Nama: Abraham Willem Hersubagyo
NIM: L0122002
Kelas: A

Disclaimer: Tema VSCode saya mungkin akan berbeda dari yang awal dengan yang belakangan karena saya menggunakan laptop teman saya

LAPORAN RESPONSI PEMWEB

1. MahasisaModel

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Mahasiswa extends Model
{
    use HasFactory;

    protected $table = 'mahasiswas';

    protected $fillable = [
        'nama', 'nim', 'ipk',
    ];
}
```

Code di atas merupakan mahasiswa model yang memiliki atribut nama, nim dan ipk.

2. UserModel

```php
<?php

namespace App\Models;

// use Illuminate\Contracts\Auth\MustVerifyEmail;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Foundation\Auth\User as Authenticatable;
use Illuminate\Notifications\Notifiable;

class User extends Authenticatable
{
    use HasFactory, Notifiable;

    /**
```

```
     * The attributes that are mass assignable.
     *
     * @var array<int, string>
     */
    protected $fillable = [
        'name',
        'email',
        'password',
        'role'
    ];

    /**
     * The attributes that should be hidden for serialization.
     *
     * @var array<int, string>
     */
    protected $hidden = [
        'password',
        'remember_token',
    ];

    /**
     * Get the attributes that should be cast.
     *
     * @return array<string, string>
     */
    protected function casts(): array
    {
        return [
            'email_verified_at' => 'datetime',
            'password' => 'hashed',
        ];
    }
}
```

Code di atas merupakan user model dengan credentials email dan password yang dapat digunakan untuk login ke akun dan membuka page mahasiswa.

3. **AuthController**

```
<?php

namespace App\Http\Controllers;
```

```php
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Hash;
use App\Models\User;

class AuthController extends Controller
{
    public function showLoginForm()
    {
        return view('auth.login');
    }

    public function login(Request $request)
    {
        $credentials = $request->only('email', 'password');

        if (Auth::attempt($credentials)) {
            return redirect()->intended('/');
        }

        return back()->withErrors([
            'email' => 'The provided credentials do not match our records.',
        ]);
    }

    public function showRegisterForm()
    {
        return view('auth.register');
    }

    public function register(Request $request)
    {
        $request->validate([
            'name' => 'required|string|max:255',
            'email' => 'required|string|email|max:255|unique:users',
            'password' => 'required|string|min:8|confirmed',
        ]);

        User::create([
            'name' => $request->name,
            'email' => $request->email,
```

```
            'password' => Hash::make($request->password),
        ]);

        return redirect('/login');
    }

    public function logout()
    {
        Auth::logout();
        return redirect('/login');
    }
}
```

Code di atas adalah AuthController yang digunakan untuk otentikasi user sebelum masuk ke dalam page mahasiswa.

4. **MahasiswaController**

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Models\Mahasiswa;

class MahasiswaController extends Controller
{
    public function index()
    {
        $mahasiswas = Mahasiswa::all();
        return view('index', ['mahasiswas' => $mahasiswas]);
    }

    public function store(Request $request)
    {
        $mahasiswa = new Mahasiswa();
        $mahasiswa->nama = $request->input('nama');
        $mahasiswa->nim = $request->input('nim');
        $mahasiswa->ipk = $request->input('ipk');
        $mahasiswa->save();

        return redirect('/');
    }
```

```
    public function show(Mahasiswa $mahasiswa)
    {
        return $mahasiswa;
    }

    public function edit(Request $request, string $id)
    {
        $mahasiswa = Mahasiswa::find($id);
        return view('/edit', compact('mahasiswa'));
    }

    public function update(Request $request, string $id)
    {
        $mahasiswa = $request->all;
        $mahasiswa['nama'] = $request->nama;
        $mahasiswa['nim'] = $request->nim;
        $mahasiswa['ipk'] = $request->ipk;

        Mahasiswa::whereId($id)->update($mahasiswa);

        return redirect('/');
    }

    public function destroy(Request $request)
{
    $mahasiswa = Mahasiswa::findOrFail($request->id);
    $mahasiswa->delete();

    return redirect('/')->with('success', 'Mahasiswa berhasil
dihapus');
}
}
```

Code di atas merupakan mahasiswacontroller yang digunakan untuk membuat data mahasiswa baru, mengedit, dan menghapus data mahasiswa.

5. **Views index.blade.php**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Tabel Mahasiswa</title>
```

```
  <script src="https://cdn.tailwindcss.com"></script>
</head>
<body>
  <div class="container">
    <div class="flex flex-col justify-center text-center
items-center">
      <h2 class="font-bold text-2xl pt-5">Data Mahasiswa S1
Informatika</h2>
    </div>
    {{-- TABLE --}}
    <div class="flex flex-col mt-10" data-aos="fade-down">
      <div class="overflow-x-auto sm:mx-0.5 lg:mx-0.5">
        <div class="py-2 inline-block min-w-full px-24">
          <div class="overflow-hidden">
            <table class="min-w-full">
              <thead class="bg-white border-b border-slate-500">
                <tr>
                  <th scope="col" class="text-sm font-medium
text-gray-900 px-6 py-4 text-center">
                    ID
                  </th>
                  <th scope="col" class="text-sm font-medium
text-gray-900 px-6 py-4 text-center">
                    Name
                  </th>
                  <th scope="col" class="text-sm font-medium
text-gray-900 px-6 py-4 text-center">
                    NIM
                  </th>
                  <th scope="col" class="text-sm font-medium
text-gray-900 px-6 py-4 text-center">
                    IPK
                  </th>
                  <th scope="col" class="text-sm font-medium
text-gray-900 px-6 py-4 text-center">
                    Action
                  </th>
                </tr>
              </thead>
              <tbody>
                @foreach($mahasiswas as $mahasiswa)
                <tr class="border-b border-slate-300">
```

```
                <td class="px-6 py-4 whitespace-nowrap text-sm
font-medium text-gray-900 text-center">{{ $mahasiswa->id }}</td>
                <td class="text-sm text-gray-900 font-light
px-6 py-4 whitespace-nowrap text-center">
                  {{ $mahasiswa->nama }}
                </td>
                <td class="text-sm text-gray-900 font-light
px-6 py-4 whitespace-nowrap text-center">
                  {{ $mahasiswa->nim }}
                </td>
                <td class="text-sm text-gray-900 font-light
px-6 py-4 whitespace-nowrap text-center">
                  {{ $mahasiswa->ipk }}
                </td>
                <td class="flex flex-row text-center
items-center justify-center text-sm text-gray-900 font-light px-6
py-4 whitespace-nowrap text-center">
                  <a href="{{ route('mahasiswa.edit', ['id' =>
$mahasiswa->id]) }}" class="bg-blue-500 px-3 py-2 text-white
rounded mx-2">Edit</a>
                  <form action="/" method="POST"
onsubmit="return confirm('Apakah Anda yakin ingin menghapus
mahasiswa ini?')">
                    @csrf
                    @method('DELETE')
                    <input type="hidden" name="id" value="{{
$mahasiswa->id }}">
                    <button type="submit" class="bg-red-500
px-3 py-2 text-white rounded">Delete</button>
                  </form>
                </td>
              </tr>
              @endforeach
            </tbody>
          </table>
        </div>
      </div>
    </div>
    <div class="flex flex-row justify-center items-center
text-center">
      <div class="w-[1000px] min-h-0.5 bg-slate-400
mt-24"></div>
```

```
    </div>
    {{-- FORM ADD --}}
    <div class="flex flex-col justify-center items-center
text-center">
      <h3 class="font-bold text-md">Tambahkan Data Mahasiswa</h3>
      <form action="/" method="POST" class="mt-2">
        @csrf
        <div class="flex flex-col">
          <label for="nama" class="text-sm font-medium
text-gray-900">Nama:</label>
          <input type="text" id="nama" name="nama" class="border
border-gray-300 px-2 py-1 rounded-sm mt-1">
        </div>
        <div class="flex flex-col mt-2">
          <label for="nim" class="text-sm font-medium
text-gray-900">NIM:</label>
          <input type="text" id="nim" name="nim" class="border
border-gray-300 px-2 py-1 rounded-sm mt-1">
        </div>
        <div class="flex flex-col mt-2">
          <label for="ipk" class="text-sm font-medium
text-gray-900">IPK:</label>
          <input type="text" id="ipk" name="ipk" class="border
border-gray-300 px-2 py-1 rounded-sm mt-1">
        </div>
        <button type="submit" class="bg-blue-500 text-white
font-bold px-4 py-2 rounded mt-3">Tambahkan</button>
      </form>
    </div>
  </div>
</body>
</html>
```

Code di atas merupakan code untuk page utama (index.blade.php) yang digunakan untuk menambah data mahasiswa, dan menghapusnya, serta tombol untuk masuk ke dalam page edit untuk mengedit data mahasiswa.

6. **View edit.blade.php**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
```

```
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Tabel Mahasiswa</title>
  <script src="https://cdn.tailwindcss.com"></script>
</head>
<body>
  <div class="container">
    <div class="flex flex-col justify-center text-center
items-center">
        <h2 class="font-bold text-2xl pt-5">Edit Mahasiswa</h2>
    </div>
    {{-- FORM ADD --}}
    <div class="flex flex-col justify-center items-center
text-center py-10 mt-24">
        <form action="{{route ('mahasiswa.update', ['id' =>
$mahasiswa->id])}}" method="POST" class="mt-2">
            @csrf
            @method('PUT')
            <div class="flex flex-col">
                <label for="nama" class="text-sm font-medium
text-gray-900">Nama:</label>
                <input type="text" id="nama" name="nama"
value="{{$mahasiswa->nama}}" class="border border-gray-300 px-2
py-1 rounded-sm mt-1">
            </div>
            <div class="flex flex-col mt-2">
                <label for="nim" class="text-sm font-medium
text-gray-900">NIM:</label>
                <input type="text" id="nim" name="nim"
value="{{$mahasiswa->nim}}" class="border border-gray-300 px-2
py-1 rounded-sm mt-1">
            </div>
            <div class="flex flex-col mt-2">
                <label for="ipk" class="text-sm font-medium
text-gray-900">IPK:</label>
                <input type="text" id="ipk" name="ipk"
value="{{$mahasiswa->ipk}}" class="border border-gray-300 px-2
py-1 rounded-sm mt-1">
            </div>
            <button type="submit" class="bg-blue-500 text-white
font-bold px-4 py-2 rounded mt-3">Tambahkan</button>
        </form>
    </div>
```

```
  </div>
</body>
</html>
```

Page edit data mahasiswa sesuai dengan id mahasiswa yang dipiilih

7. **Views login page**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Login</title>
  <script src="https://cdn.tailwindcss.com"></script>
</head>
<body>
  <div class="container">
    <div class="flex flex-col justify-center text-center
items-center min-h-screen">
      <h2 class="font-bold text-2xl pt-5">Login</h2>
      <div class="flex flex-col justify-center items-center
text-center mt-10">
        <form action="{{ route('login') }}" method="POST"
class="bg-white shadow-md rounded px-8 pt-6 pb-8 mb-4">
          @csrf
          <div class="flex flex-col mb-4">
            <label for="email" class="text-sm font-medium
text-gray-900 mb-2">Email:</label>
            <input type="email" id="email" name="email"
class="border border-gray-300 px-4 py-2 rounded-sm" required>
          </div>
          <div class="flex flex-col mb-6">
            <label for="password" class="text-sm font-medium
text-gray-900 mb-2">Password:</label>
            <input type="password" id="password" name="password"
class="border border-gray-300 px-4 py-2 rounded-sm" required>
          </div>
          <button type="submit" class="bg-blue-500 text-white
font-bold px-4 py-2 rounded">Login</button>
        </form>
        <a href="/register" class="text-blue-500
hover:text-blue-700">Register</a>
      </div>
```

```
    </div>
  </div>
</body>
</html>
```

Code di atas merupakan views bagian login yang digunakan untuk login user.

8. **register.blade.php**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Register</title>
  <script src="https://cdn.tailwindcss.com"></script>
</head>
<body>
  <div class="container">
    <div class="flex flex-col justify-center text-center
items-center min-h-screen">
      <h2 class="font-bold text-2xl pt-5">Register</h2>
      <div class="flex flex-col justify-center items-center
text-center mt-10">
        <form action="/register" method="POST" class="bg-white
shadow-md rounded px-8 pt-6 pb-8 mb-4">
          @csrf
          <div class="flex flex-col mb-4">
            <label for="name" class="text-sm font-medium
text-gray-900 mb-2">Name:</label>
            <input type="text" id="name" name="name"
class="border border-gray-300 px-4 py-2 rounded-sm" required>
          </div>
          <div class="flex flex-col mb-4">
            <label for="email" class="text-sm font-medium
text-gray-900 mb-2">Email:</label>
            <input type="email" id="email" name="email"
class="border border-gray-300 px-4 py-2 rounded-sm" required>
          </div>
          <div class="flex flex-col mb-4">
            <label for="password" class="text-sm font-medium
text-gray-900 mb-2">Password:</label>
```

```
            <input type="password" id="password" name="password"
class="border border-gray-300 px-4 py-2 rounded-sm" required>
          </div>
          <div class="flex flex-col mb-6">
            <label for="password_confirmation" class="text-sm
font-medium text-gray-900 mb-2">Confirm Password:</label>
            <input type="password" id="password_confirmation"
name="password_confirmation" class="border border-gray-300 px-4
py-2 rounded-sm" required>
          </div>
          <button type="submit" class="bg-blue-500 text-white
font-bold px-4 py-2 rounded">Register</button>
        </form>
        <a href="/login" class="text-blue-500
hover:text-blue-700">Login</a>
      </div>
    </div>
  </div>
</body>
</html>
```

Code di atas adalah page register yang dapat digunakan user untuk membuat user baru supaya dapat masuk ke page mahasiswa.

9. **Routes**

```
<?php

use Illuminate\Support\Facades\Route;
use App\Http\Controllers\MahasiswaController;
use App\Http\Controllers\AuthController;

#index
Route::get('/', [MahasiswaController::class,
'index'])->middleware('auth');
Route::post('/', [MahasiswaController::class,
'store'])->middleware('auth');
Route::delete('/', [MahasiswaController::class,
'destroy'])->middleware('auth');

#auth
Route::get('/login', [AuthController::class,
'showLoginForm'])->name('login');
Route::post('/login', [AuthController::class, 'login']);
```

```php
Route::get('/register', [AuthController::class,
'showRegisterForm']);
Route::post('/register', [AuthController::class, 'register']);
Route::post('/logout', [AuthController::class,
'logout'])->name('logout');

#mahasiswa
Route::get('/edit/{id}', [MahasiswaController::class,
'edit'])->middleware('auth')->name('mahasiswa.edit');
Route::put('/update/{id}', [MahasiswaController::class,
'update'])->middleware('auth')->name('mahasiswa.update');
Route::get('/{mahasiswa}', [MahasiswaController::class,
'show'])->middleware('auth');
```

Routes yang digunakan untuk mengatur ke mana page akan dialihkan dan controller apa yang akan digunakan berdasarkan aktivitas pengguna

10. Migration

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateMahasiswasTable extends Migration
{
    public function up()
    {
        Schema::create('mahasiswas', function (Blueprint $table)
{
            $table->id();
            $table->string('nama');
            $table->string('nim');
            $table->float('ipk');
            $table->timestamps();
        });
    }

    public function down()
    {
        Schema::dropIfExists('mahasiswas');
    }
}
```

```
```

Code di atas adalah bagian migration yang berfungsi untuk membuat schema database dengan isi atribut nama (string), nim (string), ipk (float), dan timestamp.

11. **Seeder**

```php
<?php

namespace Database\Seeders;

use App\Models\User;
use Illuminate\Database\Seeder;
use Illuminate\Database\Console\Seeds\WithoutModelEvents;

class UserSeeder extends Seeder
{
    /**
     * Run the database seeds.
     */
    public function run(): void
    {
        User::create([
            'name' => 'admin1',
            'email' => 'admin@example.com',
            'password' => bcrypt('12345678'),
        ]);
    }
}
```

Code di atas adalah UserSeeder yang nantinya daapat digunakan untuk login. User yang di atas adalah admin dengan name admin1, email admin@example.com, serta password 12345678 yang dihash dengan bcrypt.

12. **Factory**

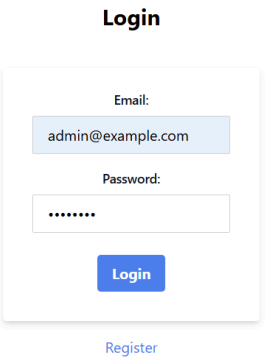
```php
<?php

namespace Database\Factories;

use Illuminate\Database\Eloquent\Factories\Factory;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Str;

/**
 * @extends
\Illuminate\Database\Eloquent\Factories\Factory<\App\Models\User>
```

```php
 */
class UserFactory extends Factory
{
    /**
     * The current password being used by the factory.
     */
    protected static ?string $password;

    /**
     * Define the model's default state.
     *
     * @return array<string, mixed>
     */
    public function definition(): array
    {
        return [
            'name' => fake()->name(),
            'email' => fake()->unique()->safeEmail(),
            'email_verified_at' => now(),
            'password' => static::$password ??=
Hash::make('password'),
            'remember_token' => Str::random(10),
        ];
    }

    /**
     * Indicate that the model's email address should be
unverified.
     */
    public function unverified(): static
    {
        return $this->state(fn (array $attributes) => [
            'email_verified_at' => null,
        ]);
    }
}
```

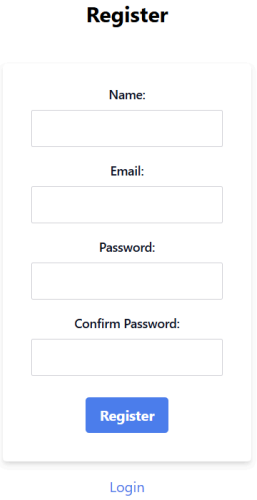
Code di atas adalah UserFactory yang berfungsi untuk membuat data dummy untuk user dengan random. Nantinya factory ini dapat digunakan untuk membuat contoh user dengan name, email, beserta passwordnya.

OUTPUT

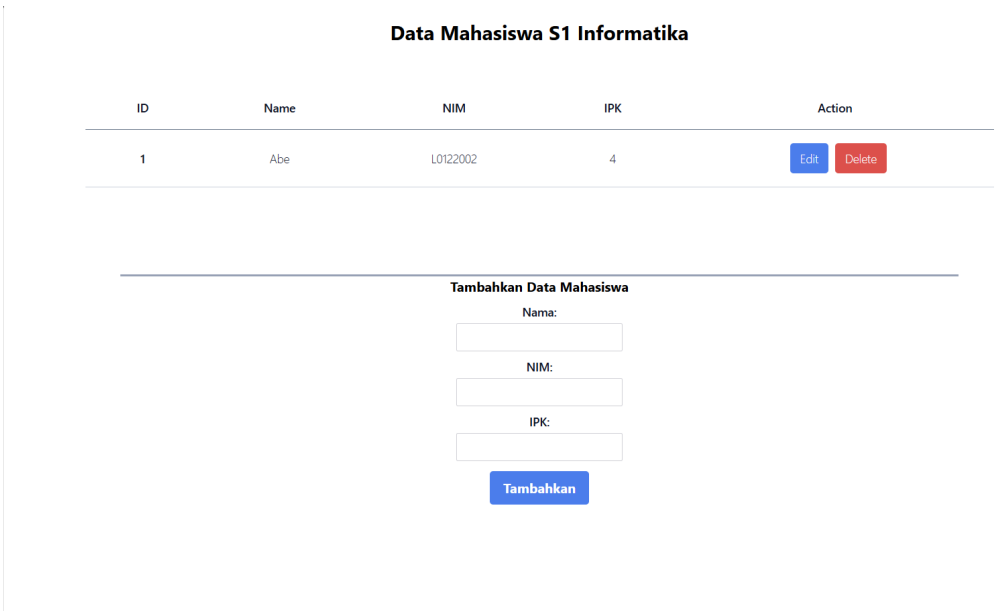
1. **Login**



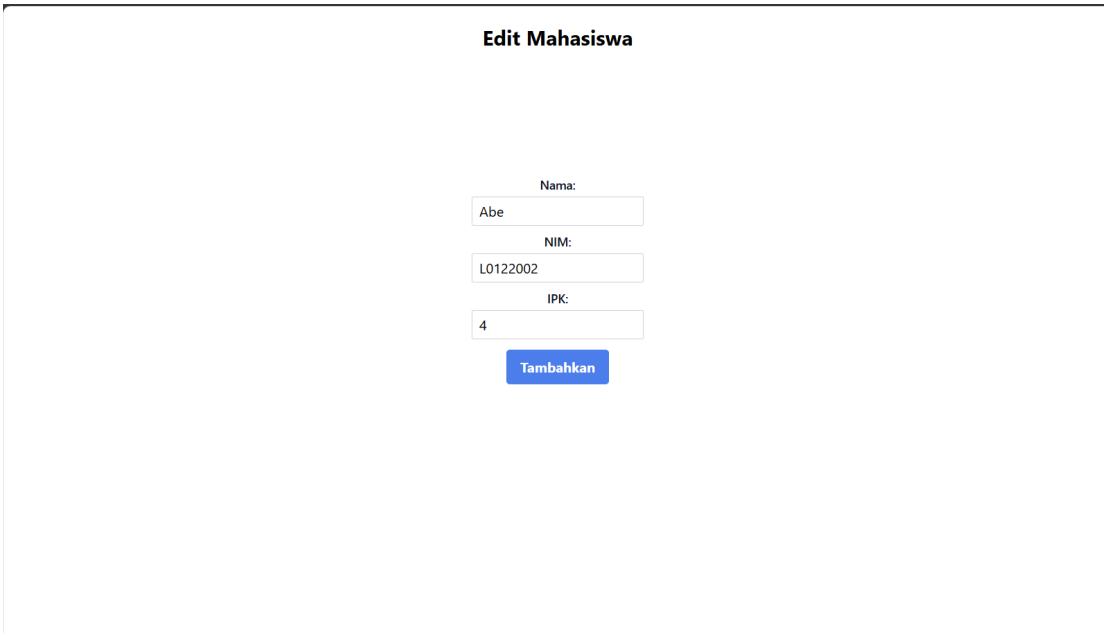
2. **Register**



3. **Index**


Data Mahasiswa S1 Informatika
ID
Name
NIM
IPK
Action
1
Abe
L0122002
4
Edit
Delete
Tambahkan Data Mahasiswa
Nama:
NIM:
IPK:
Tambahkan

4. Edit


Edit Mahasiswa
Nama:
Abe
NIM:
L0122002
IPK:
4
Tambahkan