|  |  |
| --- | --- |
| **Materi** | **Nilai** |
| Laravel dan Tailwind CSS vid 1 s/d 19 | 90 |

**Saya Sudah Belajar dan Mengerti dan Saya BISA**

1. Menggunakan laravel harus menginstal composer dahulu di web getcomposer.org. Kemudian, nyalakan XAMPP kemudian pada file xampp bagian htdocs buka cmd nya kemudian install laravelnya melalui composer dengan mengetik composer create-project laravel/laravel example-app(untuk nama projectnya). Lalu, buka project folder yang sudah dibuat tadi menggunakan cmdnya kemudian ketik code . untuk membuka code editornya. Kemudian, buka terminal dari projectnya tadi kemudian buka servernya menggunakan php artisan serve.
2. Menginstall nodejs kedalam laravel degan cara membuka terminal project laravelnya kemudian klik npm install. Kemudian, menginstall tailwind css di dalam laravel menggunakan nodejs dengan mengetik npm install -D tailwindcss di dalam terminal project laravel. Lalu menginstall file konfigurasi dari tailwind css dengan mengetik npx tailwindcss init di dalam terminal project laravel.
3. Menggunakan tailwind css ke dalam laravel. Dengan mengimport @tailwind base; @tailwind components; @tailwind utilities; ke dalam resources/css/app.css. Lalu, menambahkan perintah require('tailwindcss') kedalam file webpack.mix.js. Untuk melakukan mix menjalankan npm run watch ke dalam terminal prtoject laravel.
4. Menggunakan tailwind css ke dalam project laravel. Dengan cara menambahkan link css <link href="/css/app.css" rel="stylesheet"> ke dalam resources/views/layouts/app.blade.php. Setiap melakukan perubaan perintah di dalam project maka harus selalu melakukan npm run dev agar web project terefresh.
5. Melakukan Routing. Routing adalah penamaan suatu halaman web untuk berpindah dari halaman satau atau halaan lain. Dalam laravel ada 4 routing yaitu routing untuk api, routing untuk channel atau broadcast, routing untuk console atau artisan, dan routing untuk web. Untuk penulisan route ada dua cara, yang pertama adalah dengan menuliskan Route::get(‘login’(route yang dituju), function(){return (‘login(nama file yang dituju)’);}) digunakan untuk langsung menuliskan file yang dituju , yang kedua adalah dengan Route::get(‘login’, function(){return view (‘admin(nama folder).login(nama file)’);}) digunakan untuk menuju file yang berada di dalam folder dan tanda titik(.) digunakan untuk pengganti tanda garis miring(/), atau bisa menggunakan tanda garis miring secara langsung. Untuk penulisan file di dalam laravel untuk nama file untuk routingnya adalah dengan di tambahi dengan (nama file).blade.php.
6. Membuat Template. Template dalam laravel adalah suatu cara dimana agar merapikan tampilan coding dalam menaruh css menggunakan tailwind css. Cara penggunaannya adalah dengan membuat folder untuk template terlebih dahulu untuk memudahkan dan penataan file. Pertama, membuat file untk menaruh layout yang akan di taruh dengan template yang dibuat. Kedua, adalah dengan mengelink css dari tailwind dengan menggunakan cara <link rel="stylesheet" href="{{asset('css/app.css')}}"> kemudian, letakkan kepada bagian web yang ingin dipasangi template dengan menuliskan @yield(‘content(nama bagiannya)’) kemudian untuk penulisan isi template dan webnya adalah dengan file terpisah dengan menuliskan @extends(‘template(nama folder).layout(nama file layout template)’) @section(‘content(bagian yang diimpor)’) bagian yang ingin dipasangi template dan isi kontennya kemudian diakhiri dengan @endsection untuk pemasangan cssnya sama seperti tailwind css biasanya pada class bagian yang ingin di atur dengan css.
7. Membuat controller dengan php artisan make:controller HomeController(nama controller yang diinginkan). Cara menggunakan controller adalah dengan menuliskan dahulu routesnya di dalam file web.php di dalam folder routes dengan Route::get('/', [HomeController::class,'index']). Kemudian, di dalam HomeController isikan dengan function index, function index menjalankan view yang bernama home. Kemudian, dalam view yang bernama home menjalankan isi dari halamannya nanti dengan diisi @extends(‘template(nama folder).layout(nama file layout template)’) @section(‘content(bagian yang diimpor)’) bagian yang ingin dipasangi template dan isi kontennya kemudian diakhiri dengan @endsection.
8. Melakukan migrasi. Membuat file migrasi terlebih dahulu pada terminal mengetik php artisan make:migration kategori. Ketika sudah pada file migration yang sudah dibuat tadi ada dua funtion up dan down up untuk mengisikan dengan tabel yang ingin diisikan ke dalam migrationnya tadi dengan cara menuliskan Schema::create('kategoris', function (Blueprint $table) { $table>bigIncrements('idkategori'); $table->string('kategori'); $table->timestamps(); }) dan pada function down menuliskan Schema::dropIfExists('kategoris'). Untuk menaruhnya ke database adalah dengan melakukan php artisan migrate. Untuk menghapus migrasi yang barusan di masukkan ke dalam database adalah dengan menuliskan php artisan migrate:rollback.
9. Melakukan seeding atau mengirim data ke database. Membuat file untuk seeder adalah dengan menuliskan pada terminal php artisan make:seeder KategoriSeeder. Pada file seeder yang sudah dibuat tadi terdapat function run dan isikan dengan  DB::table('kategoris')->insert(['kategori' => 'Makanan' ]). Kemudian, pada DatabaseSeeder isikan dengan $this->call([ KategoriSeeder::class ]) untuk memanggil file seeder yang tadi dibuat. Lalu, ketika sudah siap maka lakukan pengiriman data pada terminal degan menuliskan php artisan db:seed.
10. Mengoneksian tabel dan juga kolom database dengan model. Cara membuat model adalah pada terminal menuliskan php artisan make:model Kategori(nama model). Pada file model yang sudah dibuat tadi diisi dengan protected $table = 'kategoris'(nama tabel yang dituju), protected $fillable = ['kategori'](kolom tabel yang dituju).
11. Mengisikan data menggunakan factory melalui model. Caranya adalah membuat file factory terlebih dahulu melalui terminal dengan menuliskan php artisan make:factory KategoriFactory. Pada file yanng baru dibuat tadi deklarasikan model yang digunakan protected $model = Kategori::class. Lalu, mengisikan function definition pada file factory tadi untuk mengisinya dengan data palsu atau faker return [‘kategori(nama kolom)' => $this->faker->name(data yang ingin dimasukkan)]. Kemudian, pada seeder lakukan pengisian jumlah data yang diinginkan dengan menuliskan Kategori::factory(5)->create().
12. Menampilkan data yang ada di dalam database ke dalam view atau web dengan menggunakan model. Caranya adalah pada controller isikan dan deklarasikan model yang digunakan dengan menuliskan pada function index $kategoris = Kategori::all(), return view('home', compact('kategoris')). Kemudian, pada file viewnya menuliskan dengan @foreach ($kategoris as $isi)<li>{{$isi->kategori}}</li> @endforeach.
13. Form suatu bagian yang digunakan untuk mengisikan data atau menampilkan data. Cara menggunakan form adalah pada bagian view home atau view yang digunakan oleh modelnya adalah dengan mengisikan dengan <form action="/store" method="get"> <input class="border" type="text" name="kategori"> <input type="submit" name="kategori" value="Simpan"></form> kemudian pada controllernya membuat function baru untuk action dari formnya public function store(Request $request){echo ‘store’} maka ketika button submitnya di klik akan menampilkan yang diisikan oleh function store yaitu akan menampilkan kata store. Kemudian, form yang digunakan untuk ngepost atau mengirim data yang kita tuliskan di ost adalah dengan menggunakan perintah csrf untuk mendapatkan tokennya maka, <form action="/store" method="post">@csrf<input class="border" type="text" name="kategori"><input type="submit" name="btn" value="Simpan"></form> kemudian pada controllernya ditambahkan dengan Request di dalam function storenya  public function store(Request $request){echo $request->kategori;}. Kemudian untuk routenya membuat route baru untuk action formnya yaitu dengan Route::get('/store', [HomeController::class,'store']) untuk mengambil data. Sedangkan untuk mengirim data form menggantinya dengan method post.
14. Melakukan insert. Degan cara pada bagian controlernya membuat variabel untuk class modelnya, kemudian mengambil kolom dari tambel yang ingin diisi, baru kemudian diisi, lalu disimpan atau penulisannya seperti $kategori = new Kategori();$kategori->kategori = $request->kategori;$kategori->save(). Kemudian, agar ketika kita mengisi data baru akan merefresh pada halaman yang dituju maka menambahkan redirect halaman yang dituju.
15. Melakukan update. Dengan cara pada controllernya membuat function baru untuk updatenya public function update($id, Request $request){$kategoris = ['kategori' =>$request->kategori,];Kategori::where('idkategori', $id)->update($kategoris);return redirect('/');}. Kemudian, pada formnya ditambahkan data mana yang ingin diubah dengan menggunakan id data pada bagian actonnya misal, /store/8. Kemudian, untuk routesnya isikan dengan Route::post('/store/{id}', [HomeController::class,'update']).
16. Menghapus data. Dengan cara pada controllernya membuat function baru untuk menghapus data yaitu dengan menuliskan public function destroy($id){Kategori::where('idkategori',$id)->delete();return redirect('/');}. Kemudian, pada routingnya menggunakan method get dan juga destroy Route::get('/store/{id}', [HomeController::class,'destroy']).
17. Menambahkan validasi data. Dengan cara pada controllernya menambahkan perintah di dalam function untuk insert dengan cara $this->validate($request, [‘kategori' => 'required | maks:30',]). Kemudian, untuk tampilannya pada file viewnya menggunakan @error('kategori')<p>{{$message}}</p>@enderror.
18. Mengupload gambar. Dengan cara membuat folder baru pada public untuk menyimpan gambarnya. Pada fomnya menambahkan pengaturan enctype="multipart/form-data". Kemudian menambahkan tombol pada form untuk menambahkan gambar <input type="file" name="image">. Kemudian pada controllernya menambahkan $name = $request->file('image')->getClientOriginalName(); kemudian untuk mengetahui apakah sudah berhasil atau belum dengan menambahkan dd($name) maka ketika berhasil terupload maka nama gambarnya akan tertampil. Kemudian untuk memindahkan gambar ke dalam folder project adalah dengan menambahkan $request->image->move(public\_path('images'),$name); kemudian menampilkan dd(‘sukses’) untuk mengetahui sudah berhasil atau belum dengan menampilkan pesan sukses.
19. Penggunaan tag php artisan make:model Menu –a yang digunakan untuk membentuk sebuah file secara keseluruhan di dalam project laravel.

**Saya Belum Mengerti**

2.

3.