

LAPORAN PRAKTIKUM PEMROGRAMAN WEB
RESPONSI



Disusun oleh:

Nama : A. Agim Awaluddin

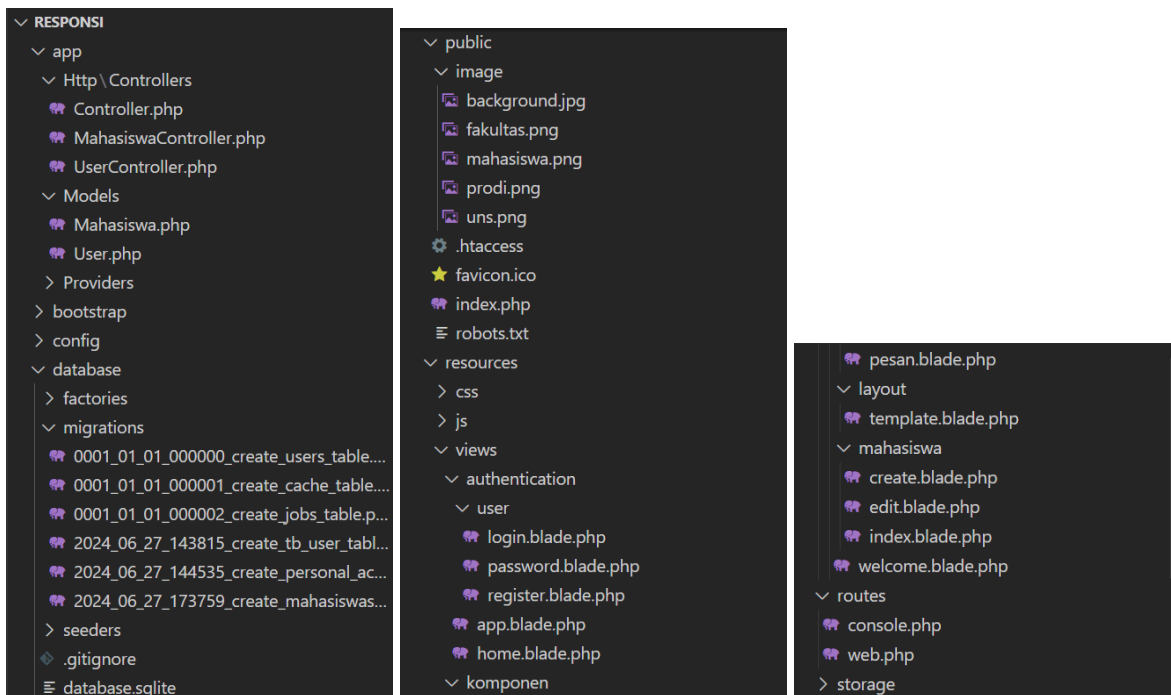
Nim : L0122007

Kelas : A

PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS DATA
UNIVERSITAS SEBELAS MARET

2024

1. Screenshot Source Code



A. Controllers/MahasiswaController.php

```
app > Http > Controllers > MahasiswaController.php
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use App\Models\Mahasiswa;
6  use Illuminate\Http\Request;
7  use Illuminate\Support\Facades\Session;
8
9  class MahasiswaController extends Controller
10 {
11     /**
12      * Display a listing of the resource.
13      */
14     public function index(Request $request)
15     {
16         $katakunci = $request->katakunci;
17         $jumlahbaris = 5;
18         if (strlen($katakunci)) {
19             $data = Mahasiswa::where('nim', 'like', "%$katakunci%")
20                 ->orWhere('name', 'like', "%$katakunci%")
21                 ->orWhere('email', 'like', "%$katakunci%")
22                 ->orWhere('jurusan', 'like', "%$katakunci%")
23                 ->orWhere('age', 'like', "%$katakunci%")
24                 ->orWhere('address', 'like', "%$katakunci%")
25                 ->paginate($jumlahbaris);
26         } else {
27             $data = Mahasiswa::orderBy('nim', 'desc')->paginate($jumlahbaris);
28         }
29         return view('mahasiswa.index')->with('data', $data);
30     }
31
32     /**
33      * Show the form for creating a new resource.
34      */
35     public function create()
```

```

35 public function create()
36 {
37     return view('mahasiswa.create');
38 }
39
40 /**
41  * Store a newly created resource in storage.
42  */
43 public function store(Request $request)
44 {
45     Session::flash('nim', $request->nim);
46     Session::flash('name', $request->name);
47     Session::flash('email', $request->email);
48     Session::flash('jurusan', $request->jurusan);
49     Session::flash('age', $request->age);
50     Session::flash('address', $request->address);
51
52     $request->validate([
53         'nim' => 'required|numeric|unique:mahasiswa,nim',
54         'name' => 'required|max:255',
55         'email' => 'required|email|unique:mahasiswa',
56         'jurusan' => 'required|max:255',
57         'age' => 'required|integer|min:0',
58         'address' => 'required',
59     ], [
60         'nim.required' => 'NIM wajib diisi',
61         'nim.numeric' => 'NIM wajib dalam angka',
62         'nim.unique' => 'NIM yang diisikan sudah ada dalam database',
63         'name.required' => 'Nama wajib diisi',
64         'email.required' => 'Email wajib diisi',
65         'jurusan.required' => 'Jurusan wajib diisi',
66         'age.required' => 'Umur wajib diisi',
67         'address.required' => 'Alamat wajib diisi',
68     ]);
69     $data = [
70         'nim' => $request->nim,
71         'name' => $request->name,
72         'email' => $request->email,
73         'jurusan' => $request->jurusan,
74         'age' => $request->age,
75         'address' => $request->address,
76     ];
77     Mahasiswa::create($data);
78     return redirect()->to('mahasiswa')->with('success', 'Berhasil menambahkan data');
79 }
80
81 /**
82  * Display the specified resource.
83  */
84 public function show(string $id)
85 {
86     //
87 }

```

```

88
89 /**
90  * Show the form for editing the specified resource.
91  */
92 public function edit(string $id)
93 {
94     $data = Mahasiswa::where('nim', $id)->first();
95     return view('mahasiswa.edit')->with('data', $data);
96 }
97
98 /**
99  * Update the specified resource in storage.
100  */
101 public function update(Request $request, string $id)
102 {
103     $request->validate([

```

```

104         'name' => 'required',
105         'email' => 'required',
106         'jurusan' => 'required',
107         'age' => 'required',
108         'address' => 'required',
109     ], [
110         'name.required' => 'Nama wajib diisi',
111         'email.required' => 'Email wajib diisi',
112         'jurusan.required' => 'Jurusan wajib diisi',
113         'age.required' => 'Umur wajib diisi',
114         'address.required' => 'Alamat wajib diisi',
115     ]
116 );
117 $data = [
118     'name' => $request->name,
119     'email' => $request->email,
120     'jurusan' => $request->jurusan,
121     'age' => $request->age,
122     'address' => $request->address,
123 ];
124 Mahasiswa::where('nim', $id)->update($data);
125 return redirect()->to('mahasiswa')->with('success', 'Berhasil melakukan update data');
126 }
127
128 /**
129  * Remove the specified resource from storage.
130  */
131
132 public function destroy(string $id)
133 {
134     Mahasiswa::where('nim', $id)->delete();
135     return redirect()->to('mahasiswa')->with('success', 'Berhasil melakukan delete data');
136 }

```

Source code di atas adalah bagian dari sebuah aplikasi web Laravel yang digunakan untuk mengelola data mahasiswa. Aplikasi ini memiliki struktur dasar MVC (Model-View-Controller) dengan menggunakan namespace App\Http\Controllers.

Namespace dan Penggunaan Class: Controller ini terletak di namespace App\Http\Controllers dan meng-import class mahasiswa dari model App\Models\mahasiswa. Selain itu, digunakan juga Request dan Session dari Laravel. Class mahasiswaController: Controller ini mengatur berbagai operasi terkait data mahasiswa.

Metode index: Metode ini menangani permintaan untuk menampilkan daftar mahasiswa. Jika terdapat kata kunci dalam request, akan dilakukan pencarian berdasarkan nim, name, email, jurusan, age, atau address. Hasil pencarian paginasi akan ditampilkan. Jika tidak ada kata kunci, data mahasiswa diurutkan berdasarkan nim secara descending. Metode create: Metode ini menampilkan form untuk menambahkan data mahasiswa.

Metode store: Metode ini menyimpan data mahasiswa yang baru ditambahkan ke dalam database. Sebelum menyimpan, dilakukan validasi untuk memastikan semua field wajib diisi dan sesuai aturan validasi. Jika validasi sukses, data disimpan dan pengguna dialihkan kembali ke halaman daftar mahasiswa dengan pesan sukses.

Metode edit: Metode ini menampilkan form untuk mengedit data mahasiswa berdasarkan nim yang diberikan. Metode update: Metode ini meng-update data mahasiswa berdasarkan nim. Seperti pada store, terlebih dahulu dilakukan validasi. Jika validasi sukses, data di-update dan pengguna dialihkan kembali ke halaman daftar mahasiswa dengan pesan sukses. Metode destroy: Metode ini menghapus data mahasiswa berdasarkan nim yang diberikan. Setelah penghapusan, pengguna dialihkan kembali ke halaman daftar mahasiswa dengan pesan sukses.

Validasi dan Flash Session: Pada metode store dan update, terdapat validasi data input dan penggunaan Session untuk menyimpan data inputan yang valid jika terjadi kesalahan validasi sehingga pengguna tidak perlu mengisi ulang form. Pesan Error: Terdapat pesan-pesan error kustom yang ditampilkan jika validasi gagal untuk memastikan input data sesuai dengan yang diharapkan.

B. Controllers/UserController.php

```
app > Http > Controllers > UserController.php
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use App\Models\User;
6  use Illuminate\Http\Request;
7  use Illuminate\Support\Facades\Auth;
8  use Illuminate\Support\Facades\Hash;
9
10 class UserController extends Controller
11 {
12     public function register()
13     {
14         $data['title'] = 'Register';
15         return view('authentication/user/register', $data);
16     }
17
18     public function register_action(Request $request)
19     {
20         $request->validate([
21             'name' => 'required',
22             'username' => 'required|unique:tb_user',
23             'password' => 'required',
24             'password_confirm' => 'required|same:password',
25         ]);
26
27         $user = new User([
28             'name' => $request->name,
29             'username' => $request->username,
30             'password' => Hash::make($request->password),
31         ]);
32         $user->save();
33
34         return redirect()->route('login')->with('success', 'Registration success. Please login!');
35     }
36
37     public function login()
38     {
39
```

```

40     $data['title'] = 'Login';
41     return view('authentication/user/login', $data);
42 }
43
44 public function login_action(Request $request)
45 {
46     $request->validate([
47         'username' => 'required',
48         'password' => 'required',
49     ]);
50     if (Auth::attempt(['username' => $request->username, 'password' => $request->password])) {
51         $request->session()->regenerate();
52         return redirect()->intended('/');
53     }
54
55     return back()->withErrors([
56         'password' => 'Wrong username or password',
57     ]);
58 }
59
60 public function password()
61 {
62     $data['title'] = 'Change Password';
63     return view('authentication/user/password', $data);
64 }
65
66 public function password_action(Request $request)
67 {
68     $request->validate([
69         'old_password' => 'required|current_password',
70         'new_password' => 'required|confirmed',
71     ]);
72     $user = User::find(Auth::id());
73     $user->password = Hash::make($request->new_password);
74     $user->save();
75     $request->session()->regenerate();
76     return back()->with('success', 'Password changed!');
77 }
78
79 public function logout(Request $request)
80 {
81     Auth::logout();
82     $request->session()->invalidate();
83     $request->session()->regenerateToken();
84     return redirect('/');
85 }
86 }
87

```

Source code di atas adalah bagian dari sebuah controller dalam aplikasi web berbasis Laravel yang mengelola proses otentikasi pengguna. Controller ini mencakup fungsi-fungsi untuk registrasi pengguna baru, login, logout, dan penggantian password.

Metode `register()` digunakan untuk menampilkan halaman registrasi dengan menggunakan view yang sesuai. Metode `register_action(Request $request)` menangani validasi input registrasi, membuat entitas User baru dengan password yang di-hash sebelum disimpan ke database, dan mengarahkan pengguna ke halaman login setelah berhasil registrasi.

Metode `login()` menampilkan halaman login, sementara `login_action(Request $request)` menangani proses login. Di dalamnya, terdapat validasi input username dan

password, serta menggunakan `Auth::attempt()` untuk melakukan autentikasi. Jika autentikasi berhasil, pengguna diarahkan ke halaman yang dimaksud, jika tidak, mereka kembali ke halaman login dengan pesan error.

Metode `password()` menampilkan halaman untuk mengganti password, sementara `password_action(Request $request)` menangani proses validasi dan perubahan password. Autentikasi digunakan dengan `current_password` untuk memastikan password lama sesuai sebelum mengubahnya.

Metode `logout(Request $request)` digunakan untuk proses logout pengguna dengan menghapus sesi, me-regenerate token sesi, dan mengarahkan pengguna kembali ke halaman utama setelah logout.

C. Models/Mahasiswa.php

```
app > Models > Mahasiswa.php
1  <?php
2
3  namespace App\Models;
4
5  use Illuminate\Database\Eloquent\Factories\HasFactory;
6  use Illuminate\Database\Eloquent\Model;
7
8  class Mahasiswa extends Model
9  {
10     use HasFactory;
11     protected $fillable = ['nim', 'name', 'email', 'jurusan', 'age', 'address'];
12     protected $table = 'mahasiswa';
13     public $timestamps = false;
14 }
15
```

Source code di atas adalah sebuah definisi dari model mahasiswa dalam aplikasi Laravel menggunakan bahasa PHP. Model ini terletak di namespace `App\Models` dan meng-extend dari kelas `Illuminate\Database\Eloquent\Model`.

Model mahasiswa menggunakan trait `HasFactory` yang disediakan oleh Eloquent untuk mempermudah pembuatan factory data. Property `$fillable` digunakan untuk menentukan kolom mana saja yang boleh diisi secara massal ketika data mahasiswa dibuat atau diupdate, yaitu `nim`, `name`, `email`, `jurusan`, `age`, dan `address`.

Property `$table` menentukan nama tabel yang terkait dengan model ini, dalam hal ini adalah `mahasiswa`. Property `$timestamps` diatur sebagai `false`, yang berarti Eloquent tidak akan secara otomatis memperbarui kolom `created_at` dan `updated_at` pada saat pembuatan atau pembaruan record dalam tabel mahasiswa.

D. Models/User.php

```
app > Models > User.php
1  <?php
2
3  namespace App\Models;
4
5  // use Illuminate\Contracts\Auth\MustVerifyEmail;
6  use Laravel\Sanctum\HasApiTokens;
7  use Illuminate\Notifications\Notifiable;
8  use Illuminate\Database\Eloquent\Factories\HasFactory;
9  use Illuminate\Foundation\Auth\User as Authenticatable;
10
11 class User extends Authenticatable
12 {
13     use HasApiTokens, HasFactory, Notifiable;
14
15     protected $table = 'tb_user';
16     protected $primaryKey = 'user_id';
17
18     protected $fillable = [
19         'name',
20         'username',
21         'password',
22     ];
23 }
24
```

Source code di atas adalah implementasi dari model User dalam sebuah aplikasi Laravel. Model ini didefinisikan di namespace `App\Models` dan meng-extend `Authenticatable` dari Laravel, sehingga memungkinkan model ini untuk berinteraksi dengan fitur otentikasi aplikasi. Model User menggunakan beberapa trait, termasuk `HasApiTokens`, `HasFactory`, dan `Notifiable`, yang memberikan fungsionalitas tambahan seperti autentikasi API, pembuatan data dengan factory, dan pengiriman notifikasi.

Properti `$table` digunakan untuk menentukan nama tabel dalam basis data yang terkait dengan model ini, dalam kasus ini adalah `'tb_user'`, sementara `$primaryKey` menentukan nama kolom kunci utama pada tabel, yang diatur menjadi `'user_id'`. Properti `$fillable` menentukan kolom mana yang dapat diisi secara massal, termasuk `'name'`, `'username'`, dan `'password'`, yang masing-masing mewakili nama lengkap pengguna, nama pengguna unik, dan sandi pengguna.

E. Migrations/tb_user

```
database > migrations > 2024_06_27_143815_create_tb_user_table.php
1  <?php
2
3  use Illuminate\Database\Migrations\Migration;
4  use Illuminate\Database\Schema\Blueprint;
5  use Illuminate\Support\Facades\Schema;
6
7  return new class extends Migration
8  {
9      /**
10       * Run the migrations.
11       */
12 
```



```

12     public function up(): void
13     {
14         Schema::create('tb_user', function (Blueprint $table) {
15             $table->id('user_id');
16             $table->string('name');
17             $table->string('username')->unique();
18             $table->string('password');
19             $table->timestamps();
20         });
21     }
22
23     /**
24      * Reverse the migrations.
25      */
26     public function down(): void
27     {
28         Schema::dropIfExists('tb_user');
29     }
30 };
31

```

Source code PHP di atas adalah sebuah migrasi database menggunakan framework Laravel. Migrasi ini bertujuan untuk membuat tabel 'tb_user' dengan beberapa kolom di dalamnya. Tabel ini memiliki kolom 'user_id' sebagai primary key yang otomatis bertambah nilainya, kolom 'name' untuk menyimpan nama pengguna, kolom 'username' untuk menyimpan username yang unik, kolom 'password' untuk menyimpan sandi pengguna, dan kolom 'timestamps' untuk mencatat waktu pembuatan dan pembaruan record. Metode up digunakan untuk menjalankan migrasi, sedangkan metode down untuk membatalkannya dengan menghapus tabel 'tb_user' jika diperlukan.

F. Migrations/mahasiswas

```

database > migrations > 2024_06_04_023308_create_mahasiswas_table.php
1  <?php
2
3  use Illuminate\Database\Migrations\Migration;
4  use Illuminate\Database\Schema\Blueprint;
5  use Illuminate\Support\Facades\Schema;
6
7  return new class extends Migration
8  {
9      /**
10       * Run the migrations.
11       */
12       public function up(): void
13       {
14           Schema::create('mahasiswa', function (Blueprint $table) {
15               $table->integer('nim');
16               $table->unique('nim');
17               $table->string('name');
18               $table->string('email')->unique();
19               $table->string('jurusan');
20               $table->integer('age');
21               $table->string('address');
22           });
23       }
24
25       /**
26       * Reverse the migrations.

```

```

27     */
28     public function down(): void
29     {
30         Schema::dropIfExists('mahasiswa');
31     }
32 };
33

```

Source code di atas adalah sebuah migration script dalam framework Laravel menggunakan fitur anonymous class untuk membuat tabel 'mahasiswa' di database. Dalam script ini, sebuah migration baru dibuat yang meng-extend kelas Migration. Pada metode up(), sebuah tabel dengan nama 'mahasiswa' dibuat menggunakan fasilitas Laravel Schema Builder. Tabel ini memiliki kolom-kolom sebagai berikut: 'nim' (integer), yang di-set sebagai unik menggunakan method unique, 'name' (string), 'email' (string) yang juga di-set sebagai unik, 'jurusan' (string), 'age' (integer), dan 'address' (string). Metode down() digunakan untuk meng-reverse migration dengan menghapus tabel 'mahasiswa' jika migrasi ini di-rollback. Teknik menggunakan anonymous class memungkinkan definisi migration yang lebih terfokus dan bersih dalam file migrasi Laravel.

G. View

1. Authentication/app.blade.php

```

resources > views > authentication > app.blade.php > html > body > div.container.mt-4 > h1
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, shrink-to-fit=no">
7      <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet">
8      <link href="https://fonts.googleapis.com/css2?family=Cabin:ital,wght@0,400..700;1,400..700&family=
9      <title>@yield('title', 'Your University')</title>
10     <style>
11         body{
12             margin-top: 0px;
13             background-image: url('image/background.jpg');
14             /* background-size: cover; */
15             background-position: center;
16             background-repeat: no-repeat;
17             background-size: 1570px;
18         }
19         .container {
20             flex: 1;
21         }
22
23         footer {
24             position: fixed;
25             bottom: 0;
26             width: 100%;
27             background-color: rgba(0, 0, 0, 0.5);
28             color: white;
29             text-align: center;
30
31             padding: 10px 0;
32         }
33     </style>
34 </head>

```

```

35 <body>
36   <header>
37     <!-- Navbar -->
38     <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
39       <div class="container">
40         <a class="navbar-brand" href="#">
41           
42           Universitas Sebelas Maret
43         </a>
44         <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav"
45           aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
46           <span class="navbar-toggler-icon"></span>
47         </button>
48         <div class="collapse navbar-collapse" id="navbarNav">
49           <ul class="navbar-nav ms-auto">
50             <li class="nav-item">
51               <a class="nav-link" href="{ route('home') }">Home</a>
52             </li>
53             <li class="nav-item">
54               <a class="nav-link" href="#">About</a>
55             </li>
56             <li class="nav-item">
57               <a class="nav-link" href="#">Contact</a>
58             </li>
59           </ul>
60         </div>
61       </div>
62     </nav>
63   </header>
64
65   <div class="container mt-4">
66     <h1>@yield('title', 'Your University')</h1>
67     @yield('content')
68   </div>
69
70   <footer class="py-2 mt-5">
71     <p>&copy; 2024 Universitas Sebelas Maret. All rights reserved.</p>
72   </footer>
73
74   <!-- Bootstrap JS -->
75   <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"></script>
76 </body>
77
78 </html>

```

Source code di atas adalah sebuah template HTML untuk halaman web yang menggunakan Bootstrap dan Google Fonts. Template ini memiliki struktur dasar seperti `<!doctype html>`, `<html>`, `<head>`, dan `<body>`. Di dalam `<head>`, terdapat pengaturan karakter, pengaturan tampilan responsif, dan tautan ke stylesheet Bootstrap serta font dari Google Fonts. Judul halaman diatur dengan `@yield('title', 'Your University')`, yang berarti dapat diubah sesuai kebutuhan. Bagian `<style>` mendefinisikan beberapa gaya CSS untuk latar belakang dan footer halaman.

Di dalam `<body>`, terdapat bagian `<header>` yang berisi navbar Bootstrap dengan logo dan nama "Universitas Sebelas Maret" serta beberapa tautan navigasi. Navbar ini menggunakan kelas-kelas Bootstrap untuk tampilan yang responsif dan bergaya gelap. Kemudian, bagian utama konten halaman diletakkan dalam `<div class="container mt-4">`,

yang berfungsi sebagai kontainer fleksibel untuk menampilkan judul dan konten yang dapat diubah dengan @yield.

Terakhir, terdapat bagian <footer> yang tetap berada di bawah halaman, berisi teks hak cipta. Di bagian paling bawah, terdapat penyertaan skrip JavaScript Bootstrap untuk fungsionalitas interaktif.

2. Aunthentication/home.blade.php

```
resources > views > authentication > home.blade.php > div.row > div.col-md-8.offset-md-2 > div.card.bg-transparent.mt-4 > div
1  @extends('authentication.app')
2
3  @section('content')
4  <div class="row">
5      <div class="col-md-8 offset-md-2">
6          @auth
7              <div class="card bg-transparent mt-4">
8                  <div class="card-body text-center">
9                      <p class="card-text" style="color: white;">Welcome <b>{{ Auth::user()->name }}</b></p>
10
11                      <div class="btn-group btn-group-toggle data-toggle="buttons" style="display: flex; gap: 10px;">
12                          <a class="btn btn-secondary mr-10" style="width: 200px; height: 150px;">
13                              
14                              <p style="margin-top: 5px; font-family: 'Cabin';">Data Fakultas</p>
15                          </a>
16                          <a class="btn btn-secondary mr-10" style="width: 200px; height: 150px;">
17                              
18                              <p style="margin-top: 5px; font-family: 'Cabin';">Data Program Studi</p>
19                          </a>
20                          <a href="mahasiswa" class="btn btn-secondary" style="width: 200px; height: 150px;">
21                              
22                              <p style="margin-top: 5px; font-family: 'Cabin';">Data Mahasiswa</p>
23                          </a>
24                      </div>
25
26                      <div class="mt-5">
27                          <a class="btn btn-outline-primary" href="{{ route('password') }}">Change Password</a>
28                          <a class="btn btn-outline-danger" href="{{ route('logout') }}">Logout</a>
29                      </div>
30                  </div>
31              </div>
32          @endauth
33          @guest
34              <div class="card bg-transparent border-0 mt-5">
35                  <p style="color: #BBE9FF; text-align: center; font-family: 'Cabin'; font-size: 17px; margin-bottom: 10px;">
36                      <P style="color: white; text-align: center; font-weight: bold; font-family: 'Cabin'; font-size: 17px;">
37                          <p style="color: white; text-align: center; font-weight: bold; font-family: 'Cabin'; font-size: 17px;">
38                              <div class="card-body text-center">
39                                  <a class="btn btn-outline-info" href="{{ route('login') }}">Login</a>
40                                  <a class="btn btn-outline-info" href="{{ route('register') }}">Register</a>
41                              </div>
42                          </p>
43                      </div>
44                  @endguest
45              </div>
46          </div>
47      @endsection
48
```

Source code di atas adalah template tampilan halaman yang menggunakan framework Laravel Blade. Kode tersebut meng-extend layout dari 'authentication.app' dan mendefinisikan bagian konten utama dengan section 'content'. Halaman ini dibagi menjadi

dua bagian, satu untuk pengguna yang sudah login (authenticated) dan satu untuk pengguna yang belum login (guest).

Jika pengguna sudah login (diketahui dengan @auth), akan ditampilkan kartu (card) dengan pesan selamat datang yang menyebutkan nama pengguna yang sedang login. Di dalam kartu tersebut terdapat tiga tombol yang masing-masing berfungsi untuk mengakses data fakultas, data program studi, dan data mahasiswa, dengan masing-masing tombol memiliki gambar dan teks yang sesuai. Ada juga dua tombol untuk mengubah kata sandi dan keluar (logout).

Jika pengguna belum login (diketahui dengan @guest), akan ditampilkan kartu dengan pesan "Do You Need Our Help?" dan "WELCOME TO OUR UNIVERSITY" yang mengajak pengguna untuk login atau mendaftar. Tombol login dan register disediakan di bawah pesan tersebut.

3. Authentication/user/login.blade.php

```
resources > views > authentication > user > login.blade.php > div.row > div.col-md-6 > form > div.mb-3
1  @extends('authentication/app')
2  @section('content')
3  <div class="row">
4      <div class="col-md-6">
5          @if(session('success'))
6              <p class="alert alert-success">{{ session('success') }}</p>
7          @endif
8          @if($errors->any())
9              @foreach($errors->all() as $err)
10                 <p class="alert alert-danger">{{ $err }}</p>
11             @endforeach
12         @endif
13         <form action="{{ route('login.action') }}" method="POST">
14             @csrf
15             <div class="mb-3" style="color: white;">
16                 <label>Username <span class="text-danger">*</span></label>
17                 <input class="form-control" type="text" name="username" value="{{ old('username') }}" />
18             </div>
19             <div class="mb-3" style="color: white;">
20                 <label>Password <span class="text-danger">*</span></label>
21                 <input class="form-control" type="password" name="password" />
22             </div>
23             <div class="mb-3">
24                 <button class="btn btn-outline-primary">Login</button>
25                 <a class="btn btn-outline-danger" href="{{ route('home') }}">Back</a>
26             </div>
27         </form>
28     </div>
29 </div>
30 @endsection
31
```

Source code di atas merupakan sebuah template view dalam framework Laravel yang digunakan untuk halaman login. Template ini menggunakan Blade templating engine dengan meng-extend layout dari authentication/app. Pada bagian @section('content'),

terdapat div row dengan col-md-6 untuk mengatur layout. Di dalamnya, jika ada session bernama 'success', maka akan ditampilkan pesan sukses dalam bentuk alert hijau. Jika terdapat error, semua error akan ditampilkan dalam alert merah.

Formulir login tersebut mengarah ke route login.action dengan metode POST dan menggunakan CSRF token untuk keamanan. Form ini memiliki dua input: satu untuk username dan satu lagi untuk password. Kedua input ini dilengkapi dengan label berwarna putih dan penanda wajib diisi (span dengan kelas text-danger). Selain itu, terdapat dua tombol: satu untuk mengirimkan form (Login) dan satu lagi untuk kembali ke halaman utama (Back). Keduanya menggunakan kelas btn btn-outline untuk styling tombol.

4. Authentication/user/password.blade.php

```
resources > views > authentication > user > password.blade.php > div.row > div.col-md-6 > form > div.mb-3 > a.btn.btn-outli
1  @extends('authentication/app')
2  @section('content')
3  <div class="row">
4      <div class="col-md-6">
5          @if(session('success'))
6              <p class="alert alert-success">{{ session('success') }}</p>
7          @endif
8          @if($errors->any())
9              @foreach($errors->all() as $err)
10                 <p class="alert alert-danger">{{ $err }}</p>
11             @endforeach
12         @endif
13         <form action="{{ route('password.action') }}" method="POST">
14             @csrf
15             <div class="mb-3">
16                 <label>Password <span class="text-danger">*</span></label>
17                 <input class="form-control" type="password" name="old_password" />
18             </div>
19             <div class="mb-3">
20                 <label>New Password <span class="text-danger">*</span></label>
21                 <input class="form-control" type="password" name="new_password" />
22             </div>
23             <div class="mb-3">
24                 <label>New Password Confirmation<span class="text-danger">*</span></label>
25                 <input class="form-control" type="password" name="new_password_confirmation" />
26             </div>
27             <div class="mb-3">
28                 <button class="btn btn-outline-primary">Change</button>
29                 <a class="btn btn-outline-danger" href="{{ route('home') }}">Back</a>
30             </div>
31         </form>
32     </div>
33 </div>
34 @endsection
35
```

Kode di atas merupakan bagian dari aplikasi Laravel yang mengelola perubahan kata sandi pengguna. Pada bagian ini, terdapat formulir HTML yang memungkinkan pengguna untuk memasukkan kata sandi lama, kata sandi baru, dan konfirmasi kata sandi baru. Jika ada pesan sukses dalam sesi (session), pesan tersebut akan ditampilkan sebagai alert berwarna hijau. Jika ada kesalahan (errors) saat pengisian formulir, masing-masing

kesalahan akan ditampilkan sebagai alert berwarna merah. Formulir ini menggunakan metode POST dan diarahkan ke rute 'password.action'. Ada tiga input untuk kata sandi lama, kata sandi baru, dan konfirmasi kata sandi baru, masing-masing diberi label dan span dengan kelas 'text-danger' untuk menunjukkan bahwa input tersebut wajib diisi. Setelah pengguna mengisi formulir, mereka dapat menekan tombol "Change" untuk mengirimkan data atau tombol "Back" untuk kembali ke halaman utama. Kode ini diapit dalam blok @extends yang menunjukkan bahwa template ini memperluas template 'authentication/app' dan bagian 'content' diisi dengan kode formulir tersebut.

5. Authentication/user/register.blade.php

```
resources > views > authentication > user > register.blade.php > div.row > div.col-md-6 > form > div.mb-3
1 extends('authentication/app')
2 section('content')
3 div class="row"
4     <div class="col-md-6">
5         @if($errors->any())
6             @foreach($errors->all() as $err)
7                 <p class="alert alert-danger">{{ $err }}</p>
8             @endforeach
9         @endif
10        <form action="{{ route('register.action') }}" method="POST">
11            @csrf
12            <div class="mb-3" style="color: white;">
13                <label>Name <span class="text-danger">*</span></label>
14                <input class="form-control" type="text" name="name" value="{{ old('name') }}" />
15            </div>
16            <div class="mb-3" style="color: white;">
17                <label>Username <span class="text-danger">*</span></label>
18                <input class="form-control" type="text" name="username" value="{{ old('username') }}" />
19            </div>
20            <div class="mb-3" style="color: white;">
21                <label>Password <span class="text-danger">*</span></label>
22                <input class="form-control" type="password" name="password" />
23            </div>
24            <div class="mb-3" style="color: white;">
25                <label>Password Confirmation <span class="text-danger">*</span></label>
26                <input class="form-control" type="password" name="password_confirm" />
27            </div>
28            <div class="mb-3">
29                <button class="btn btn-outline-primary">Register</button>
30                <a class="btn btn-outline-danger" href="{{ route('home') }}">Back</a>
31            </div>
32        </form>
33    </div>
34 </div>
35 endsection
36
```

Source code di atas adalah sebuah template untuk halaman registrasi pengguna yang menggunakan Blade template engine dalam framework Laravel. Template ini diperluas dari authentication/app dan mendefinisikan sebuah section content. Di dalam section ini, terdapat sebuah form registrasi yang menampilkan berbagai input field untuk nama, username, password, dan konfirmasi password. Jika terdapat error, pesan kesalahan akan ditampilkan dalam bentuk alert berwarna merah. Form ini menggunakan metode POST dan

mengarah ke route register.action. Selain itu, terdapat dua tombol, yaitu tombol "Register" untuk mengirimkan form dan tombol "Back" yang mengarah ke halaman home. Elemen-elemen dalam form ini juga disertai dengan validasi dan penggunaan token CSRF untuk keamanan. Seluruh label dan input field dalam form ini memiliki warna teks putih.

6. Komponen -> pesan.blade.php

```
resources > views > komponen > pesan.blade.php > ...
1  @if (Session::has('success'))
2      <div class="pt-3">
3          <div class="alert alert-success">
4              {{ Session::get('success') }}
5          </div>
6      </div>
7  @endif
8
9  @if ($errors->any())
10     <div class="pt-3">
11         <div class="alert alert-danger">
12             <ul>
13                 @foreach ($errors->all() as $item)
14                     <li>{{ $item }}</li>
15                 @endforeach
16             </ul>
17         </div>
18     </div>
19 @endif
20
```

Source code di atas adalah potongan dari sebuah template blade pada aplikasi Laravel. Pertama, kode ini menggunakan kontrol @if dari Blade untuk mengecek apakah ada pesan 'success' dalam session. Jika ada, maka akan menampilkan sebuah div dengan kelas alert alert-success yang berisi pesan yang diambil dari session tersebut.

Selanjutnya, kode juga menggunakan @if untuk mengecek apakah ada kesalahan validasi yang dikirimkan kembali dari server (dalam hal ini, menggunakan \$errors yang merupakan variabel yang tersedia secara otomatis di Laravel setelah validasi). Jika ada kesalahan, kode akan menampilkan div dengan kelas alert alert-danger yang berisi daftar pesan kesalahan yang diulang menggunakan loop @foreach.

Kode tersebut digunakan untuk memberikan umpan balik kepada pengguna terkait status operasi yang dilakukan, seperti pemberitahuan kesuksesan atau daftar kesalahan validasi yang mungkin terjadi selama proses penginputan data. Ini adalah bagian penting dari praktik pengembangan web untuk meningkatkan pengalaman pengguna dengan memberikan respons yang jelas dan informatif.

7. Layout/template.blade.php

```
resources > views > layout > template.blade.php > html > body.bg-light > main.container
1 <!doctype html>
2 <html lang="en">
3   <head>
4     <meta charset="utf-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1">
6     <title>Data Mahasiswa</title>
7     <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css" rel="stylesheet">
8   </head>
9   <body class="bg-light">
10    <main class="container">
11      @include('komponen.pesan')
12      @yield('konten')
13    </main>
14    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js" integrity="sha384-q8R/X238h3XJJbE0baKtW1keMiXrBbUkr31vW46BEE0132Z/46Cl0LN8+8Z" crossorigin="anonymous"></script>
15  </body>
16 </html>
17
18
```

Source code di atas adalah sebuah template HTML untuk halaman web yang berjudul "Data Mahasiswa." File ini menggunakan framework Bootstrap versi 5.2.1 untuk styling dan responsivitas. Di bagian <head>, terdapat meta tag untuk pengaturan karakter dan viewport agar halaman terlihat baik pada berbagai perangkat, serta link ke file CSS Bootstrap. Bagian <body> memiliki kelas "bg-light" yang memberikan latar belakang berwarna terang pada halaman. Di dalam <main> dengan kelas "container", terdapat dua directive Blade (@include dan @yield) yang digunakan dalam framework Laravel untuk menyisipkan komponen pesan dan konten dinamis. Terakhir, terdapat tag <script> untuk memuat file JavaScript Bootstrap, yang diambil dari CDN (Content Delivery Network) untuk mendukung fitur interaktif pada halaman web.

8. Mahasiswa/create.blade.php

```
resources > views > mahasiswa > create.blade.php > form > div.my-3.p-3.bg-body.rounded.shadow-sm > a.btn.btn-secondary >
1 @extends('layout.template')
2 <!-- START FORM -->
3 @section('konten')
4
5 <form action="{{ url('mahasiswa') }}" method='post'>
6   @csrf
7   <div class="my-3 p-3 bg-body rounded shadow-sm">
8     <a href="{{ url('mahasiswa') }}" class="btn btn-secondary">< kembali</a>
9     <div class="mb-3 row">
10       <label for="nim" class="col-sm-2 col-form-label">NIM</label>
11       <div class="col-sm-10">
12         <input type="number" class="form-control" name="nim" value="{{ Session::get('nim') }}" id="nim">
13       </div>
14     </div>
15     <div class="mb-3 row">
16       <label for="name" class="col-sm-2 col-form-label">Nama</label>
17       <div class="col-sm-10">
18         <input type="text" class="form-control" name="name" value="{{ Session::get('name') }}" id="name">
19       </div>
20     </div>
21   </div>
22 </form>
23
24
```

```

20 </div>
21 <div class="mb-3 row">
22     <label for="email" class="col-sm-2 col-form-label">Email</label>
23     <div class="col-sm-10">
24         <input type="text" class="form-control" name='email' value="{{ Session::get('email') }}"
25     </div>
26 </div>
27 <div class="mb-3 row">
28     <label for="jurusan" class="col-sm-2 col-form-label">Jurusan</label>
29     <div class="col-sm-10">
30         <input type="text" class="form-control" name='jurusan' value="{{ Session::get('jurusan') }}"
31     </div>
32 </div>
33 <div class="mb-3 row">
34     <label for="age" class="col-sm-2 col-form-label">Umur</label>
35     <div class="col-sm-10">
36         <input type="number" class="form-control" name='age' value="{{ Session::get('age') }}" id="
37     </div>
38 </div>
39 <div class="mb-3 row">
40     <label for="address" class="col-sm-2 col-form-label">Alamat</label>
41     <div class="col-sm-10">
42         <input type="text" class="form-control" name='address' value="{{ Session::get('address') }}"
43     </div>
44 </div>
45 <div class="mb-3 row">
46     <label for="address" class="col-sm-2 col-form-label"></label>
47     <div class="col-sm-10"><button type="submit" class="btn btn-success" name="submit">Simpan</but
48 </div>
49 </div>
50 </form>
51 <!-- AKHIR FORM -->
52 @endsection

```

Source code di atas adalah sebuah template blade yang digunakan dalam framework Laravel untuk membuat halaman form tambah data mahasiswa. Hal ini diasumsikan berdasarkan penggunaan fungsi `Session::get()` yang mengindikasikan bahwa nilai-nilai yang diisi pada form ini dapat diambil dari session sebelumnya. Pada template ini:

- Form dibungkus dalam div dengan class `my-3 p-3 bg-body rounded shadow-sm` untuk memberikan padding, background, border radius, dan shadow.
- Tombol "kembali" dibuat dengan menggunakan anchor `<a>` yang mengarahkan kembali ke halaman 'mahasiswa'.
- Input form menggunakan elemen `<input>` dengan berbagai jenis (number, text) yang memiliki label dan kolom input berdampingan menggunakan grid system Bootstrap (`col-sm-2` untuk label dan `col-sm-10` untuk input).
- Data input diinisialisasi dengan nilai-nilai dari session sebelumnya, masing-masing memiliki atribut name yang sesuai untuk diolah oleh Laravel dalam request.
- Tombol "Simpan" berfungsi sebagai submit untuk mengirimkan form ke URL 'mahasiswa' dengan metode POST.
- Directive Blade `@csrf` digunakan untuk melindungi form dari serangan CSRF dengan menambahkan token CSRF yang dibutuhkan oleh Laravel.

Template ini digunakan untuk memfasilitasi pembuatan dan penyimpanan data mahasiswa dalam aplikasi berbasis Laravel dengan tampilan yang bersih dan responsif menggunakan Bootstrap.

9. Mahasiswa/edit.blade.php

```
resources > views > mahasiswa > edit.blade.php > ...
1  @extends('layout.template')
2  <!-- START FORM -->
3  @section('konten')
4
5  <form action='{{ url('mahasiswa/'.$data->nim) }}' method='post'>
6  @csrf
7  @method('PUT')
8  <div class="my-3 p-3 bg-body rounded shadow-sm">
9      <a href='{{ url('mahasiswa') }}' class="btn btn-secondary">< kembali</a>
10     <div class="mb-3 row">
11         <label for="nim" class="col-sm-2 col-form-label">NIM</label>
12         <div class="col-sm-10">
13             {{ $data->nim }}
14         </div>
15     </div>
16     <div class="mb-3 row">
17         <label for="name" class="col-sm-2 col-form-label">Nama</label>
18         <div class="col-sm-10">
19             <input type="text" class="form-control" name='name' value="{{ $data->name }}" id="name">
20         </div>
21     </div>
22     <div class="mb-3 row">
23         <label for="email" class="col-sm-2 col-form-label">Email</label>
24         <div class="col-sm-10">
25             <input type="text" class="form-control" name='email' value="{{ $data->email }}" id="email">
26         </div>
27     </div>
28     <div class="mb-3 row">
29         <label for="jurusan" class="col-sm-2 col-form-label">Jurusan</label>
30         <div class="col-sm-10">
31             <input type="text" class="form-control" name='jurusan' value="{{ $data->jurusan }}" id="jurusan">
32         </div>
33     </div>
34     <div class="mb-3 row">
35         <label for="age" class="col-sm-2 col-form-label">Umur</label>
36         <div class="col-sm-10">
37             <input type="number" class="form-control" name='age' value="{{ $data->age }}" id="age">
38         </div>
39     </div>
40     <div class="mb-3 row">
41         <label for="address" class="col-sm-2 col-form-label">Alamat</label>
42         <div class="col-sm-10">
43             <input type="text" class="form-control" name='address' value="{{ $data->address }}" id="address">
44         </div>
45     </div>
46     <div class="mb-3 row">
47         <label for="address" class="col-sm-2 col-form-label"></label>
48         <div class="col-sm-10"><button type="submit" class="btn btn-success" name="submit">Simpan Per
49     </div>
50 </div>
51 </form>
52 <!-- AKHIR FORM -->
53 @endsection
54
```

Source code di atas adalah sebuah halaman view dalam framework Laravel yang menggunakan Blade templating engine. Halaman ini digunakan untuk menampilkan formulir edit data mahasiswa yang sudah ada.

Pertama, menggunakan layout template default yang diatur di file layout.template. Di dalamnya terdapat section konten yang digunakan untuk menampilkan konten khusus halaman ini.

Formulir dibuka dengan tag <form> yang mengarah ke URL dengan metode PUT menggunakan metode POST karena HTML hanya mendukung GET dan POST. Dalam formulir ini terdapat beberapa input fields:

- NIM ditampilkan tanpa bisa diubah karena menggunakan <div> untuk menampilkan nilai dari variabel \$data->nim.
- Nama, email, jurusan, umur, dan alamat ditampilkan dalam <input> yang sudah terisi dengan nilai dari variabel \$data yang sesuai.
- Tombol "Simpan Perubahan" yang akan mengirimkan formulir.

Setiap input field memiliki label dan kontrol input yang dibungkus dalam elemen <div> dengan kelas col-sm-10 untuk menyesuaikan tata letak Bootstrap grid. Tombol "Kembali" di bagian atas memberikan link untuk kembali ke halaman sebelumnya.

Kode menggunakan Bootstrap untuk styling dengan class btn, btn-secondary, form-control, btn-success, dan lainnya untuk tata letak dan penampilan formulir yang responsif. Formulir ini digunakan untuk mengubah data mahasiswa dengan metode HTTP PUT pada saat formulir disubmit.

10. Mahasiswa/index.blade.php

```
resources > views > mahasiswa > index.blade.php > div.my-3.p-3.bg-body.rounded.shadow-sm > table.table.table-striped > thead
1  @extends('layout.template')
2  <!-- START DATA -->
3  @section('konten')
4  <div class="my-3 p-3 bg-body rounded shadow-sm">
5
6      <!-- TOMBOL TAMBAH DATA -->
7      <div class="pb-3">
8          <a href="{{ url('mahasiswa/create') }}" class="btn btn-primary">+ Tambah Data</a>
9      </div>
10
11      <table class="table table-striped">
12          <thead>
13              <tr>
14                  <th class="col-md-1">No</th>
15                  <th class="col-md-1">NIM</th>
16                  <th class="col-md-1">Nama</th>
17                  <th class="col-md-1">Email</th>
18                  <th class="col-md-1">Jurusan</th>
```

```

19         <th class="col-md-1">Umur</th>
20         <th class="col-md-2">Alamat</th>
21         <th class="col-md-2">Aksi</th>
22     </tr>
23 </thead>
24 <tbody>
25     <?php $i = $data->firstItem() ?>
26     @foreach ($data as $item)
27     <tr>
28         <td>{{ $i }}</td>
29         <td>{{ $item->nim }}</td>
30         <td>{{ $item->name }}</td>
31         <td>{{ $item->email }}</td>
32         <td>{{ $item->jurusan }}</td>
33         <td>{{ $item->age }}</td>
34         <td>{{ $item->address }}</td>
35         <td>
36             <a href="{{ url('mahasiswa/'.$item->nim.'/edit') }}" class="btn btn-secondary btn-sm">Edit</a>
37             <form onsubmit="return confirm('Yakin akan menghapus data?')" class="d-inline-block">
38                 @csrf
39                 @method('DELETE')
40                 <button type="submit" name="submit" class="btn btn-danger btn-sm">Delete</button>
41             </form>
42         </td>
43     </tr>
44     <?php $i++ ?>
45     @endforeach
46 </tbody>
47 </table>
48 {{ $data->withQueryString()->links() }}
49 </div>
50 <!-- AKHIR DATA -->
51 @endsection
52

```

Source code di atas merupakan sebuah view template dalam framework Laravel atau serupa. Hal ini terlihat dari sintaks-sintaks khas Laravel seperti @extends, @section, {{ }}, dan penggunaan directive @foreach, @csrf, serta @method.

Pada bagian @extends('layout.template'), view ini meng-extend atau mengambil template layout yang sudah ada dengan nama template.blade.php yang terletak di dalam folder layout.

Di dalam section @section('konten'), terdapat struktur HTML yang menampilkan data mahasiswa. Hal ini dimulai dari sebuah div dengan kelas my-3 p-3 bg-body rounded shadow-sm untuk styling, kemudian di dalamnya terdapat tombol tambah data dengan link ke route mahasiswa/create menggunakan tombol berkelas btn btn-primary.

Data mahasiswa ditampilkan dalam sebuah tabel dengan kelas table table-striped. Kolom-kolom yang ditampilkan meliputi NIM, Nama, Email, Jurusan, Umur, Alamat, dan Aksi. Data mahasiswa yang diambil dari variabel \$data di-loop menggunakan @foreach, dan setiap baris data ditampilkan dalam <tr> dengan tombol Edit dan Delete. Tombol Delete dilengkapi dengan form yang melakukan konfirmasi sebelum menghapus data menggunakan JavaScript onsubmit. Paging juga diimplementasikan menggunakan {{ \$data->withQueryString()->links() }} untuk membagi data menjadi halaman-halaman.

Kode ini memanfaatkan fitur-fitur Laravel seperti routing menggunakan url(), penggunaan Blade directive untuk output variabel dan looping, serta fitur-fitur form bawaan Laravel seperti @csrf untuk token CSRF protection dan @method('DELETE') untuk mengirimkan metode HTTP DELETE. Dengan ini, view ini secara efektif menampilkan data mahasiswa dengan fitur CRUD dasar.

H. Routes/web.php

```
routes > web.php
1  <?php
2
3  use Illuminate\Support\Facades\Route;
4  use App\Http\Controllers\UserController;
5  use App\Http\Controllers\MahasiswaController;
6
7  Route::get('/', function () {
8      return view('authentication/home', ['title' => 'Home']);
9  }->name('home');
10
11
12  Route::resource('mahasiswa', MahasiswaController::class);
13
14  Route::get('register', [UserController::class, 'register'])->name('register');
15  Route::post('register', [UserController::class, 'register_action'])->name('register.action');
16  Route::get('login', [UserController::class, 'login'])->name('login');
17  Route::post('login', [UserController::class, 'login_action'])->name('login.action');
18  Route::get('password', [UserController::class, 'password'])->name('password');
19  Route::post('password', [UserController::class, 'password_action'])->name('password.action');
20  Route::get('logout', [UserController::class, 'logout'])->name('logout');
21
```

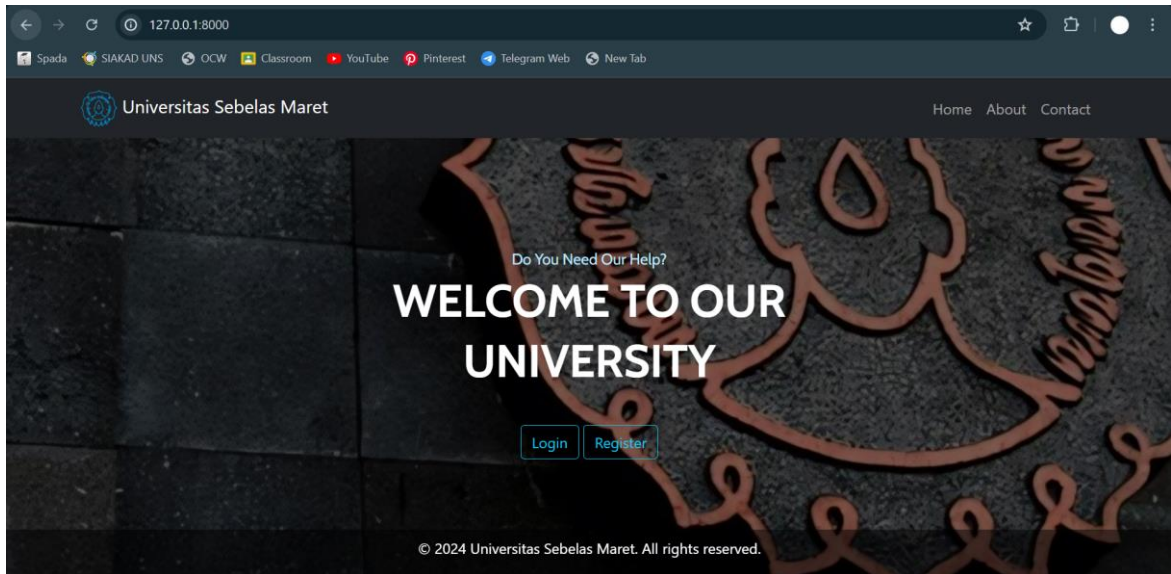
Source code di atas merupakan bagian dari konfigurasi routing dalam aplikasi Laravel. Pada bagian ini, beberapa route ditentukan untuk mengarahkan permintaan HTTP ke controller dan metode yang sesuai. Route pertama adalah route untuk URL root ("/") yang mengarahkan ke view 'authentication/home' dengan variabel 'title' yang di-set ke 'Home'. Route ini juga diberi nama 'home'.

Selanjutnya, ada route resource untuk 'mahasiswa' yang akan mengarahkan ke semua metode CRUD dalam MahasiswaController. Route ini mengkonfigurasi semua route standar (index, create, store, show, edit, update, destroy) untuk resource 'mahasiswa'.

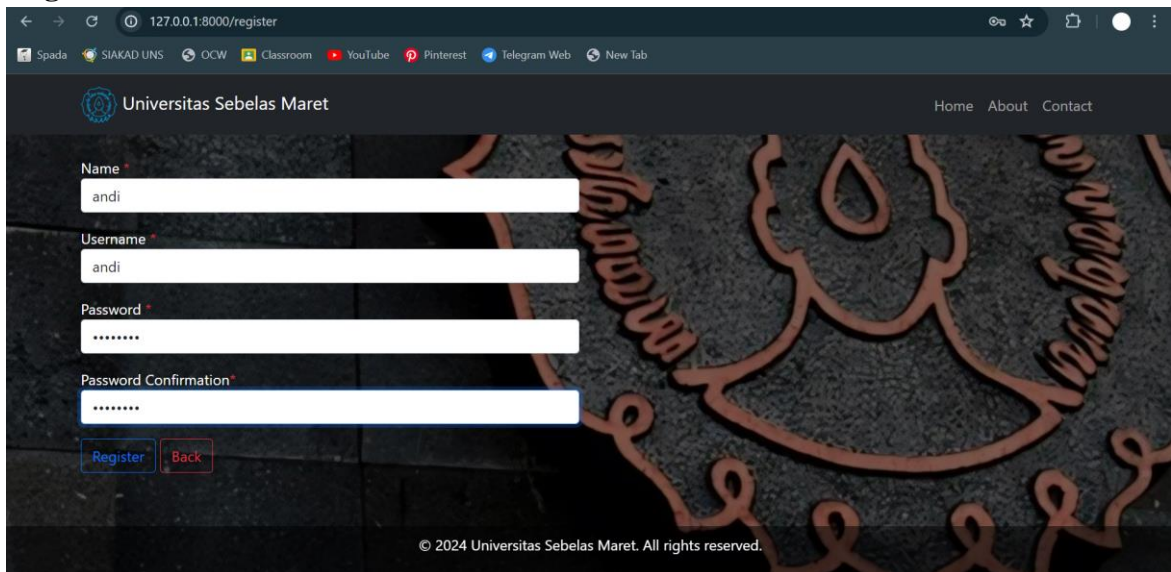
Selain itu, terdapat beberapa route yang terkait dengan proses otentikasi pengguna. Route 'register' mengarahkan ke metode 'register' dan 'register_action' dalam UserController untuk menampilkan formulir registrasi dan memproses data registrasi. Route 'login' mengarahkan ke metode 'login' dan 'login_action' untuk menampilkan formulir login dan memproses data login. Route 'password' mengarahkan ke metode 'password' dan 'password_action' untuk menampilkan formulir pengaturan ulang kata sandi dan memproses data pengaturan ulang kata sandi. Terakhir, route 'logout' mengarahkan ke metode 'logout' dalam UserController untuk memproses logout pengguna.

2. Tampilan user interface

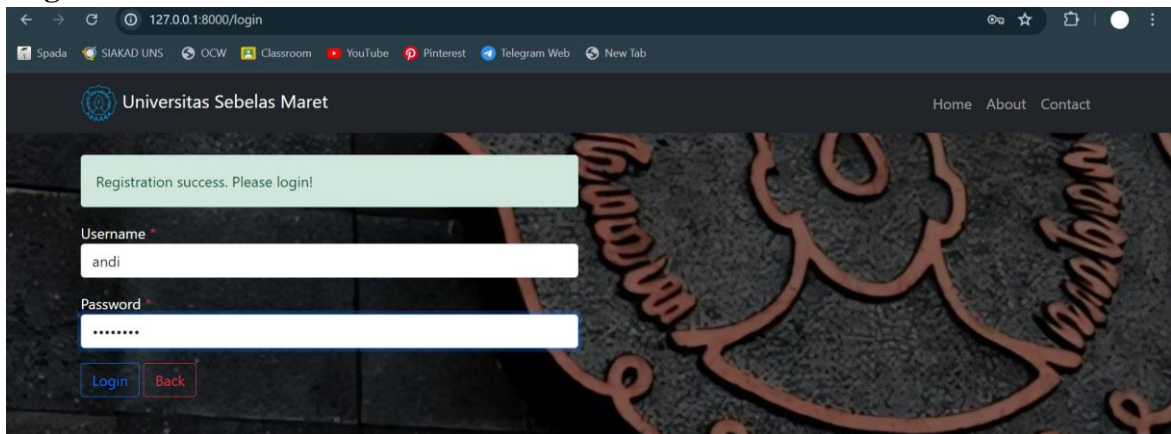
a. Homepage



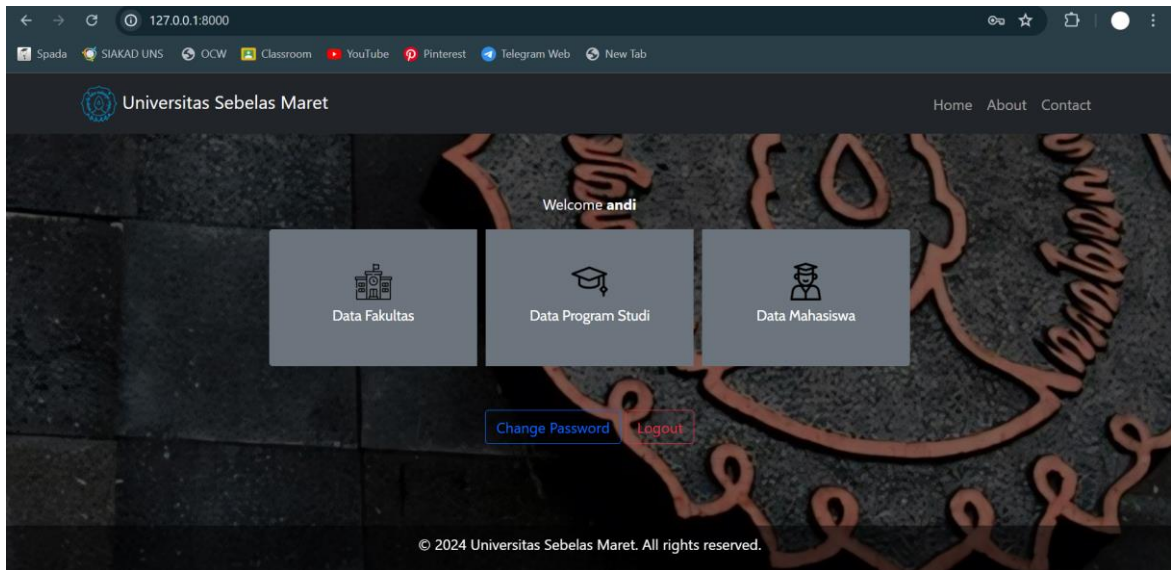
b. Register



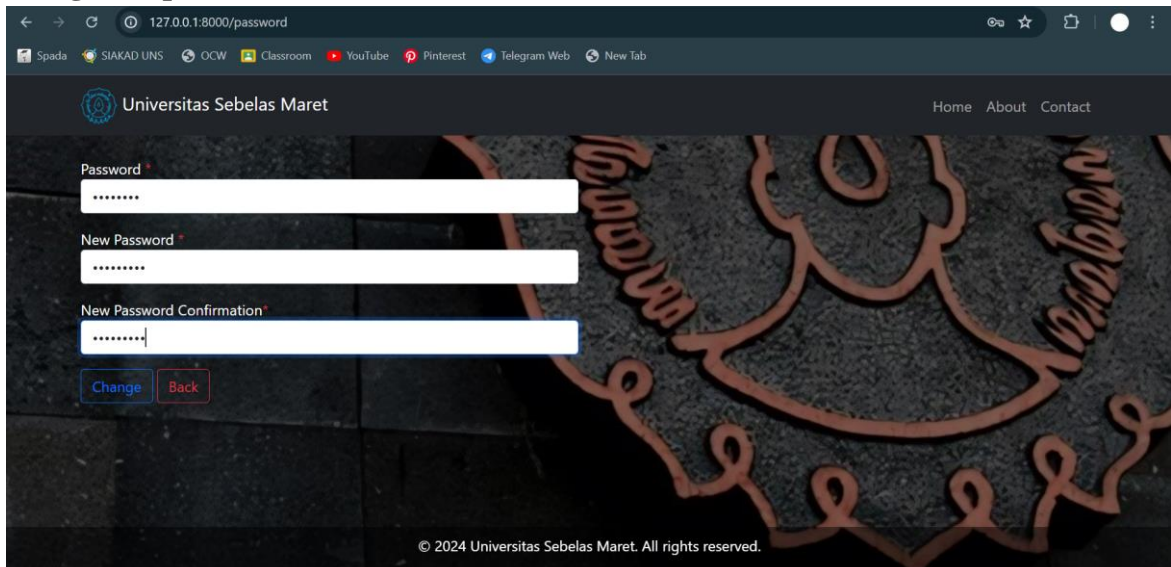
c. Login



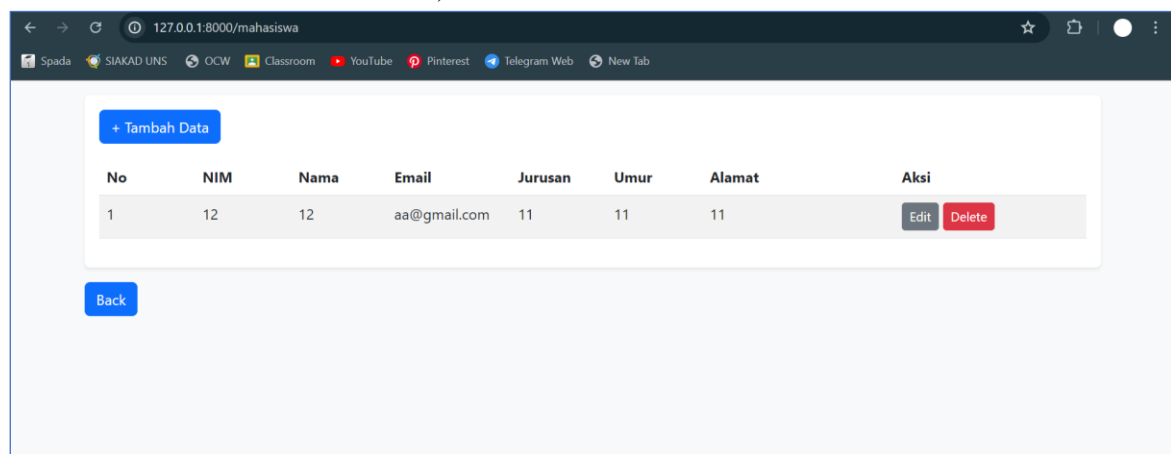
d. Dashboard



e. Mengubah password



f. Tampilan Data mahasiswa (jika opsi data mahasiswa ditekan pada dashboard, Tombol back untuk kembali ke dashboard)



g. Create data

Before

No	NIM	Nama	Email	Jurusan	Umur	Alamat	Aksi
1	12	12	aa@gmail.com	11	11	11	<button>Edit</button> <button>Delete</button>

Create data

<< kembali

NIM: 123456

Nama: andi

Email: andi@gmail.com

Jurusan: Informatika

Umur: 20

Alamat: Rumah

Simpan abc

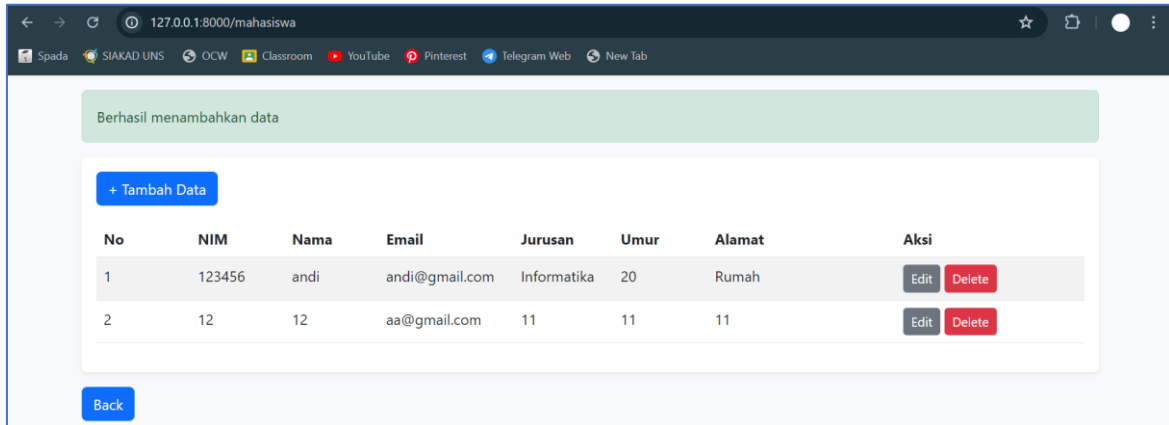
After

Berhasil menambahkan data

No	NIM	Nama	Email	Jurusan	Umur	Alamat	Aksi
1	123456	andi	andi@gmail.com	Informatika	20	Rumah	<button>Edit</button> <button>Delete</button>
2	12	12	aa@gmail.com	11	11	11	<button>Edit</button> <button>Delete</button>

h. Edit data

Before



127.0.0.1:8000/mahasiswa

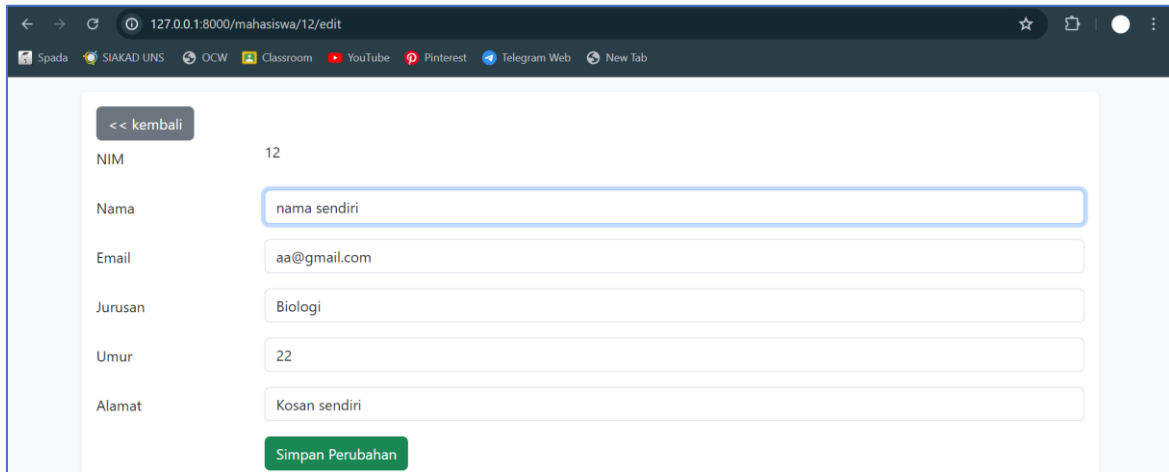
Berhasil menambahkan data

+ Tambah Data

No	NIM	Nama	Email	Jurusan	Umur	Alamat	Aksi
1	123456	andi	andi@gmail.com	Informatika	20	Rumah	Edit Delete
2	12	12	aa@gmail.com	11	11	11	Edit Delete

Back

Edit data



127.0.0.1:8000/mahasiswa/12/edit

<< kembali

NIM 12

Nama nama sendiri

Email aa@gmail.com

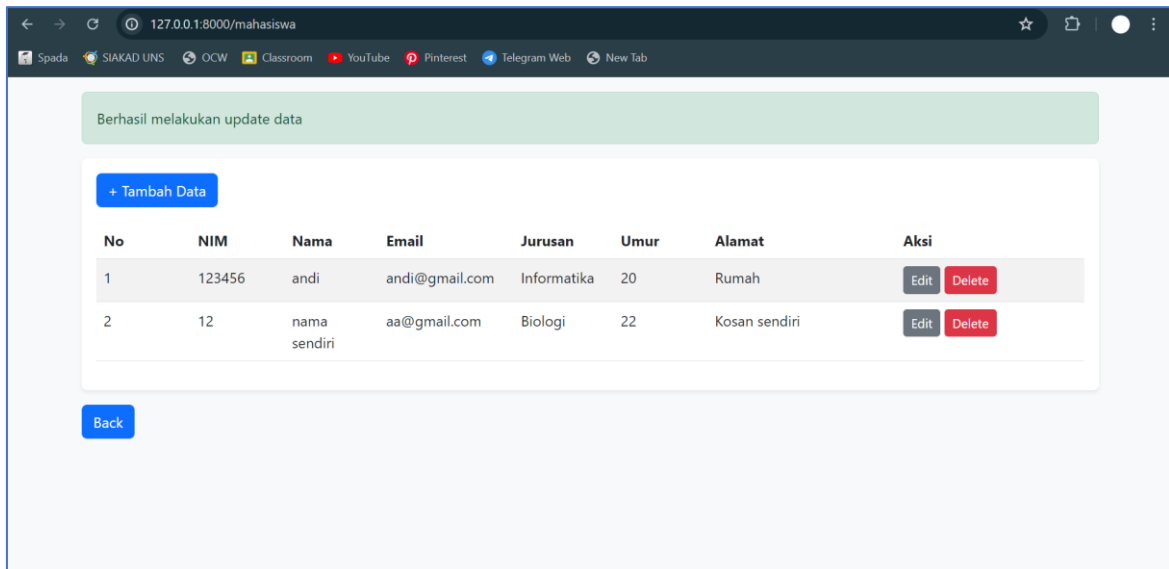
Jurusan Biologi

Umur 22

Alamat Kosan sendiri

Simpan Perubahan

After



127.0.0.1:8000/mahasiswa

Berhasil melakukan update data

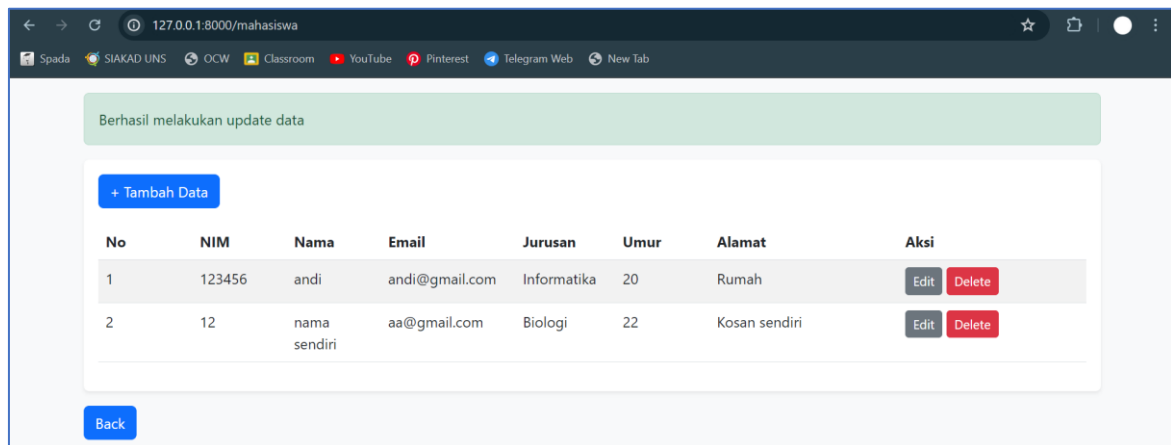
+ Tambah Data

No	NIM	Nama	Email	Jurusan	Umur	Alamat	Aksi
1	123456	andi	andi@gmail.com	Informatika	20	Rumah	Edit Delete
2	12	nama sendiri	aa@gmail.com	Biologi	22	Kosan sendiri	Edit Delete

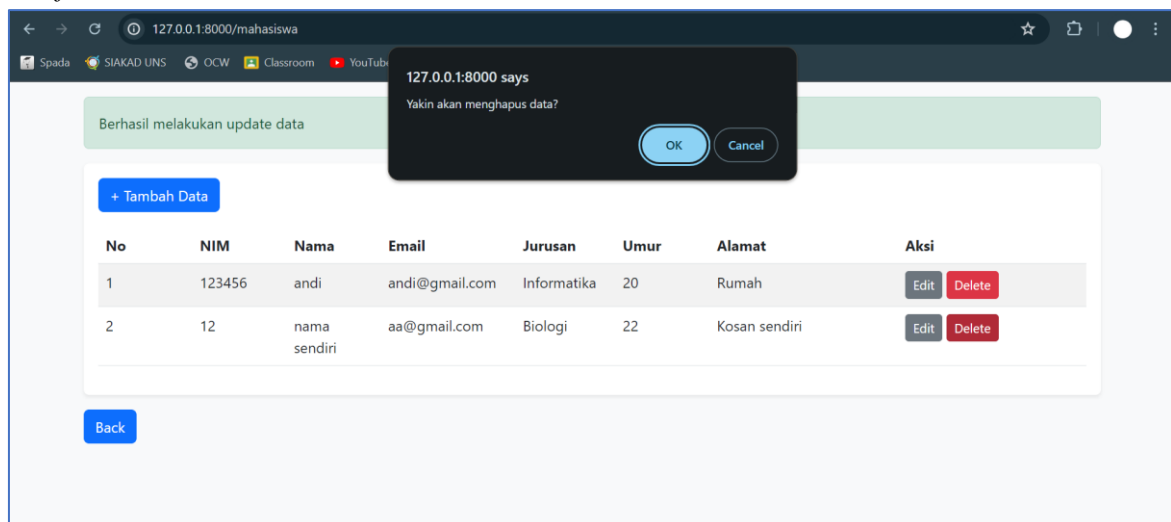
Back

i. Delete data

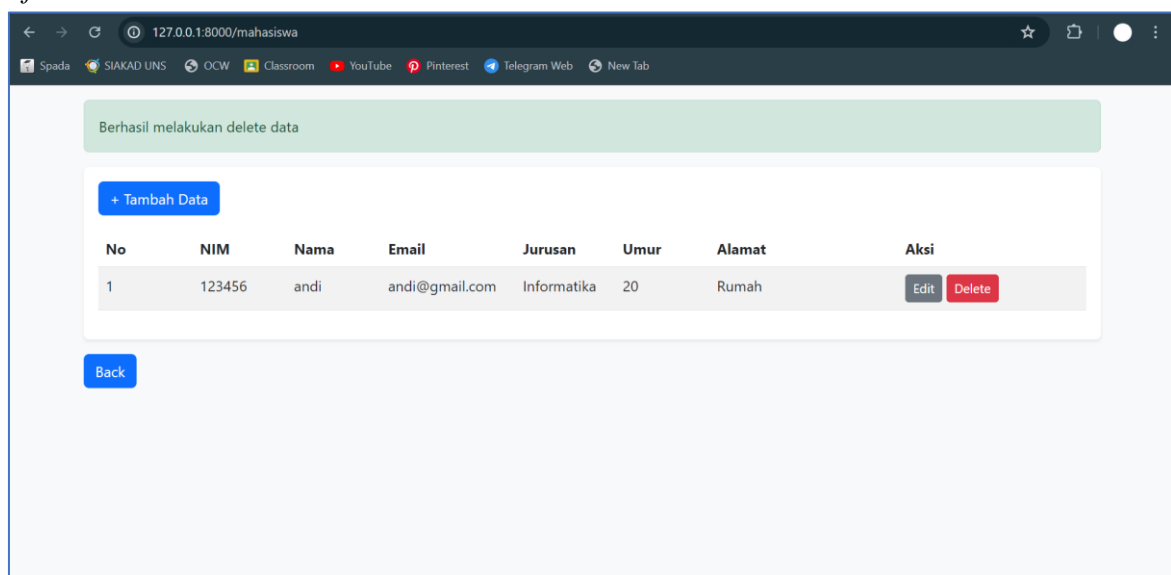
Before



Konfirmasi



After



3. Kesimpulan

Framework Laravel mengadopsi arsitektur MVC (Model-View-Controller) yang memisahkan logika aplikasi ke dalam tiga komponen utama: Model, View, dan Controller. Dalam contoh yang diberikan, Model seperti Mahasiswa dan User merepresentasikan entitas data dalam aplikasi dan berinteraksi dengan basis data menggunakan Eloquent ORM. Model ini mendefinisikan properti-properti yang bisa diisi secara massal dan menggunakan fitur seperti HasFactory untuk mempermudah pembuatan data uji. View dalam Laravel, seperti berbagai file blade.php, bertanggung jawab untuk menampilkan antarmuka pengguna. View ini menggunakan Blade templating engine untuk mempermudah penyusunan HTML dinamis dengan directive khusus seperti @extends, @section, @foreach, dan lain-lain. Controller, seperti MahasiswaController dan UserController, menangani logika aplikasi dan merespons permintaan HTTP. Controller ini mengelola operasi CRUD (Create, Read, Update, Delete) dengan metode seperti index, create, store, edit, update, dan destroy.

Migration dalam Laravel digunakan untuk mengelola skema basis data. Contohnya, file Migrations/tb_user dan Migrations/mahasiswas berfungsi untuk membuat tabel tb_user dan mahasiswa dengan kolom-kolom yang diperlukan. Metode up dalam migration mendefinisikan struktur tabel yang akan dibuat, sedangkan metode down mendefinisikan cara untuk menghapus tabel tersebut jika migrasi dibatalkan.

Authentication dalam Laravel dikelola melalui controller seperti UserController yang mencakup fungsi registrasi, login, logout, dan penggantian password. Proses ini melibatkan validasi input, hashing password sebelum menyimpannya ke database, dan penggunaan session untuk mengelola status pengguna yang sedang login.

Operasi CRUD diimplementasikan dalam MahasiswaController yang menyediakan metode untuk menampilkan daftar mahasiswa (index), menambahkan data mahasiswa baru (create dan store), mengedit data mahasiswa (edit dan update), serta menghapus data mahasiswa (destroy). Dengan kombinasi routing yang didefinisikan dalam web.php, Laravel memudahkan pengaturan alur aplikasi mulai dari menerima permintaan HTTP hingga memberikan respons yang sesuai.