{{$produk->name}}">

</div>

<div class="form-group">

<label for="stock">Stok</label>

<input type="number" class="form-control" id="stock" name="stock" value="{{$produk->stock}}">

</div>

<div class="form-group">

<label>Kategori</label>

<select class="form-control" name="category\_id">

@foreach($data as $dt )

<option value="{{ $dt->id }}" @if($dt->id===$produk->category\_id) selected

@endif>{{ $dt->name }}

</option>

@endforeach

</select>

</div>

<div class="form-group">

<label for="price">Harga</label>

<input type="number" class="form-control" id="price" name="price" value="{{$produk->price}}">

</div>

<div class=" form-group">

<label for="description">Deskripsi</label>

<textarea id="description" name="description" class=" form-control"

rows="4">{{ $produk->description }}</textarea>

</div>

</div>

<!-- /.card-body -->

<div class="card-footer">

<button type="submit" class="btn btn-warning float-right">Ubah</button>

</div>

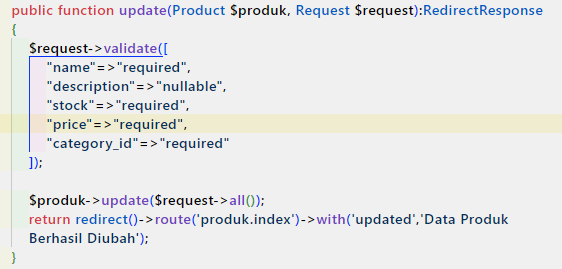
</form>

</div>

</div>

@endsection

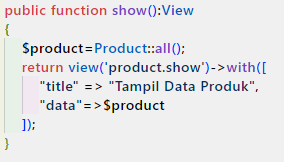
1. Buka kembali ProductController lalu tambahkan script berikut pada class ProductController.



Selanjutnya uji coba proses update, apakah berjalan lancar atau tidak.

### Membuat Method show()

1. Tambahkan script berikut pada class ProductController.



1. Buka file show.blade.php pada folder views\product, lalu tambahkan script berikut:

@extends('layouts.template')

@section('judulh1','Admin - Product')

@section('konten')

<div class="col-md-6">

@if ($errors->any())

<div class="alert alert-danger">

<strong>Whoops!</strong> There were some problems with your input.<br><br>

<ul>

@foreach ($errors->all() as $error)

<li>{{ $error }}</li>

@endforeach

</ul>

</div>

@endif

<div class="card card-primary">

<div class="card-header">

<h3 class="card-title">Data Product</h3>

</div>

<!-- /.card-header -->

<div class=" card-body">

<table>

<tr>

<th>Nama Produk</th>

<td>:</td>

<td>{{ $data[0]->name }}</td>

</tr>

<tr>

<th>Kategori Produk</th>

<td>:</td>

<td>{{ $data[0]->category->name }}</td>

</tr>

<tr>

<th>Stok</th>

<td>:</td>

<td>{{ $data[0]->stock }}</td>

</tr>

<tr>

<th>Harga</th>

<td>:</td>

<td>@money($data[0]->price)</td>

</tr>

<tr>

<th>Deskripsi</th>

<td>:</td>

<td>{{ $data[0]->description}}</td>

</tr>

</table>

</div>

<!-- /.card-body -->

</div>

</div>

@endsection

1. Uji coba method show.

### Membuat Method destroy()

1. Tambahkan script berikut pada class ProductController:



1. Selanjutnya uji coba seluruh method yang telah dibuat.

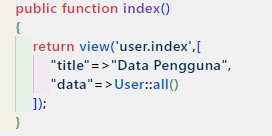
## Manajemen User

### Membuat controller UserController

1. Untuk membuat controller user kita cukup mengetikkan perintah artisan

php artisan make:controller UserController

1. Tambahkan Method index() pada UserController



1. Jangan lupa untuk mengimport class model User pada UserController



### Membuat Route

Route yang akan kita buat untuk sementara ada 2, yaitu index dan store, 5 lainnya sementara tidak kita perlukan.

1. Tambahkan script berikut pada routes/web.php



1. Lalu cek daftar route menggunakan perintah artisan

php artisan route:list

### Membuat Tampilan Manajemen User

1. Buat folder baru di dalam views dengan nama user
2. Buat file baru di dalam folder user dengan nama index.blade.php, lalu tambahkan script berikut pada file index.blade.php

@extends('layouts.template')

@section('tambahanCSS')

<!-- DataTables -->

<link rel="stylesheet" href="plugins/datatables-bs4/css/dataTables.bootstrap4.min.css">

<link rel="stylesheet" href="plugins/datatables-responsive/css/responsive.bootstrap4.min.css">

<link rel="stylesheet" href="plugins/datatables-buttons/css/buttons.bootstrap4.min.css">

<!-- Toastr -->

<link rel="stylesheet" type="text/css" href="//cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/css/toastr.css">

@endsection

@section('judulh1','Admin - Users')

@section('judulh3','Users')

@section('konten')

<div class="col-md-4">

<div class="card card-success">

<div class="card-header">

<h3 class="card-title">Input User</h3>

</div>

<!-- /.card-header -->

<!-- form start -->

<form action="{{ route('pengguna.store') }}" method="POST">

@csrf

<div class=" card-body">

<div class="form-group">

<label for="name">Nama</label>

<input type="text" class="form-control" id="name" name="name" placeholder=" Nama Lengkap" required value="{{ old('name') }}">

</div>

<div class="form-group">

<label for="email">Email</label>

<input type="email" class="form-control @error('email') is-invalid @enderror" id="email" name="email" placeholder="JohnDoe@example.com" required value="{{ old('email') }}">

@error('email')

<div class="invalid-feedback">

{{ $message }}

</div>

@enderror

</div>

<div class="form-group">

<label for="password">Password</label>

<input type="password" class="form-control" id="password" name="password" required>

</div>

</div>

<!-- /.card-body -->

<div class="card-footer">

<button type="submit" class="btn btn-success float-right">Simpan</button>

</div>

</form>

</div>

</div>

<div class="col-md-8">

<div class="card card-info">

<div class="card-header">

<h3 class="card-title">Data Category</h3>

</div>

<!-- /.card-header -->

<div class="card-body">

<table id="example1" class="table table-bordered table-striped ">

<thead>

<tr>

<th>No</th>

<th>Nama </th>

<th>Email</th>

</tr>

</thead>

<tbody>

@foreach($data as $dt)

<tr>

<td>{{ $loop->iteration }}</td>

<td>{{ $dt->name }}</td>

<td>{{ $dt->email }}</td>

</tr>

@endforeach

</tbody>

</table>

</div>

</div>

</div>

@endsection

@section('tambahanJS')

<!-- DataTables & Plugins -->

<script src="plugins/datatables/jquery.dataTables.min.js"></script>

<script src="plugins/datatables-bs4/js/dataTables.bootstrap4.min.js"></script>

<script src="plugins/datatables-responsive/js/dataTables.responsive.min.js"></script>

<script src="plugins/datatables-responsive/js/responsive.bootstrap4.min.js"></script>

<script src="plugins/datatables-buttons/js/dataTables.buttons.min.js"></script>

<script src="plugins/datatables-buttons/js/buttons.bootstrap4.min.js"></script>

<script src="plugins/jszip/jszip.min.js"></script>

<script src="plugins/pdfmake/pdfmake.min.js"></script>

<script src="plugins/pdfmake/vfs\_fonts.js"></script>

<script src="plugins/datatables-buttons/js/buttons.html5.min.js"></script>

<script src="plugins/datatables-buttons/js/buttons.print.min.js"></script>

<script src="plugins/datatables-buttons/js/buttons.colVis.min.js"></script>

<!-- Toastr -->

<script src="https://cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/toastr.min.js"></script>

@endsection

@section('tambahScript')

<script>

$(function() {

$("#example1").DataTable({

"responsive": true,

"lengthChange": true,

"autoWidth": false,

"responsive": true,

}).buttons().container().appendTo('#example1\_wrapper .col-md-6:eq(0)');

});

@if($message = Session::get('success'))

toastr.success("{{ $message}}");

@elseif($message = Session::get('updated'))

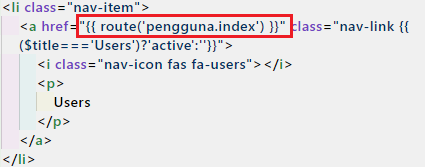
toastr.warning("{{ $message}}");

@endif

</script>

@endsection

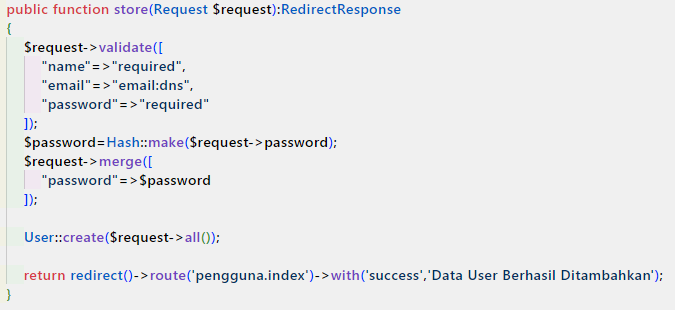
1. Perbaiki **template.blade.php**. Dengan menambahkan route pada sidebar user.



1. Pada manajemen user kita tidak menambahkan menu edit, karena nantinya kita akan membuat menu edit data user tidak di admin, namun di private menu masing-masing pengguna. Sedangkan untuk menu hapus tidak kita buat juga karena User berkaitan dengan tabel order, sehingga jika data user dapat dihapus maka akan terjadi kecacatan data nantinya.

### Membuat method store()

1. Untuk menyimpan data ke database, tambahkan method store pada UserController

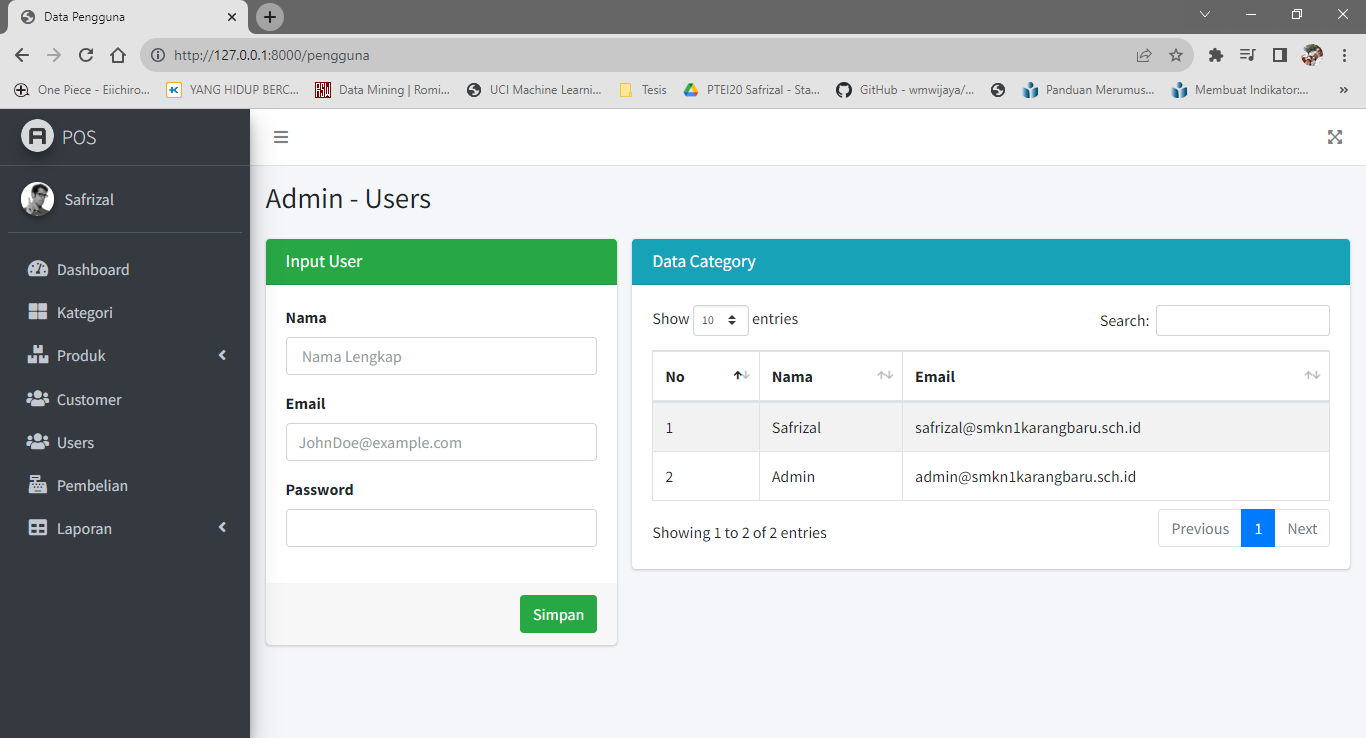


1. Import class Hash dan Redirect Response pada file UserController.php

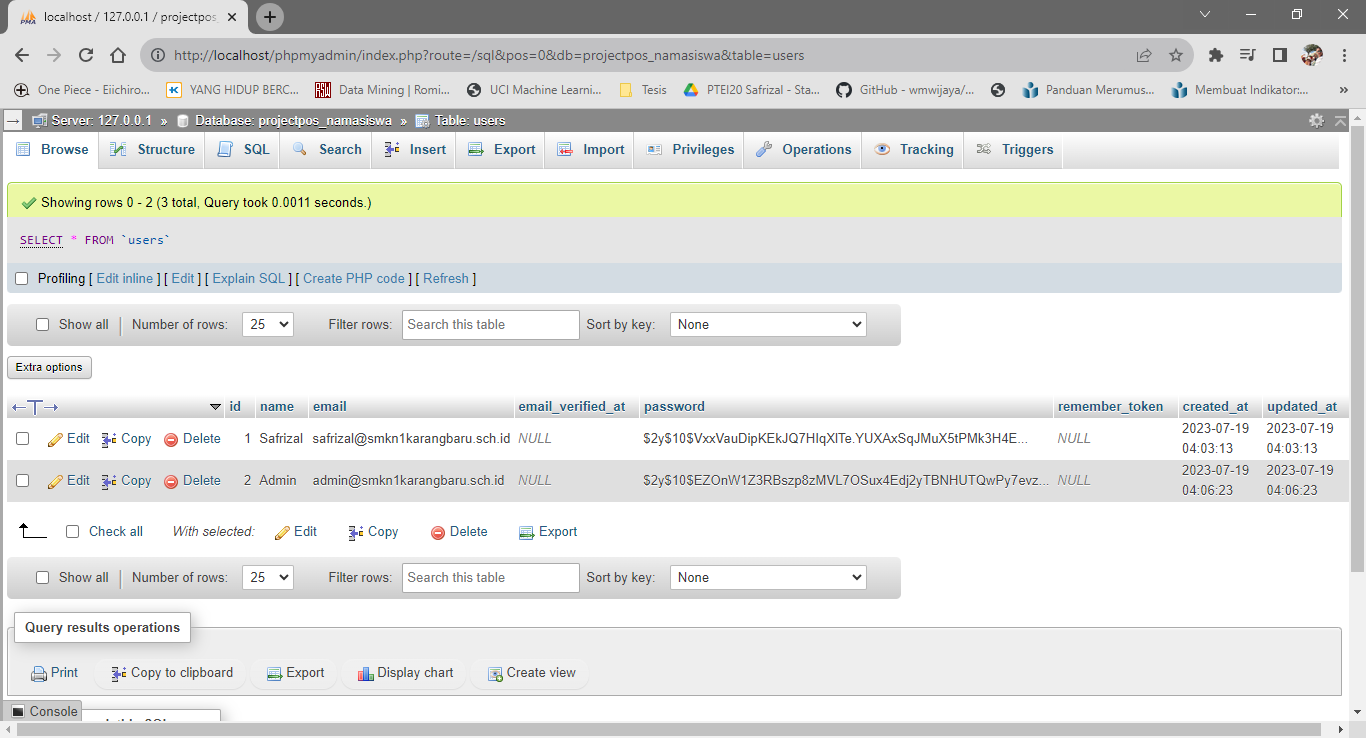




1. Lalu uji coba menyimpan data user



1. Buka localhost/phpmyadmin, lalu buka tabel user, perhatikan bagian password!



1. Mengapa passwordnya bisa menjadi seperti itu ?

**JIKA SUDAH SELESAI HINGGA TAHAP INI.**

1. **Buka file berikut:** [**Essay Praktikum**](https://docs.google.com/document/d/1WrlW1hvJugUjv2h6p3u6VZC8S5oPP6g7qsbQPQSdXRA/edit?usp=sharing)
2. **Download file tersebut**
3. **Lalu jawab pertanyaan yang terdapat didalamnya.**
4. **Kumpulkan softcopy file tersebut hari ini 29 Juli 2023** [**di sini**](https://forms.gle/bmUM68PrtH7pEtms9)**.**

## Autentikasi User

### Menyiapkan Form Login

1. Buat file baru di dalam folder Views dengan nama **login.blade.php**.
2. Lalu tambahkan script berikut:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Login Pengguna</title>

<!-- Google Font: Source Sans Pro -->

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

<!-- Font Awesome -->

<link rel="stylesheet" href="{{ asset('plugins/fontawesome-free/css/all.min.css') }}">

<!-- icheck bootstrap -->

<link rel="stylesheet" href="{{ asset('plugins/icheck-bootstrap/icheck-bootstrap.min.css') }}">

<!-- Theme style -->

<link rel="stylesheet" href="{{ asset('dist/css/adminlte.min.css') }}">

</head>

<body class="hold-transition login-page">

<div class="login-box">

<!-- /.login-logo -->

<div class="card card-outline card-primary">

<div class="card-header text-center">

<h1 class="h1">Aplikasi<b>POS</b></h1>

</div>

<div class="card-body">

<p class="login-box-msg">Please Login</p>

@if(session()->has('loginError'))

<div class="alert alert-danger alert-dismissible">

<button type="button" class="close" data-dismiss="alert" aria-hidden="true">×</button>

<h5><i class="icon fas fa-ban"></i> Alert!</h5>

{{ session('loginError') }}

</div>

@endif

<form action="/login" method="post">

@csrf

<div class="input-group mb-3">

<input type="email" class="form-control @error('email') is-invalid @enderror" placeholder="Email" id="email" name="email" value="{{ old('email') }}">

<div class="input-group-append">

<div class="input-group-text">

<span class="fas fa-envelope"></span>

</div>

</div>

@error('email')

<div class="invalid-feedback">

{{ $message }}

</div>

@enderror

</div>

<div class="input-group mb-3">

<input type="password" class="form-control" placeholder="Password" id="password" name="password">

<div class="input-group-append">

<div class="input-group-text">

<span class="fas fa-lock"></span>

</div>

</div>

</div>

</div>

<div class="card-footer">

<button type="reset" class="btn btn-warning">Cancel</button>

<button type="submit" class="btn btn-primary float-right">Login</button>

</div>

<!-- /.card-body -->

</div>

</form>

<!-- /.card -->

</div>

<!-- /.login-box -->

<!-- jQuery -->

<script src="{{asset('plugins/jquery/jquery.min.js')}}"></script>

<!-- Bootstrap 4 -->

<script src="{{asset('plugins/bootstrap/js/bootstrap.bundle.min.js')}}"></script>

<!-- AdminLTE App -->

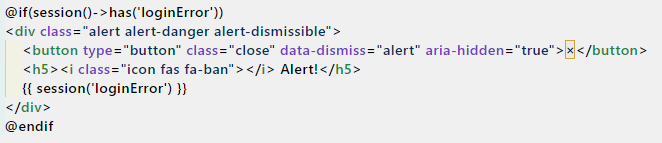
<script src="{{asset('dist/js/adminlte.min.js')}}"></script>

</body>

</html>

Penjelasan:

* Blok kode program berikut berfungsi untuk menampilkan pesan error ketika Login tidak dapat dilakukan:



* Blok kode program email



@error('email') is-invalid @enderror → berguna memberikan class is-invalid (merubah style input) jika terjadi kesalahan pada saat menginput email.

@error('email')

<div class="invalid-feedback">

{{ $message }}

</div>

@enderror

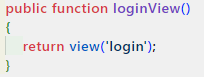
Berguna untuk menampilkan div invalid-feedback dengan pesan error yang terjadi ketika salah menginput email.

### Menambahkan Controller LoginController

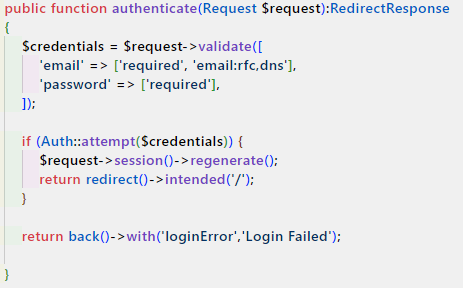
1. Buat controller baru dengan nama LoginController

php artisan make:controller LoginController

1. Tambahkan method loginView pada class LoginController



1. Tambahkan Method authenticate pada class LoginController



1. Jangan lupa untuk mengimport 3 class berikut:



### Menambahkan Route Loginu

1. Tambahkan 2 buah route berikut pada **routes/web.php**



Route tersebut berguna untuk mengarahkan ke method loginView pada controller LoginController ketika mengakses login dengan fungsi get.

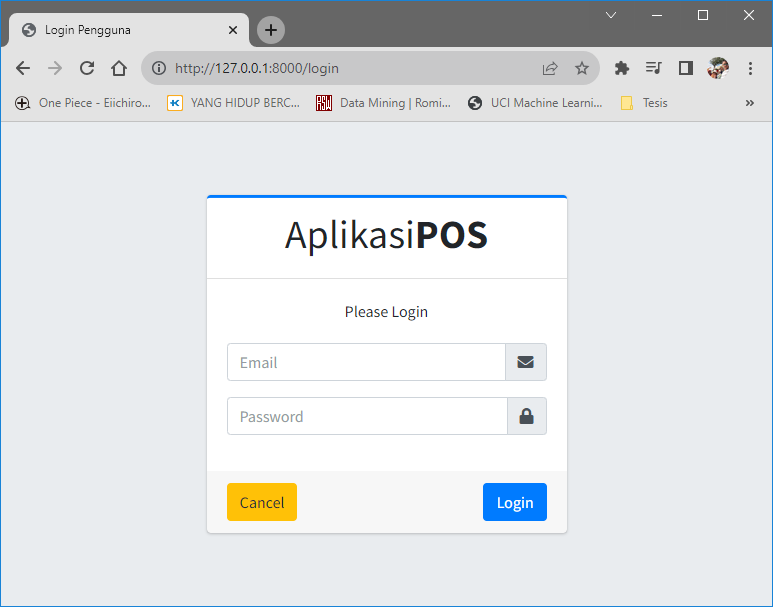
Berikutnya,



Route tersebut berguna untuk mengarahkan ke method authenticate pada controller LoginController ketika mengakses login dengan fungsi post.

**POST dan GET keduanya adalah method atau fungsi yang digunakan untuk mengirimkan data ke halaman lain.**

1. Silahkan ketikkan 127.0.0.1:8000/login maka seharusnya akan tampil form login seperti dibawah berikut:



1. Masukkan email dan password yang sudah di daftarkan pada Manajemen User sebelumnya. Jika berhasilnya seharusnya akan mengarah ke Route ( / ) yang jika mengikuti Buku ini akan mengarah pada tampilan [view (dahsboard)](#1g78ehf8jgam).

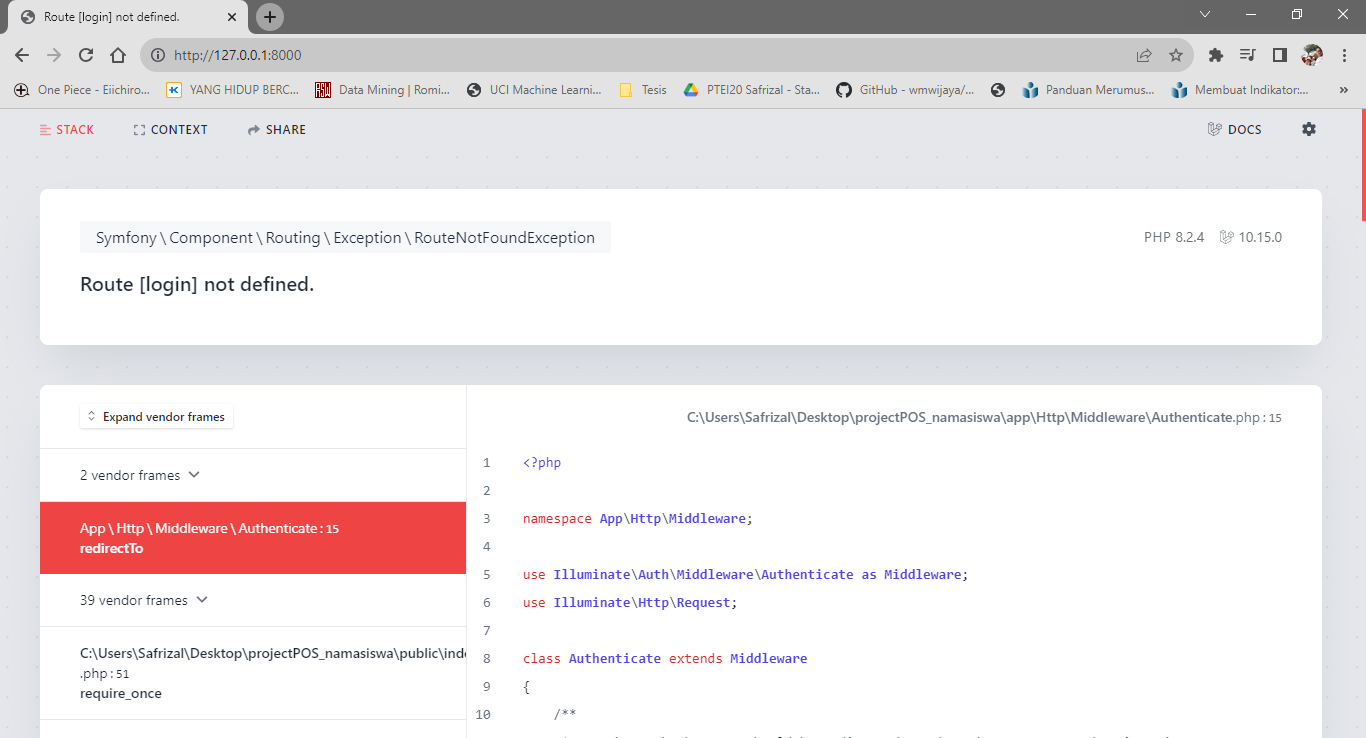
### Memperbaiki Route

Saat ini jika user yang belum login mencoba mengakses route langsung dari browser address tetap akan bisa diakses, hal ini terjadi karena kita belum melindungi akses ke route kita yang lain menggunakan middleware. Middleware seperti namanya yaitu perangkat lunak yang berfungsi sebagai penengah. Dalam konteks laravel khususnya bagian Autentikasi, sebagai penengah antara request yang dilakukan oleh user dan Route yang tersedia. Jadi seandainya ada User atau pengguna ilegal yang mencoba mengakses Route tertentu, maka aksesnya akan diblokir oleh middleware. Cara menggunakan middleware cukup gampang, secara default Laravel sudah menyediakan middleware autentikasi.

1. Buka kembali routes/web.php
2. Kemudian tambahkan ->middleware('auth'); pada setiap Route yang telah kita buat, kecuali Route yang menangani fungsi login. Berikut perubahan keseluruhan **routes/web/php**:



1. Saat ini ketika kita mengakses controller tertentu akan muncul pesan error berikut.



1. Hal ini terjadi karena kita belum memberikan nama pada route login kita. Jadi perbaiki kembali routes/web.php. Perbaiki route login yang menuju ke tampilan login menjadi seperti berikut:



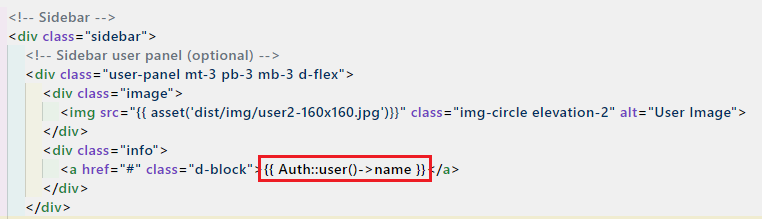
1. Sekarang seharusnya ketika kita coba mengakses routes tertentu yang kita batasi dengan middleware tanpa login maka akan diarahkan ke halaman login.

### 

### Menambahkan Method Logout

Saat ini kita sudah memiliki form login, dan fungsi login yang berjalan. Yang belum kita miliki adalah fungsi logout. Sebelum kita membuat fungsi logout, kita perbaiki dulu tampilan Aplikasi POS yang sedang kita kerjakan. Perbaikan ini bertujuan untuk menambahkan keterangan User yang sedang login dan menambahkan tombol logout.

1. Buka template.blade.php
2. Cari bagian Sidebar, lalu perbaiki script sesuai dengan lokasi yang ditandai pada gambar di bawah ini.



**Penjelasan:** {{ Auth::user()->name }} berguna untuk menampilkan nama user yang sedang login

1. Kemudian cari bagian Right navbar links, tambahkan script <li> berikut di dalam tag <ul class=”navbar-nav ml-auto> </ul>

<li class="nav-item dropdown">

<a class="nav-link" data-toggle="dropdown" href="#">

<i class="far fa-user mr-2"></i>{{ Auth::user()->name }}

<span class="badge badge-warning navbar-badge"></span>

</a>

<div class="dropdown-menu dropdown-menu-lg dropdown-menu-right">

<span class="dropdown-item dropdown-header">User Menu</span>

<div class="dropdown-divider"></div>

<a href="#" class="dropdown-item">

<i class="fas fa-user mr-2"></i> {{ Auth::user()->name }}

<span class="float-right text-muted text-sm"></span>

</a>

<div class="dropdown-divider"></div>

<form action="logout" method="POST">

@csrf

<button type="submit" class="dropdown-item">

<i class="fas fa-sign-out-alt mr-2"></i>Logout

</button>

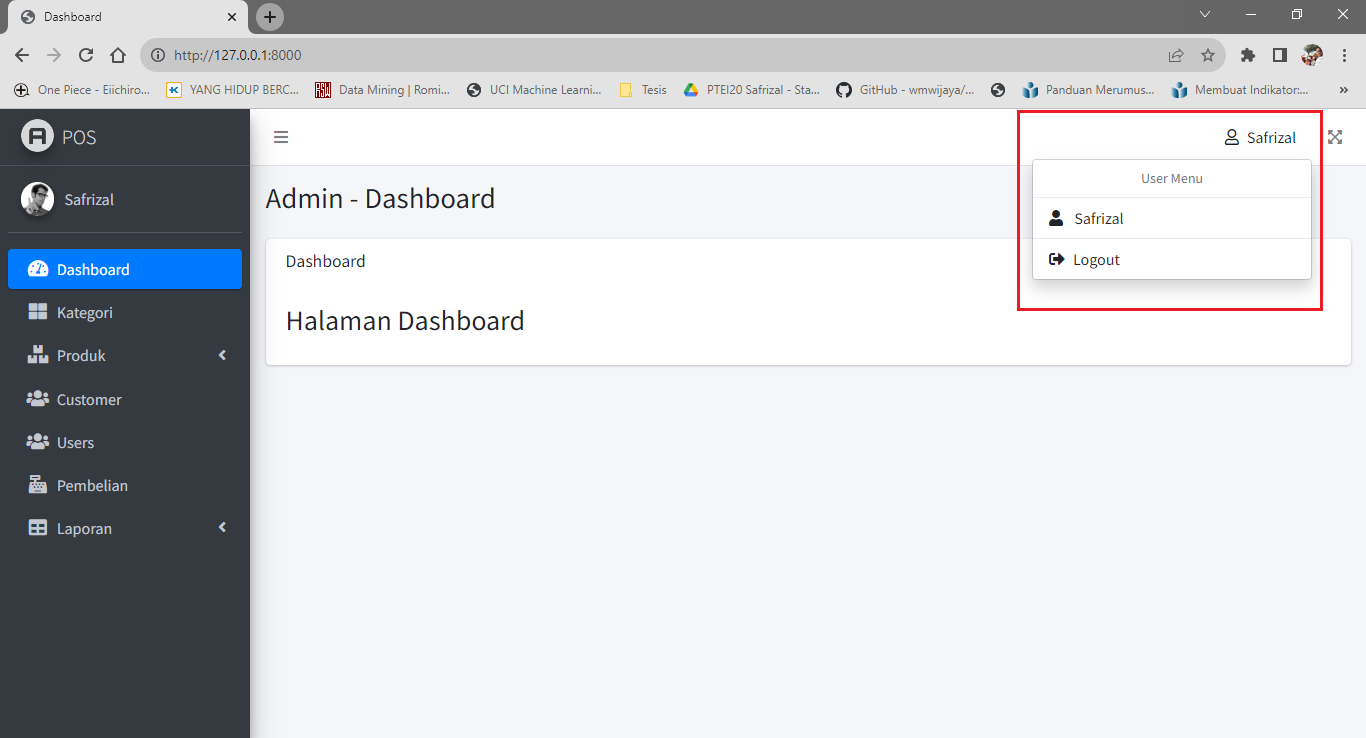
</form>

</div>

</li>

**Penjelasan:** jika diperhatikan kembali tombol logout diletakkan didalam sebuah form dengan method POST, tujuannya adalah untuk mencegah orang lain mengeluarkan kita dari aplikasi (logout).

1. Jika penempatan script benar maka pada halaman utama akan muncul menu baru seperti berikut:



1. Lalu buka kembali controller LoginController lalu tambahkan method logout di dalam class LoginController.



1. Silahkan uji coba tombol logout, lalu coba akses Routes tanpa melakukan login terlebih dahulu.

## Penjualan

Untuk bagian penjualan, bagian ini agak spesial karena kita harus membuat halaman penjualan berubah tanpa me-refresh halaman, untuk melakukan hal tersebut kita memiliki beberapa opsi (pilihan), yang pertama kita dapat menggunakan bantuan javascript dan ajax, yang kedua kita dapat menggunakan framework javascript seperti Vue JS, pilihan ketiga kita dapat menggunakan Livewire. nah , kita akan gunakan opsi ketiga, karena Livewire adalah framework fullstack untuk laravel yang dapat digunakan untuk membuat antar muka yang dinamis dengan cara yang sederhana tanpa meninggalkan kenyamanan dari laravel. Namun di modul ini saya tidak akan menjelaskan dasar-dasar livewire, kalian harus mempelajarinya sendiri di youtube, berikut video yang saya gunakan untuk belajar, kalo tidak cocok dengan kalian, kalian bisa mencari sumber lain.

[0 Belajar Dasar Livewire - Apa itu Laravel Livewire?](https://www.youtube.com/watch?v=ia3QN9ud-yI&list=PL-X81XM3cE187IIB2_RM6eNjnsYkwG8tW&index=1)

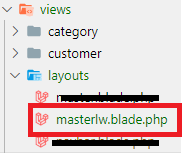
### Membuat Komponen

1. Buat 2 buah komponen baru dengan nama penjualan dan orders

php artisan livewire:make penjualan

php artisan livewire:make orders

1. Buat sebuah file baru didalam layouts dan beri nama **masterlw.blade.php**



1. Pada file **masterlw.blade.php**, tambahkan script berikut

<!doctype html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Bootstrap demo</title>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-rbsA2VBKQhggwzxH7pPCaAqO46MgnOM80zW1RWuH61DGLwZJEdK2Kadq2F9CUG65" crossorigin="anonymous">

<!-- Font Awesome Icons -->

<link rel="stylesheet" href="{{ asset('plugins/fontawesome-free/css/all.min.css') }}">

@livewireStyles

</head>

<body>

<nav class="navbar navbar-expand-lg navbar-dark bg-dark border-bottom border-body" data-bs-theme="dark">

<div class="container-fluid">

<a class="navbar-brand" href="#">Navbar</a>

<button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNavAltMarkup" aria-controls="navbarNavAltMarkup" aria-expanded="false" aria-label="Toggle navigation">

<span class="navbar-toggler-icon"></span>

</button>

<div class="collapse navbar-collapse" id="navbarNavAltMarkup">

<div class="navbar-nav ms-auto">

<a class="nav-link active" aria-current="page" href="#">{{ Auth::user()->name }}</a>

<form action="logout" method="POST" class="d-flex">

@csrf

<button type="submit" class="btn btn-outline-secondary"><i class="fas fa-sign-out-alt mr-2"></i>Logout</button>

</form>

</div>

</div>

</div>

</nav>

<div class="container-fluid py-3">

@yield('konten')

</div>

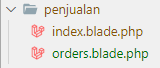
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-kenU1KFdBIe4zVF0s0G1M5b4hcpxyD9F7jL+jjXkk+Q2h455rYXK/7HAuoJl+0I4" crossorigin="anonymous"></script>

@livewireScripts

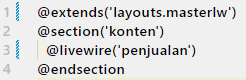
</body>

</html>

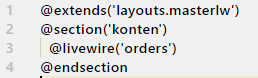
1. Buat 1 buah folder baru didalam folder views dengan nama folder penjualan
2. Di dalam folder penjualan buat 2 buah file baru dengan nama **index.blade.php** dan **orders.blade.php**



1. Di dalam file **index.blade.php** tambahkan script berikut:



1. Di dalam file **orders.blade.php** tambahkan script berikut:



1. Sekarang pindah ke folder **views/livewire**, buka file **penjualan.blade.php** tambahkan script berikut.

<div class="row">

<div class="d-flex aligns-items-center justify-content-center">

<div class="col-6">

<div class="card ">

<div class="card-header">

<h5 class="card-title">Invoice</h5>

</div>

<!-- /.card-header -->

<!-- form start -->

@if ($errors->any())

<div class="alert alert-danger">

<strong>Whoops!</strong> There were some problems with your input.<br><br>

<ul>

@foreach ($errors->all() as $error)

<li>{{ $error }}</li>

@endforeach

</ul>

</div>

@endif

<form wire:submit.prevent="store" method="POST">

@csrf

<div class=" card-body">

<div class="form-group">

<label for="name">Customer</label>

<select class="form-control" wire:model="customer\_id">

<option hidden>--Pilih Customer--</option>

@foreach($data as $dt )

<option value="{{ $dt->id }}">{{ $dt->name }}</option>

@endforeach

</select>

</div>

</div>

<!-- /.card-body -->

<div class="card-footer text-end">

<button type="submit" class="btn btn-success btn-sm">Submit</button>

</div>

</form>

</div>

</div>

</div>

</div>

1. Buka file orders.blade.php pada folder views/livewire, lalu tambahkan script berikut:

<div class="row">

@if ($errors->any())

<div class="alert alert-danger">

<strong>Whoops!</strong> There were some problems with your input.<br><br>

<ul>

@foreach ($errors->all() as $error)

<li>{{ $error }}</li>

@endforeach

</ul>

</div>

@endif

<div class="d-flex aligns-items-center justify-content-center">

<div class="col-lg-6">

<div class="card">

<div class="card-header">Invoice</div>

<div class="card-body">

<table class="table table-sm">

<tr>

<th>Invoice</th>

<th>:</th>

<td>{{ $data->invoice }}</td>

</tr>

<tr>

<th>Customer</th>

<th>:</th>

<td>{{ $data->customer->name}}</td>

</tr>

<tr>

<th>Tanggal</th>

<th>:</th>

<td>{{ $data->created\_at->format('d M Y')}}</td>

</tr>

<tr>

<th>Total</th>

<th>:</th>

<td>@money($total)</td>

</tr>

</table>

</div>

</div>

</div>

<div class="col-lg-6 ps-3">

<div class="card">

<div class="card-header">Produk </div>

<div class="card-body">

<form wire:submit.prevent="store" method="POST">

@csrf

<div class=" card-body">

<input type="hidden" name="order\_id" wire:model="order\_id" value="{{ $data->id }}">

<div class="form-group">

<label for="product\_id">Produk</label>

<select class="form-control" wire:model="product\_id" name="product\_id">

<option hidden>--Pilih Produk--</option>

@foreach($dataProduk as $dt )

<option value="{{ $dt->id }}">{{ $dt->name }}</option>

@endforeach

</select>

</div>

<div class="form-group">

<label for="qty">Jumlah</label>

<input type="number" class="form-control" wire:model="qty" name="qty">

</div>

</div>

<!-- /.card-body -->

<div class="card-footer text-end">

<button type="submit" class="btn btn-success btn-sm"><i class="fas fa-cart-plus"></i> Tambahkan</button>

</div>

</form>

</div>

</div>

</div>

</div>

<div class="d-flex aligns-items-center justify-content-center pt-3">

<div class="col-lg-12">

<div class="card">

<div class="card-header">Daftar Belanja</div>

<div class="card-body">

<table class="table table-sm">

<thead>

<tr>

<th>No</th>

<th>Nama Produk</th>

<th>Harga</th>

<th>Jumlah</th>

<th>Sub Total</th>

<th>Hapus</th>

</tr>

</thead>

<tbody>

@foreach($dataOrderDetail as $dt)

<tr>

<td>{{ $loop->iteration }}</td>

<td>{{ $dt->product->name }}</td>

<td>@money($dt->price)</td>

<td>{{ $dt->qty}}</td>

<td>@money($dt->price \* $dt->qty)</td>

<td><button class="btn btn-sm btn-danger" wire:click="delete({{ $dt->id }})"><i class=" fas fa-trash"></i></a></td>

</tr>

@endforeach

</tbody>

</table>

</div>

<div class="card-footer text-end">

<table class="table table-sm">

<tr>

<th>

<h2>Total</h2>

</th>

<th>

:

</th>

<th><h2>@money($total)</h2>

</th>

</tr>

<tr>

<th>

<h2>Uang</h2>

</th>

<th>

:

</th>

<th>

<input type="number" class="mt-2" wire:model="uang">

</th>

</tr>

<tr>

<th>

<h2>Kembali</h2>

</th>

<th>

:

</th>

<th><h2>

@money($kembali)</h2>

</th>

</tr>

</table>

<button class="btn btn-lg btn-success"><i class="fas fa-print"></i> Cetak</button>

</div>

</div>

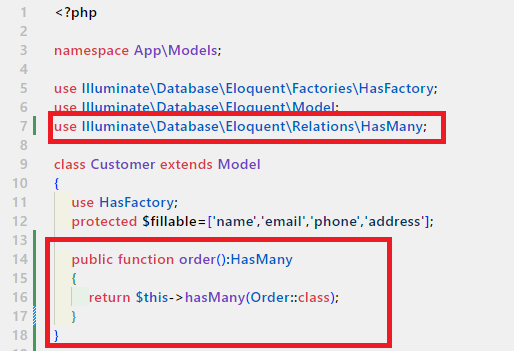
</div>

</div>

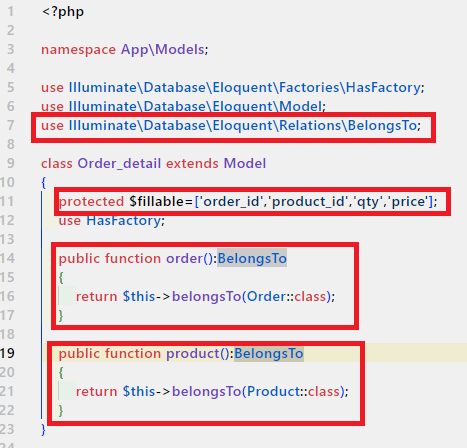
</div>

### Membuat seluruh relasi

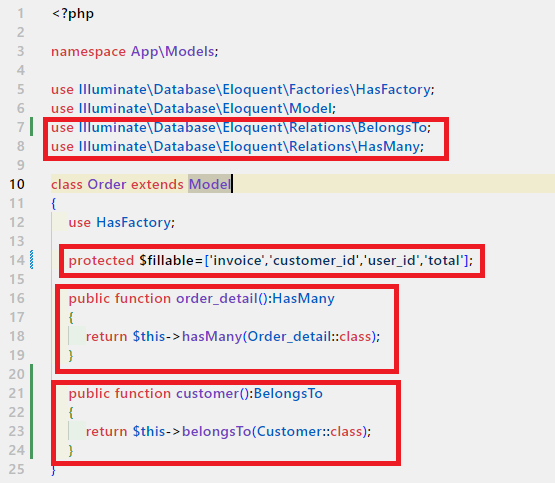
1. Relasi pada Model **Customer.php**



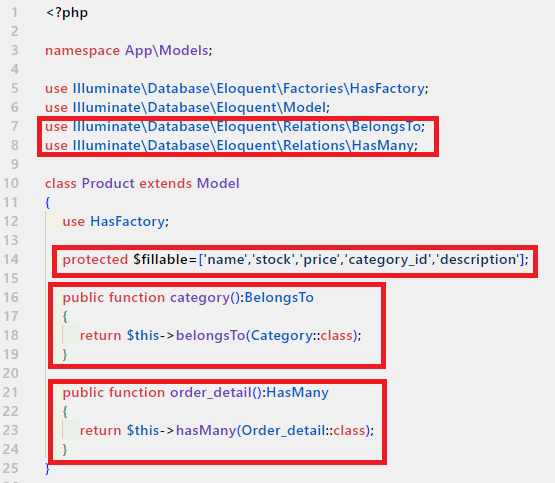
1. Relasi pada model Order\_detail.php



1. Relasi pada Model Order.php

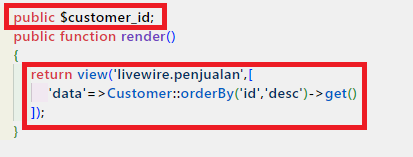


1. Relasi pada Model **Product.php**

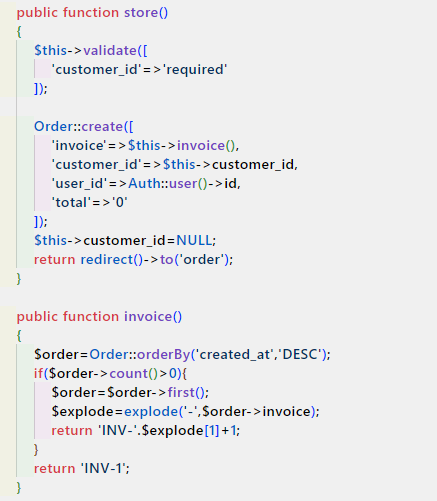


### Memperbaiki komponen livewire

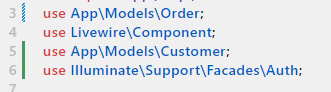
1. Buka file Penjualan.php yang terdapat pada folder Http\Livewire, kemudian tambahkan beberapa script berikut:



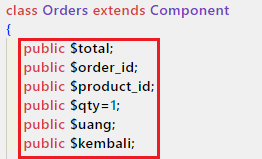
1. Masih di dalam class Penjualan, tambahkan function store dan function invoice berikut:



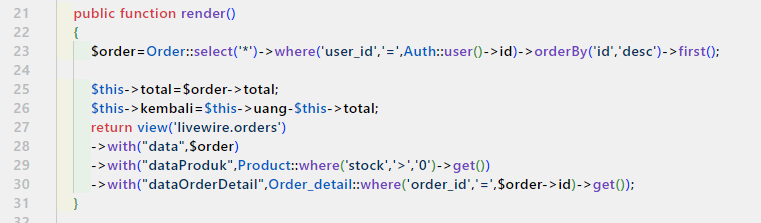
1. Jangan lupa untuk mengimport beberapa class yang sudah digunakan



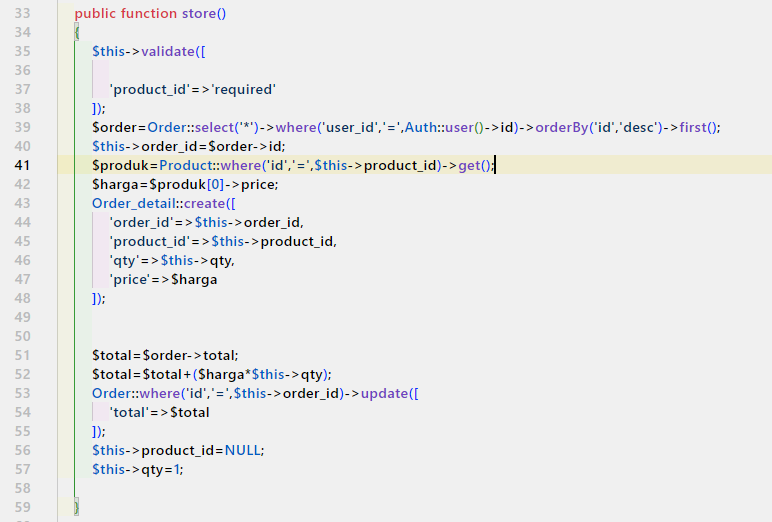
1. Pada file Orders.php yang terdapat di dalam folder Http\Livewire tambahkan property pada class.



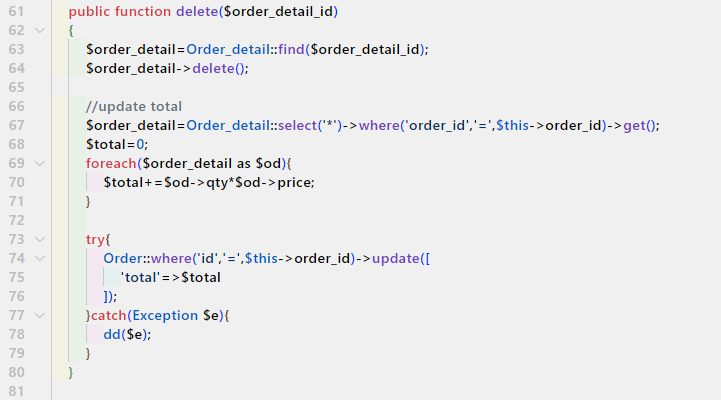
1. Perbaiki function render



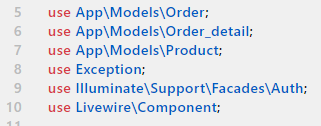
1. Tambahkan function store



1. Tambahkan function delete()

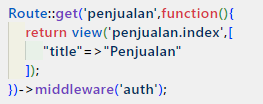


1. Jangan lupa untuk mengimport beberapa class yang telah digunakan.

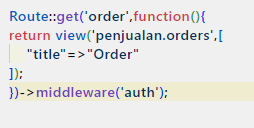


### Menambahkan route

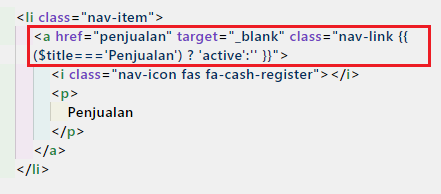
1. Buka file web.php pada folder routes
2. Tambahkan route penjualan



1. Tambahkan route order



1. Selanjutnya sebelum melakukan uji coba, buka file template.blade.php pada folder layouts, lalu perbaiki menu penjualan pada bagian side menu



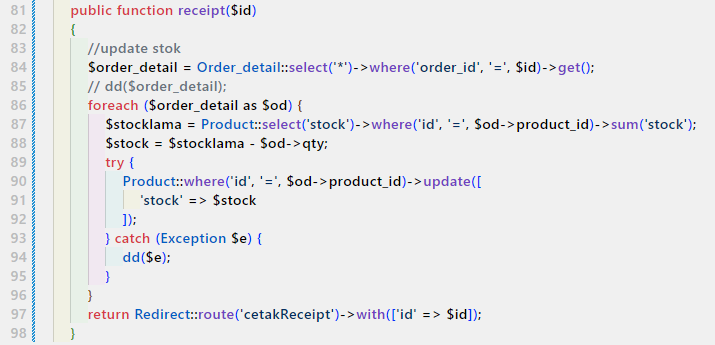
1. Silahkan lakukan uji coba

### Update Stok dan Cetak Receipt

1. Perbaiki script tombol cetak pada file **views/livewire/order.blade.php** seperti pada gambar berikut:



1. Selanjutnya buka file Orders.php pada folder App\Http\Livewire, lalu tambah function receipt berikut:



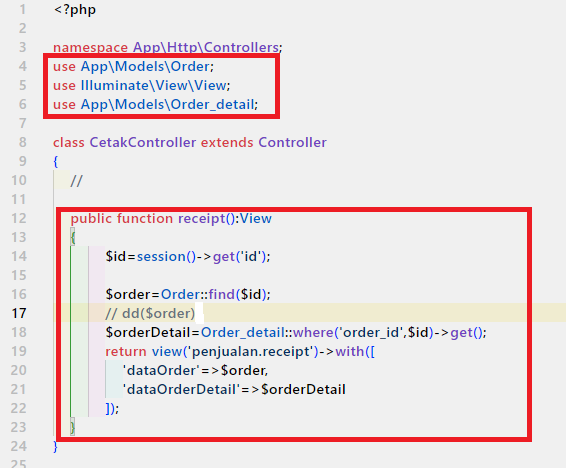
1. Buka file routes\web.php, tambahkan route berikut:



1. Buat controlerr baru, dengan nama CetakController

php artisan make:Controller CetakController

1. Lalu jangan lupa import class CetakController ke dalam routes/web.php
2. Buka Controller CetakController lalu tambahkan script berikut:



1. Buat file baru didalam folder **views/penjualan** dengan nama **receipt.blade.php**
2. Tambahkan script berikut

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Cetak Receipt</title>

<link rel="preconnect" href="https://fonts.googleapis.com">

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

<link href="https://fonts.googleapis.com/css2?family=Inconsolata:wght@300;400&display=swap" rel="stylesheet">

<style>

@page{

margin:0;

border:0;

size:80mm,100%;

}

body{

font-size: 9pt;

font-family: 'Inconsolata', monospace;

text-align: justify;

}

</style>

</head>

<body>

<p>invoice:{{ $dataOrder->invoice}}<br>Customer:{{ $dataOrder->customer->name}}<br>Cashier:{{ Auth::user()->name}}<br>Tanggal:{{$dataOrder->created\_at->format('d M Y H:m')}}</p>

<table>

<thead>

<tr>

<td>Product</td>

<td>Qty</td>

<td>Price</td>

<td>Amount</td>

</tr>

</thead>

@foreach($dataOrderDetail as $dod)

<tr>

<td>{{ $dod->product->name }}</td>

<td>{{$dod->qty}}</td>

<td>@money($dod->price)</td>

<td>@money($dod->price\*$dod->qty)</td>

</tr>

@endforeach

<tr>

<td colspan="3">Total:</td>

<td>@money($dataOrder->total)</td>

</tr>

</table>

<script src="{{ asset('plugins/jquery/jquery.min.js')}}"></script>

<script>

$(document).ready(function(){

window.print();

setInterval(myURL,5000);

});

function myURL() {

document.location.href = "{{ route('penjualan') }}";

// clearInterval(interval);

}

</script>

</body>

</html>

1. Lakukan uji coba kembali, pastikan stok terupdate dan semua menu dan tombol berfungsi dengan baik.

## Laporan Stok Barang

## Laporan Harian

## Laporan Bulanan

## Laporan Tahunan

# Perbaikan Project

## Memperbaiki Tabel Product

## Mencetak Barcode

## Menambahkan Discount bagi Customer

## Memperbaiki Tampilan Dashboard

**Keseluruhan script bisa dilihat di :** [**https://github.com/rizaru22/projectPOS**](https://github.com/rizaru22/projectPOS)**, jika ada yang kurang dari penjelasan dari script di atas, bisa berikan komentar di file ini, atau buat issue melalui github di atas.**