**Catatan Koding**

**Instalisasi**

1. Installisasi sesuai dengan versi Laravel kalian

composer create-project --prefer-dist laravel/laravel blog "5.7.\*"

// ini merupakan versi 5.7

**Database dan migration**

1. Buka phpMyAdmin dan buat database baru dan inisiasi ke Laravel pada folder .env

DB\_CONNECTION=mysql

DB\_HOST=127.0.0.1

DB\_PORT=3306

DB\_DATABASE=universitas // nama database

DB\_USERNAME=root

DB\_PASSWORD=

1. Buat migration di git bash

php artisan make:migration create\_mahasiswa\_table

// membuat migration untuk table di database dengan nama create\_mahasiswa\_table

* Mengatasi error ketika migrate masuk ke file AppServiceProvider.php
* Bagian atas tambahkan use Illuminate\Support\Facades\Schema;
* Dan di bagian function boot() tambahkan => Schema::defaultStringLength(191);

1. Menambahkan data atau field pada table

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

$table->increments('id');

$table->string('nama\_depan');

$table->string('nama\_belakang');

$table->string('jenis\_kelamin');

$table->string('agama');

$table->text('alamat');

$table->timestamps();

});

1. Melakukan migrate

php artisan migrate

Data Mahasiswa - Controller

1. Masuk ke serve Laravel dengan mengetik di git bash

php artisan serve

1. Buat route baru di file web.php

Route::get('/mahasiswa', 'MahasiswaController@index');

// ini merupakan contoh route dengan nama mahasiswa controller

1. Buat controllernya di git bash

php artisan make:controller namacontroller

1. Pada file controllernya di app/http/controller/namacontroller

class MahasiswaController extends Controller // nama controllernya mahasiswacontroller

{

public function index() // methodnya index

{

return view('mahasiswa.index'); // mengarahkan ke view di folder mahasiswa. Dan ke file index

}

}

template

1. buat folder dan filenya di resource/views/nama\_folder/nama\_file kemudian edit file sesuai yang ingin ditampilkan.

Data Mahasiswa - Model

1. Isi table dengan data di phpmyadmin



1. Buat model Mahasiswa untuk mengatur hubungan dengan database pada laravel

$ php artisan make:model Mahasiswa

Model created successfully.

1. Deklarasikan table sesuai dengan nama yang dibuat (supaya tidak jamak ‘s’)
2. Atur controller untuk terhubung dengan view dan modelnya

protected $table = 'mahasiswa';

$data\_mahasiswa = App\Mahasiswa::all();

return view('mahasiswa.index', ['data\_mahasiswa' => $data\_mahasiswa]);

Data Siswa- view

1. Mengecek data sudah masuk ke view
2. Memasukkan data ke table pada view dengan perulangan foreach

{{ dd($nama\_data) }}

<h1> Data Mahasiswa </h1>

<table>

<tr>

<!-- Nama judul tiap fiel di tabel yang dimunculkan -->

<th>Nama Depan</th>

<th>Nama Belakang</th>

<th>Jenis Kelamin</th>

<th>Agama</th>

<th>Alamat</th>

</tr>

<!-- Melakukan perulangan, memasukkan data\_mahasiswa ke variabel mahasiswa -->

@foreach($data\_mahasiswa as $mahasiswa)

<tr>

<!-- memilih data sesuai field ($data seluruh->nama\_fieldnya) -->

<td>{{$mahasiswa->nama\_depan}}</td>

<td>{{$mahasiswa->nama\_belakang}}</td>

<td>{{$mahasiswa->jenis\_kelamin}}</td>

<td>{{$mahasiswa->agama}}</td>

<td>{{$mahasiswa->alamat}}</td>

</tr>

<!-- penutup pengulangan -->

@endforeach

</table>

**Data siswa - bootstrap**

1. **Memasukkan bootstrap ke dalam kodingan**

**- Memasukkan CSS**

**- Memasukkan javascript**

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css" integrity="sha384-9aIt2nRpC12Uk9gS9baDl411NQApFmC26EwAOH8WgZl5MYYxFfc+NcPb1dKGj7Sk" crossorigin="anonymous">

**- Untuk membagi luas dari tampilan gunakan class=”col-…”,( ingat maksimal kontak col-12)**

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj" crossorigin="anonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js" integrity="sha384-Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo" crossorigin="anonymous"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js" integrity="sha384-OgVRvuATP1z7JjHLkuOU7Xw704+h835Lr+6QL9UvYjZE3Ipu6Tp75j7Bh/kR0JKI" crossorigin="anonymous"></script>

**- Menambah modal dengan bootstrap( tampilan mockup ketika di clik)**

<div class="col-6"> // membagi tampilan kotak 6

- **Insert Data**

Search di bootstrap dengan kata kunci “modal”

1. **Pada tag form tambahkan action=”/ alamat route” dan method=”POST”**
2. **Tambahkan csrv\_field dibawah tag form**

<form action="/mahasiswa/create" method="POST">

1. **pada penulisan route tujuan gunakan post**

Route::post('/mahasiswa/create','MahasiswaController@create');

{{csrf\_field()}}

1. **Pada controllernya untuk mengambil data input ke database**

\App\Mahasiswa::create($request->all());

return redirect(‘nama route yang dituju setelah memasukkan inputan’)

1. **Pada model nya deklarasikan field table yang akan diinput**
2. **Pada setiap inputan tambahkan name=”nama\_field”**

protected $fillable = ['nama\_depan', 'nama\_belakang', 'jenis\_kelamin', 'agama', 'alamat'];

**Membuat template master blade untuk memperingkas code front end**

<input name="nama\_depan" type="text" ….>

1. **Membuat file di dalam file views**
2. **Meng cut bagian awal kodingan pada file frontend yang diambil ( bagian head dan beberapa body) yang akan selalu muncul di dalam halaman yang akan di load**
3. **Meng cut bagian footer (akhir) pada kode yang akan slelau di munculkan di halaman yang ingin di load atau sama**
4. **Pada konten gunakan kode @yield sebagai parent dengan parameter yang bebas**

@yield('content') // penamaaan yield gunakan tanda kutip

1. **Pada file yang diselipkan konten gunakan @extends(‘nama file master bladenya’)**

@extends('layouts.master')

1. **Gunakan @section(‘nama parameter yield’) dan di tutup dengan @endsection**

@section('content')

Code . . . . .

@endsection

**- Hapus Data (delete)**

1. **Buat fitur hapus di frontend dengan memberikan alamat menggambil {id}**

<a href="/mahasiswa/{{$mahasiswa->id}}/delete" class="btn btn-danger btn-sm" onclick="return confirm('Yakin mau di hapus ?')">Delete</a>

1. **Buat routenya dengan tipe get**

Route::get('/mahasiswa/{id}/delete', 'MahasiswaController@delete');

1. **Membuat fungsi delete di controller**

public function delete($id) {

$mahasiswa = \App\Mahasiswa::find($id);

$mahasiswa->delete();

return redirect('/mahasiswa')->with('sukses', 'Data berhasil di hapus');

}

1. **Untuk notifikasi tambahkan javascript pada parameter di deletnya (code sepertii langkah 1 tadi)**

**Pemasangan template BackEnd (Klorofil)**

1. **Untuk memasukkan template tambahkan “ {{asset(‘letakfolder’)}}” pada javascript dan css**
2. **Tambahkan@ include(‘letakfolder) pada bagian yang ingin di masukkan**

@include('layouts.includes.\_sidebar')

<link rel="stylesheet" href="{{asset('admin/assets/vendor/bootstrap/css/bootstrap.min.css')}}">