

**TUGAS MODUL 11 PRAKTIKUM
PENGEMBANGAN APLIKASI WEBSITE
(PAW)**



oleh:

Kyla Salsabillah (IS-06-03 - 1204230061)

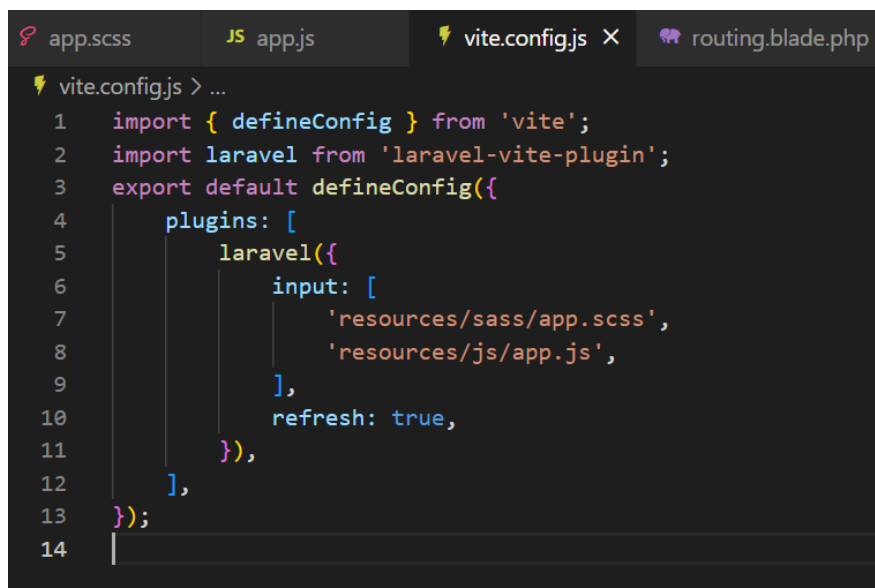
**PROGRAM STUDI SISTEM INFORMASI
FAKULTAS REKAYASA INDUSTRI
TELKOM UNIVERSITY SURABAYA
2024**

1. Generate Laravel Project dan LaravelUI

- buat project laravel baru via composer isi perintah “composer create-project laravel/laravel your-project-name” (your project name diisi sesuai dgn BelajarRouting)
- kemudian masuk ke dalam folder project laravel yang baru dibuat, dan install package laravel UI. Dgn masukan perintah “composer require laravel/ui”
- lalu setelah itu generate scaffolding untuk project Laravel berbasis css framework Bootstrap dgn masukan perintah “php artisan ui bootstrap”
- jalankan script “npm install” untuk compile scaffolding Bootstrap yang barusan disetup
- jalankan scrpt “php artisan serve” untuk mengaktifkan local development server Laravel sehingga aplikasi web dapat diakses melalui browser

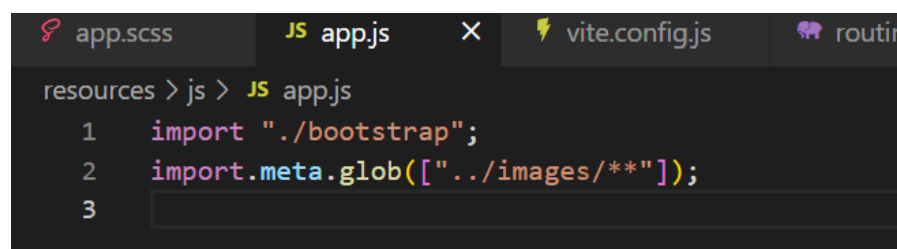
2. Bundling Asset dgn Vite

- Download dan install NodeJS melalui link (<https://nodejs.org/en/download/>)
- cek instalasi dgn memasukan perintah di command “node -v” dan “npm -v” scr satu-satu
- setelah itu install semua dependencies yang dibutuhkan untuk Bundling Asset dengan Vite, dgn masukan perintah “npm install”
- terapkan fitur “RefreshingonSave” dgn memeriksa file konfigurasi file vite pada vite.config.js sesuai perintah modul



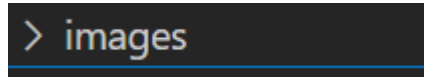
```
vite.config.js > ...
1  import { defineConfig } from 'vite';
2  import laravel from 'laravel-vite-plugin';
3  export default defineConfig({
4    plugins: [
5      laravel({
6        input: [
7          'resources/sass/app.scss',
8          'resources/js/app.js',
9        ],
10       refresh: true,
11     }),
12   ],
13 });
14
```

- Terapkan fitur “Processing Static Assets With Vite” dgn buka file /resources/js/app.js sesuaikan kode program seperti modul. Vite akan merujuk pada path direktori yang akan di definisikan buat mengambil asset gambar/image yang dibutuhkan nanti



```
resources > js > JS app.js
1  import './bootstrap';
2  import.meta.glob(["../images/**"]);
3
```

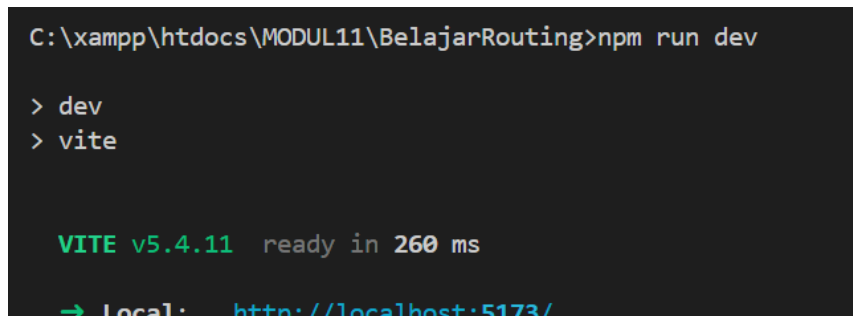
- Buat folder pada /resources dgn nama images dibuat meletakkan aset gambar/image yang akan digunakan pada website pada folder tersebut



- buka file View bernama welcome.blade.php. Lalu masukan kode seperti di gambar pada file view (blade) menggunakan Blade Directive @vite() buat memanggil asset gambar/image dgn pendekatan Vite seperti di bawah ini



- Jalankan Vite untuk mode development dengan memasukan perintah “npm run dev”



3. Install Bootstrap dan Bootstrap Icons Terbaru pada Project Laravel

- masukan perintah “npm install bootstrap@5.3.0-alpha3 bootstrap-icons @popperjs/core” pada terminal vscode buat menginstal bootstrap, popper dan bootstrap icons terbaru
- @import "bootstrap/scss/bootstrap"; digunakan buat mengimpor file scss utama bootstrap. Bagian aktif pada kode diatas berisi gaya dan komponen bootstrap biasa. Komentar pada font digunakan buat mengomentari import buat font karena tidak diperlukan pada implementasi saat ini

```
app.scss X JS app.js JS bootstrap.js vite.config.js routing
resources > sass > app.scss
1 // Fonts
2 // @import url("https://fonts.bunny.net/css?family=Nunito");
3 // Variables
4 // @import "variables";
5 // Bootstrap
6 @import "bootstrap/scss/bootstrap";
7 @import "bootstrap-icons/font/bootstrap-icons.css";
8
```

4. Praktik Lavel Routing

```
1 Route::get('/routing', function() {
2     return view('routing');
3 });
4
```

5. Routing.Blade.php

pemisahan antara kode view dan logika rute dapat membuat proyek lebih terstruktur, dan helper route() memastikan url berubah secara otomatis jika nama rute diubah

```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5     <meta charset="UTF-8">
6     <meta name="viewport" content="width=device-width,initial-scale=1.0">
7     <meta http-equiv="X-UA-Compatible" content="ie=edge">
8     <title>Belajar Laravel Routing</title>
9     @vite('resources/sass/app.scss')
10 </head>
```

Belajar Laravel Routing

6. Basic Routing

basic routing di Laravel digunakan untuk mendefinisikan rute yg sederhana tanpa memerlukan controller atau view, cukup mendefinisikan rute menggunakan `Route::get()` dan memberikan respons langsung

```
1 Route::get('/basic_routing', function() {  
2     return 'Hello World';  
3 });
```

Buka file `view routing.blade.php`

```
1 <a href="{{ url('/basic_routing') }}"  
2     class="list-group-item list-group-item-action -bs-info-bg-subtle text-white bg-primary">  
3     Basic Routing (No View, No Controller)  
4 </a>
```

7. View Route

tujuan file `view_route.blade.php` yaitu view Laravel yg menampilkan data dinamis seperti nama pengguna melalui sebuah variable yg dikirim dari controller atau rute

- buka file `routes/web.php`, praktikkan View Route

```
1 Route::view('/view_route', 'view_route');  
2 Route::view('/view_route', 'view_route', ['name' => 'Purnama']);  
3
```

- Buat file View dengan nama view_route.blade.php

```

1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5     <meta charset="UTF-8">
6     <meta name="viewport" content="width=device-width, initial
7 scale=1.0">
8     <meta http-equiv="X-UA-Compatible" content="ie=edge">
9     <title>View Route</title>
10    @vite('resources/sass/app.scss')
11 </head>
12
13 <body>
14     <div class="container m-5">
15         <h1>This is from View Route</h1>
16         <p>Hello, My name is {{ $name }}</p>
17     </div>
18     @vite('resources/js/app.js')
19 </body>
20
21 </html>

```

- Buka file view routing.blade.php

```

1 <a href="{{ url('/view_route') }}"
2     class="list-group-item list-group-item-action -bs-info-bg-subtle text-white bg-primary">
3     View Route
4 </a>

```

Belajar Laravel Routing

1. Basic Routing (No View, No Controller)

2. View Route

8. Controller Route

kode ini menunjukan cara untuk membuat tautan yg akan mengarahkan ke rute yg dikelola oleh controller

```

1 <?php
2
3 use Illuminate\Support\Facades\Route;
4 use App\Http\Controllers\RouteController;
5

```

```

1 <a href="{{ url('/controller_route') }}"
2   class="list-group-item list-group item-action -bs-info-bg-subtle text-white bg-primary">
3   Controller Route
4 </a>

```

import RouteController langkah penting buat memastikan bahwa controller dapat digunakan dalam mendefinisikan rute di Laravel

9. Redirect Route

redirect route dapat mempermudah pengalihan tanpa perlu membuat logika tambahan dalam controller. Hal ini sangat berguna buat kasus seperti memindahkan halaman

```

1
2 Route::get('/routing', function() {
3     return view('routing');
4 });

```

```

1 <a href="{{ url('/') }}"
2   class="list-group-item list-group item-action -bs-info-bg-subtle text-white bg-primary">
3   Redirect Route
4 </a>

```

10. Route Parameter (Required Parameter)

route parameter berguna buat mennagani url dgn mudah dan dapat mengambil nilai dari url lalu menggunakannya dalam aplikasi

```

1 Route::get('/user/{id}', function($id) {
2     return "User Id: ".$id;
3 });
4
5 Route::get('/posts/{post}/comments/{comment}', function($postId,
6 $commentId) {
7     return "Post Id: ".$postId.", Comment Id: ".$commentId;
8 });

```

buka file view routing.blade.php kemudian tambahkan kode sesuai modul

```
1 <a href="{{ url('/user/12345') }}"
2     class="list-group-item list-group item-action -bs-info-bg-subtle text-white bg-primary">
3     Route Parameter (Required Parameter) - 1
4
5     </a>
6 <a href="{{ url('/posts/01/comments/20') }}"
7     class="list-group-item list-group item-action -bs-info-bg-subtle text-white bg-primary">
8     Route Parameter (Required Parameter) - 2
9     </a>
10
```

11. Praktik router parameter

```
1 Route::get('username/{name?}', function($name = null) {
2     return 'Username: '.$name;
3 });
```

```
1 <a href="{{ url('/username') }}"
2     class="list-group-item list-group item-action -bs-info-bg-subtle text-white bg-primary">
3     Route Parameter (Optional Parameter)
4     </a>
```

12. Praktik Route With Regular Expression Constraints

```
1 Route::get('/title/{title}', function($title) {
2     return "Title: ".$title;
3 })->where('title', '[A-Za-z]+');
4
```

buka file view routing.blade.php kemudian tambahkan kode sesuai modul

```
1 <a href="{{ url('/title/this-is-my-title') }}"
2     class="list-group-item list-group item-action -bs-info-bg-subtle text-white bg-primary">
3     Route With Regular Expression Constraints
4     </a>
```


13. Named route

```
1 Route::get('/profile/{profileId}', [RouteController::class, 'profile'])->name('profileRouteName');
2
```

```
1 public function profile($profileId) {
2     return "This is Profile from Controller, profile id: ".$profileId;
3 }
4
```

```
1 <a href="{{ route('profileRouteName', ['profileId' => '123']) }}"
2     class="list-group-item list-group-item-action bg-primary text-white">
3     Named Route
4 </a>
```

14. Route priority

route priority di laravel sangat bergantung pada urutan deklarasi pada rute dalam file web.php. rute yg lebih spesifik seharusnya dapat mendeklarasi sebelum rute yg lebih umum

```
1 Route::get('/route_priority/user', function() {
2     return "This is Route 1";
3 });
4 Route::get('/route_priority/user', function() {
5     return "This is Route 2";
6 });
```

```
1 <a href="{{ url('/route_priority/user') }}"
2     class="list-group-item list-group-item-action -bs-info-bg-subtle text-white bg-primary">
3     Route Priority
4 </a>
```

15. Fallback routes

fallback routes di Laravel memungkinkan buat menangani sebuah permintaan yg tidak mempunyai sebuah rute yg sesuai dgn cara yg fleksible

```
1 Route::fallback(function() {
2     return 'This is Fallback Route';
3 });
4
```

```
1 <a href="{{ url('/asdqwezxc') }}"
2     class="list-group-item list-group item-action -bs-info-bg-subtle text-white bg-primary">
3     Fallback Routes
4 </a>
```

16. Route groups

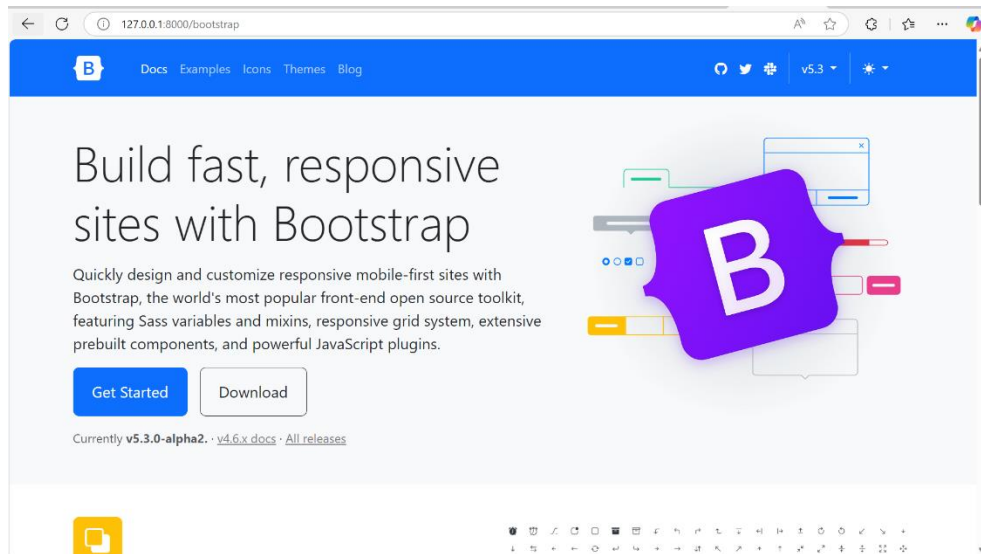
```
1 Route::prefix('admin')->name('admin.')->group(function() {
2     Route::get('/dashboard', function() {
3         return "This is admin dashboard";
4     })->name('dashboard');
5     Route::get('/users', function() {
6         return "This is user data on admin page";
7     })->name('users');
8     Route::get('/items', function() {
9         return "This is item data on admin page";
10    })->name('items');
11 });
```

```
1 <h6 class="mt-4">Route Groups (Route Prefixes & Route Name Prefixes)</h6>
2 <div class="list-group list-group-numbered mt-4">
3     <a href="{{ route('admin.dashboard') }}" class="list-group-item list-group-item-action">
4         Admin Dashboard
5     </a>
6     <a href="{{ route('admin.users') }}" class="list-group-item list-group-item-action">
7         Admin Users
8     </a>
```

- masukan perintah “php artisan route:list” pada terminal
- masukan perintah “php artisan route:cache” pada terminal buat menerapkan route caching
- masukan perintah “php artisan route:clear” terminal buat menghapus route cache

TUGAS

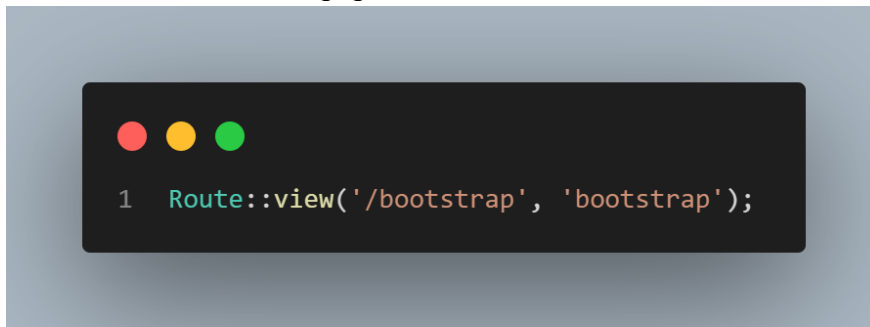
1. Clonning website bootstrap



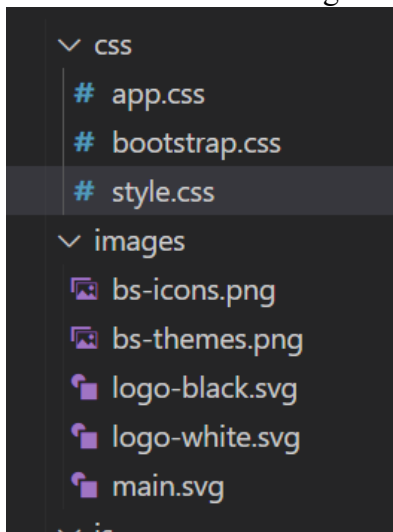
2. Route dan file View baru agar halaman Bootstrap Clone ini bisa diakses dari browser

```
1 // routes.js
2 const express = require('express');
3 const router = express.Router();
4
5 // Import controllers
6 const authController = require('../controllers/auth');
7 const userController = require('../controllers/user');
8 const postController = require('../controllers/post');
9 const commentController = require('../controllers/comment');
10 const likeController = require('../controllers/like');
11 const followController = require('../controllers/follow');
12 const notificationController = require('../controllers/notification');
13 const searchController = require('../controllers/search');
14 const profileController = require('../controllers/profile');
15
16 // Routes
17 // Auth routes
18 router.post('/register', authController.register);
19 router.post('/login', authController.login);
20 router.post('/logout', authController.logout);
21
22 // User routes
23 router.get('/users', userController.getAllUsers);
24 router.get('/users/:id', userController.getUserById);
25 router.put('/users/:id', userController.updateUser);
26 router.delete('/users/:id', userController.deleteUser);
27
28 // Post routes
29 router.post('/posts', postController.createPost);
30 router.get('/posts', postController.getAllPosts);
31 router.get('/posts/:id', postController.getPostById);
32 router.put('/posts/:id', postController.updatePost);
33 router.delete('/posts/:id', postController.deletePost);
34
35 // Comment routes
36 router.post('/posts/:postId/comments', commentController.createComment);
37 router.get('/posts/:postId/comments', commentController.getAllComments);
38 router.put('/posts/:postId/comments/:commentId', commentController.updateComment);
39 router.delete('/posts/:postId/comments/:commentId', commentController.deleteComment);
40
41 // Like routes
42 router.post('/posts/:postId/likes', likeController.createLike);
43 router.get('/posts/:postId/likes', likeController.getAllLikes);
44 router.delete('/posts/:postId/likes/:likeId', likeController.deleteLike);
45
46 // Follow routes
47 router.post('/users/:userId/follows', followController.createFollow);
48 router.get('/users/:userId/follows', followController.getAllFollows);
49 router.delete('/users/:userId/follows/:followId', followController.deleteFollow);
50
51 // Notification routes
52 router.get('/notifications', notificationController.getAllNotifications);
53 router.delete('/notifications/:notificationId', notificationController.deleteNotification);
54
55 // Search routes
56 router.get('/search', searchController.search);
57
58 // Profile routes
59 router.get('/profile/:userId', profileController.getUserProfile);
60 router.put('/profile/:userId', profileController.updateProfile);
61
62 // Error handling
63 router.use((err, req, res, next) => {
64   console.error(err.stack);
65   res.status(500).send('Something went wrong!');
66 });
67
68 module.exports = router;
```

3. tambahkan route di web.php



4. tambahkan css sama images



5. Link github : https://github.com/kylasalsa/modul11_belajarRouting-main.git