

# Laporan Modul 5: Form Submission & Data Validation

---

**Mata Kuliah:** Workshop Web Lanjut

**Nama:** Bunga Alfa Zahrah **NIM:** 2024573010023 **Kelas:** TI-2C

---

## Abstrak

- Pada praktikum ini, dipelajari bagaimana cara membuat dan mengelola form (form submission) di framework Laravel 12, termasuk cara memproses input pengguna, melakukan validasi data, menampilkan pesan kesalahan (error message), serta mengimplementasikan multi-step form menggunakan session. Melalui praktikum ini, mahasiswa diharapkan dapat memahami alur request-response pada Laravel dan mampu membangun form dinamis yang aman dan interaktif.
- 

## 1. Dasar Teori

- Form submission merupakan proses pengiriman data dari pengguna melalui antarmuka web ke server untuk diproses. Dalam Laravel, pengelolaan form menjadi lebih mudah melalui fitur seperti:
- Routing – Mengarahkan request dari browser ke controller.
- Controller – Menangani logika bisnis, memproses data input, dan mengembalikan response ke view.
- Blade Template – Digunakan untuk membangun tampilan form dan menampilkan pesan validasi.
- Validation – Fitur bawaan Laravel untuk memeriksa keabsahan data yang dikirim pengguna sebelum disimpan atau diproses lebih lanjut.
- Session – Menyimpan data sementara antar halaman, terutama digunakan dalam multi-step form.

Dengan kombinasi konsep-konsep tersebut, Laravel memudahkan pembuatan form yang aman, rapi, dan terstruktur.

---

## 2. Langkah-Langkah Praktikum

### 2.1 Praktikum 1 – Menangani Request dan Response View di Laravel 12

- Buat proyek laravel dengan perintah `Laravel new form-app`, kemudian masuk ke proyek dengan perintah `cd form-app`.
- Tambahkan route pada `routes/web.php`. Editkan di `routes/web.php` menjadi seperti pada gambar berikut.

```

<?php

use Illuminate\Support\Facades\Route;
use App\Http\Controllers\FormController;
use App\Http\Controllers\RegisterController;

Route::get('/register', [RegisterController::class, 'showForm'])->name('register.show');
Route::post('/register', [RegisterController::class, 'handleForm'])->name('register.handle');

Route::get('/form', [FormController::class, 'showForm'])->name('form.show');
Route::post('/form', [FormController::class, 'handleForm'])->name('form.handle');
Route::get('/result', [FormController::class, 'showResult'])->name('form.result');
Tabnine | Edit | Test | Explain | Document
Route::get('/', function () {
    return view('welcome');
});

```

- Buat Controller untuk menangani logika form, dengan perintah php artisan make:controller FormController. Ini akan membuat file controller baru di app/Http/Controllers/FormController.php. Buka file ini dan tambahkan metode berikut :

```

<?php

namespace App\Http\Controllers;
use Illuminate\Http\Request;

class FormController extends Controller{
    Tabnine | Edit | Test | Explain | Document
    public function showForm()
    {
        return view('form');
    }
    Tabnine | Edit | Test | Explain | Document
    public function handleForm(Request $request){
        $validated = $request->validate([
            'name' => 'required|string|max:255',
            'email' => 'required|email',
            'age' => 'required|integer|min:1',
            'password' => 'required|min:6',
            'gender' => 'required',
            'role' => 'required',
            'bio' => 'required',
            'confirm' => 'accepted',
        ]);
        return redirect()->route('form.result')->with('data', $validated);
    }
    Tabnine | Edit | Test | Explain | Document
    public function showResult(){
        $data = session('data');
        return view('result', compact('data'));
    }
}

```

- Buat Metode untuk menghandle data pada Controller Buka `app/Http/Controllers/DasarBladeController.php` dan tambahkan metode berikut:

```
<?php

namespace App\Http\Controllers;
use Illuminate\Http\Request;
class DasarBladeController extends Controller
{
    Tabnine | Edit | Test | Explain | Document
    public function showData(){
        $name = 'bunga';
        $fruits = ['Apple', 'Banana', 'Cherry'];
        $user = [
            'name' => 'bunga',
            'email' => 'bunga@pnl.ac.id',
            'is_active' => true,
        ];
        $product = (object) [
            'id' => 1,
            'name' => 'Laptop',
            'price' => 12000000
        ];
        return view('dasar', compact('name', 'fruits', 'user', 'product'));
    }
}
```

- Buat View Form Buat file view Blade baru untuk form. Buat direktori bernama form di dalam `resources/views` dan buat file bernama `form.blade.php` di dalamnya.

```

<!DOCTYPE html>
<html>
<head>
<title>Pengiriman Form</title>
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
</head>
<body class="container mt-5">
<h2>Form Registrasi</h2>
<form method="POST" action="{{ route('form.handle') }}">
    @csrf
    <div class="mb-3">
        <label>Nama</label>
        <input type="text" name="name" class="form-control" value="{{ old('name') }}">
        @error('name') <div class="text-danger">{{ $message }}</div> @enderror
    </div>
    <div class="mb-3">
        <label>Email</label>
        <input type="email" name="email" class="form-control" value="{{ old('email') }}">
        @error('email') <div class="text-danger">{{ $message }}</div> @enderror
    </div>
    <div class="mb-3">
        <label>Umur</label>
        <input type="number" name="age" class="form-control" value="{{ old('age') }}">
        @error('age') <div class="text-danger">{{ $message }}</div> @enderror
    </div>
    <div class="mb-3">
        <label>Password</label>
        <input type="password" name="password" class="form-control">
        @error('password') <div class="text-danger">{{ $message }}</div> @enderror
    </div>
    <div class="mb-3">
        <label>Jenis Kelamin</label><br>
        <div class="form-check form-check-inline">
            <input class="form-check-input" type="radio" name="gender" value="male"> Laki-laki
        </div>
        <div class="form-check form-check-inline">
            <input class="form-check-input" type="radio" name="gender" value="female"> Perempuan
        </div>
        @error('gender') <div class="text-danger">{{ $message }}</div> @enderror
    </div>
    <div class="mb-3">
        <label>Role</label>
        <select name="role" class="form-select">
            <option value="">-- Pilih Role --</option>
            <option value="user">User</option>
            <option value="admin">Admin</option>
        </select>
        @error('role') <div class="text-danger">{{ $message }}</div> @enderror
    </div>
    <div class="mb-3">
        <label>Bio</label>
        <textarea name="bio" class="form-control">{{ old('bio') }}</textarea>
        @error('bio') <div class="text-danger">{{ $message }}</div> @enderror
    </div>
    <div class="form-check mb-3">
        <input type="checkbox" id="confirm" name="confirm" class="form-check-input">
        <label class="form-check-label" for="confirm">Saya konfirmasi informasi sudah benar</label>
        @error('confirm') <div class="text-danger">{{ $message }}</div> @enderror
    </div>
    <button type="submit" class="btn btn-primary" id="submitBtn" disabled>Submit</button>
</form>

<script>
    document.getElementById('confirm').addEventListener('change', function() {
        document.getElementById('submitBtn').disabled = !this.checked;
    });
</script>
</body>
</html>

```

- Buat View Hasil Buat file view Blade baru untuk menampilkan hasil. Buat file bernama result.blade.php di dalam direktori resources/views dan isikan kode berikut:

```
<!DOCTYPE html>
<html>
<head>
  <title>Hasil Form</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
</head>
<body class="container mt-5">
  <h2>Data Form yang Dikirim</h2>
  @if ($data)
    <ul class="list-group">
      @foreach ($data as $key => $value)
        <li class="list-group-item"><strong>{{ ucfirst($key) }}:</strong> {{ $value }}</li>
      @endforeach
    </ul>
  @else
    <p>Tidak ada data tersedia.</p>
  @endif
</body>
</html>
```

- Jalankan aplikasi dan tunjukkan hasil di browser. <http://127.0.0.1:8000/form>



Form Registrasi

Nama  
Bunga Alfa Zahrah

Email  
bungaalfazahrah@gmail.com

Umur  
19

Password  
...

Jenis Kelamin  
☐ Laki-laki ☒ Perempuan

Role  
User

Bio  
good job

☒ Saya konfirmasi informasi sudah benar

Submit

## 2.2 Praktikum 2 – Validasi Kustom dan Pesan Error di Laravel 12

- Mendefinisikan Route Buka project praktikum 1 sebelumnya, kemudian tambahkan route baru berikut:

```
<?php

use Illuminate\Support\Facades\Route;
use App\Http\Controllers\FormController;
use App\Http\Controllers\RegisterController;

Route::get('/register', [RegisterController::class, 'showForm'])->name('register.show');
Route::post('/register', [RegisterController::class, 'handleForm'])->name('register.handle');

Route::get('/form', [FormController::class, 'showForm'])->name('form.show');
Route::post('/form', [FormController::class, 'handleForm'])->name('form.handle');
Route::get('/result', [FormController::class, 'showResult'])->name('form.result');
Route::get('/', function () {
    return view('welcome');
});
```

- Buat Controller baru Di dalam project form-app buatlah sebuah controller baru: php artisan make:controller RegisterController Ini membuat app/Http/Controllers/RegisterController.php

```
<?php

namespace App\Http\Controllers;
use Illuminate\Http\Request;
class RegisterController extends Controller
{
    public function showForm()
    {
        return view('register');
    }

    public function handleForm(Request $request)
    {
        $customMessages = [
            'name.required' => 'Kami perlu tahu nama Anda!',
            'email.required' => 'Email Anda penting bagi kami.',
            'email.email' => 'Hmm... itu tidak terlihat seperti email yang valid.',
            'password.required' => 'Jangan lupa untuk set password.',
            'password.min' => 'Password harus minimal :min karakter.',
            'username.regex' => 'Username hanya boleh berisi huruf dan angka.',
        ];

        $request->validate([
            'name' => 'required|string|max:100',
            'email' => 'required|email',
            'username' => ['required', 'regex:/^[a-zA-Z0-9]+$/'],
            'password' => 'required|min:6',
        ], $customMessages);

        return redirect()->route('register.show')->with('success', 'Registrasi berhasil!');
    }
}
```

- Buat Blade View. Buat file view di resources/views/register.blade.php:

```
<!DOCTYPE html>
<html>
<head>
  <title>Form Register</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
</head>
<body class="container mt-5">
  <h2>Contoh Validasi Kustom</h2>

  @if (session('success'))
    <div class="alert alert-success">
      {{ session('success') }}
    </div>
  @endif

  <form method="POST" action="{{ route('register.handle') }}">
    @csrf
    <div class="mb-3">
      <label for="name">Nama Lengkap</label>
      <input name="name" class="form-control" value="{{ old('name') }}">
      @error('name') <div class="text-danger">{{ $message }}</div> @enderror
    </div>
    <div class="mb-3">
      <label for="email">Alamat Email</label>
      <input name="email" class="form-control" value="{{ old('email') }}">
      @error('email') <div class="text-danger">{{ $message }}</div> @enderror
    </div>
    <div class="mb-3">
      <label for="username">Username</label>
      <input name="username" class="form-control" value="{{ old('username') }}">
      @error('username') <div class="text-danger">{{ $message }}</div> @enderror
    </div>
    <div class="mb-3">
      <label for="password">Password</label>
      @error('password') <div class="text-danger">{{ $message }}</div> @enderror
      <input type="password" name="password" class="form-control">
    </div>
    <button type="submit" class="btn btn-primary">Register</button>
  </form>
</body>
</html>
```

- Jalankan aplikasi <http://127.0.0.1:8000/logic>

127.0.0.1:8000/register

Ringkas

### Contoh Validasi Kustom

Nama Lengkap  
Bunga Alfa Zahrah

Alamat Email  
bunga.com

Username  
Bunga

Password  
\*\*\*\*\*

Register

## 2.3 Praktikum 3 – Multi-Step Form Submission dengan Session Data

- Buat proyek laravel dengan perintah `Laravel new multistep-form-app`, kemudian masuk ke proyek dengan perintah `cd multistep-form-app`.

- Buat Layout Dasar Buat file layout di resources/views/layouts/app.blade.php :

```

<!DOCTYPE html>
<html lang="id">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>@yield('title', 'Multi-Step Form')</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
  <style>
    .step-progress {
      margin-bottom: 30px;
    }
    .step-item {
      text-align: center;
      position: relative;
    }
    .step-item .step-number {
      width: 40px;
      height: 40px;
      border-radius: 50%;
      background: #e9ecef;
      display: flex;
      align-items: center;
      justify-content: center;
      margin: 0 auto 10px;
      font-weight: bold;
    }
    .step-item.active .step-number {
      background: #007bff;
      color: white;
    }
    .step-item.completed .step-number {
      background: #28a745;
      color: white;
    }
    .step-line {
      position: absolute;
      top: 20px;
      left: 50%;
      width: 100%;
      height: 2px;
      background: #e9ecef;
      z-index: -1;
    }
    .form-step {
      display: none;
    }
    .form-step.active {
      display: block;
    }
    .card {
      box-shadow: 0 0.5rem 1rem rgba(0, 0, 0, 0.15);
    }
  </style>
</head>
<body>
  <nav class="navbar navbar-dark bg-dark">
    <div class="container">
      <span class="navbar-brand mb-0 h1">Multi-Step Form Demo</span>
    </div>
  </nav>

  <div class="container mt-4">
    @yield('content')
  </div>

  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>

```



- Buat Routes Buka file routes/web.php dan definisikan routes untuk setiap langkah form:

```
<?php

use Illuminate\Support\Facades\Route;
use App\Http\Controllers\MultiStepFormController;

Route::get('/multistep', [MultiStepFormController::class, 'showStep1'])->name('multistep.step1');
Route::post('/multistep/step1', [MultiStepFormController::class, 'storeStep1'])->name('multistep.storeStep1');
Route::get('/multistep/step2', [MultiStepFormController::class, 'showStep2'])->name('multistep.step2');
Route::post('/multistep/step2', [MultiStepFormController::class, 'storeStep2'])->name('multistep.storeStep2');
Route::get('/multistep/step3', [MultiStepFormController::class, 'showStep3'])->name('multistep.step3');
Route::post('/multistep/step3', [MultiStepFormController::class, 'storeStep3'])->name('multistep.storeStep3');
Route::get('/multistep/summary', [MultiStepFormController::class, 'showSummary'])->name('multistep.summary');
Route::post('/multistep/complete', [MultiStepFormController::class, 'complete'])->name('multistep.complete');

Tabnine | Edit | Test | Explain | Document
Route::get('/', function () {
    return view('welcome');
});
```

- Buat Controller Buat Controller untuk menangani logika multi-step form, dengan perintah php artisan make:controller MultiStepFormController, dan buka file

app/Http/Controllers/MultiStepFormController.php dan tambahkan metode berikut :

```

<?php

namespace App\Http\Controllers;
use Illuminate\Http\Request;
class MultiStepFormController extends Controller
{
    // Step 1 - Informasi Pribadi
    Tabname | Edit | Test | Explain | Document
    public function showStep1(){
        return view('multistep.step1', [
            'step' => 1,
            'progress' => 0
        ]);
    }

    Tabname | Edit | Test | Explain | Document
    public function storeStep1(Request $request){
        $validated = $request->validate([
            'full_name' => 'required|string|max:100',
            'email' => 'required|email',
            'phone' => 'required|string|max:15',
            'address' => 'required|string|max:200',
        ]);
        // Simpan data ke session
        session(['step1_data' => $validated]);
        return redirect()->route('multistep.step2');
    }

    // Step 2 - Informasi Pendidikan
    Tabname | Edit | Test | Explain | Document
    public function showStep2(){
        if (!session('step1_data')) {
            return redirect()->route('multistep.step1');
        }
        return view('multistep.step2', [
            'step' => 2,
            'progress' => 33
        ]);
    }

    Tabname | Edit | Test | Explain | Document
    public function storeStep2(Request $request){
        $validated = $request->validate([
            'education' => 'required|string|max:50',
            'institution' => 'required|string|max:100',
            'graduation_year' => 'required|integer|min:1900|max:' . date('Y'),
            'major' => 'required|string|max:100',
        ]);
        session(['step2_data' => $validated]);
        return redirect()->route('multistep.step3');
    }

    // Step 3 - Pengalaman Kerja
    Tabname | Edit | Test | Explain | Document
    public function showStep3(){
        if (!session('step1_data') || !session('step2_data')) {
            return redirect()->route('multistep.step1');
        }
        return view('multistep.step3', [
            'step' => 3,
            'progress' => 66
        ]);
    }

    Tabname | Edit | Test | Explain | Document
    public function storeStep3(Request $request){
        $validated = $request->validate([
            'current_job' => 'nullable|string|max:100',
            'company' => 'nullable|string|max:100',
            'experience_years' => 'required|integer|min:0|max:50',
            'skills' => 'required|string|max:200',
        ]);
        session(['step3_data' => $validated]);
        return redirect()->route('multistep.summary');
    }

    // Summary - Ringkasan Data
    Tabname | Edit | Test | Explain | Document
    public function showSummary(){
        $step1Data = session('step1_data');
        $step2Data = session('step2_data');
        $step3Data = session('step3_data');
        if (!$step1Data || !$step2Data || !$step3Data) {
            return redirect()->route('multistep.step1');
        }
        return view('multistep.summary', [
            'step' => 4,
            'progress' => 100,
            'step1Data' => $step1Data,
            'step2Data' => $step2Data,
            'step3Data' => $step3Data
        ]);
    }

    // Complete - Proses Final
    Tabname | Edit | Test | Explain | Document
    public function complete(Request $request){
        // Di sini Anda bisa menyimpan data ke database
        // Untuk demo, kita hanya akan menampilkan pesan sukses
        $allData = [
            'personal' => session('step1_data'),
            'education' => session('step2_data'),
            'experience' => session('step3_data')
        ];
        // Hapus session data
        $request->session()->forget(['step1_data', 'step2_data', 'step3_data']);
        return view('multistep.complete', [
            'data' => $allData
        ]);
    }
}

```

- Buat view untuk Setiap Step Buat direktori resources/views/multistep - Step 1- Informasi Pribadi : resources/views/multistep/step1.blade.php

- Step 2- Informasi Pendidikan :

11 / 18

- Step 3-

12 / 18

- Summary :

13 / 18

- Complete :

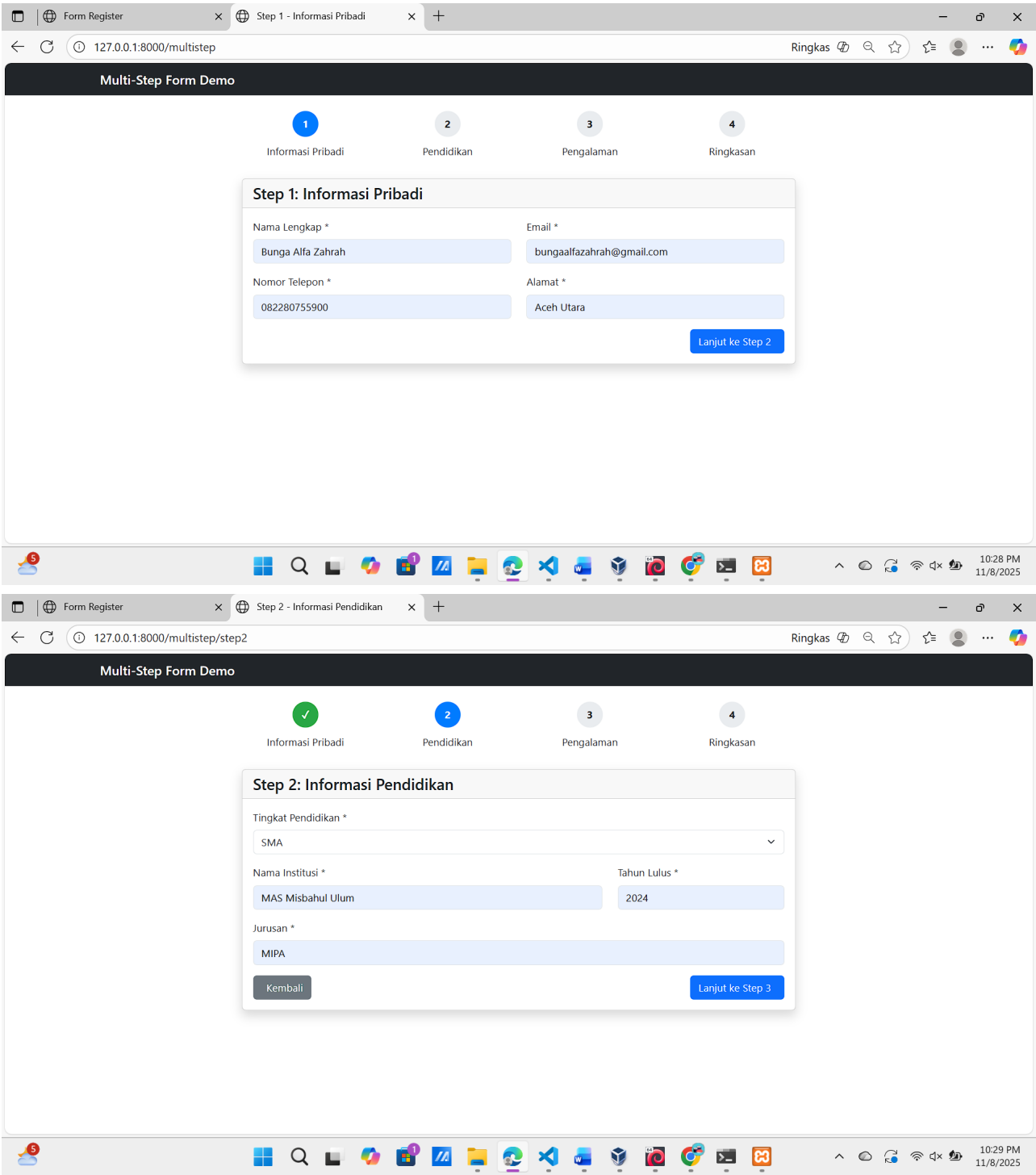
14 / 18

```

@extends('layouts.app')
@section('title', 'Pendaftaran Selesai')
@section('content')
<div class="row justify-content-center">
    <div class="col-md-6">
        <div class="card text-center">
            <div class="card-body py-5">
                <div class="mb-4">
                    <i class="fas fa-check-circle text-success" style="font-size: 4rem;"></i>
                </div>
                <h2 class="card-title text-success mb-3">Pendaftaran Berhasil!</h2>
                <p class="card-text lead mb-4">
                    Terima kasih telah melengkapi formulir pendaftaran. Data Anda telah berhasil disimpan.
                </p>
                <div class="alert alert-info text-start">
                    <h6><i class="fas fa-envelope me-2"></i>Informasi Penting:</h6>
                    <ul class="mb-0">
                        <li>Anda akan menerima email konfirmasi dalam 24 jam</li>
                        <li>Proses verifikasi membutuhkan waktu 2-3 hari kerja</li>
                        <li>Tim kami akan menghubungi Anda untuk langkah selanjutnya</li>
                    </ul>
                </div>
                <div class="mt-4">
                    <a href="{{ route('multistep.step1') }}" class="btn btn-primary me-2">
                        <i class="fas fa-plus me-2"></i>Daftar Lagi
                    </a>
                    <a href="/" class="btn btn-outline-secondary">
                        <i class="fas fa-home me-2"></i>Kembali ke Home
                    </a>
                </div>
            </div>
        </div>
    </div>
    <!-- Debug Data (bisa dihapus di production) -->
    @if(env('APP_DEBUG'))
        <div class="card mt-4">
            <div class="card-header">
                <h6 class="mb-0">Data yang Disimpan (Debug)</h6>
            </div>
            <div class="card-body">
                <pre>{{ json_encode($data, JSON_PRETTY_PRINT) }}</pre>
            </div>
        </div>
    @endif
</div>
</div>
@endsection

```

- Jalankan aplikasi `http://127.0.0.1:8000/multistep` Output:





The first screenshot shows the 'Step 3: Pengalaman Kerja' (Work Experience) form. It includes fields for 'Pekerjaan Saat Ini' (Current Job) with the value 'Mahasiswa', 'Perusahaan' (Company) with 'D'Mentai', 'Pengalaman Kerja (Tahun) \*' (Work Experience in Years) with '5', and 'Keahlian/Keterampilan \*' (Skills) with 'Contoh: PHP, Laravel, JavaScript, MySQL...'. There are 'Kembali' (Back) and 'Lihat Ringkasan' (View Summary) buttons.

The second screenshot shows the 'Ringkasan Data' (Data Summary) screen. It displays a consolidated view of the user's information across four steps: 'Informasi Pribadi' (Personal Info), 'Pendidikan' (Education), 'Pengalaman' (Experience), and 'Ringkasan' (Summary). The summary includes fields for 'Nama Lengkap' (Full Name), 'Email', 'Telepon' (Phone), 'Alamat' (Address), 'Tingkat Pendidikan' (Education Level), 'Institusi' (Institution), 'Tahun Lulus' (Graduation Year), 'Jurusan' (Major), 'Pekerjaan Saat Ini' (Current Job), 'Perusahaan' (Company), and 'Keahlian' (Skills). There are 'Edit' buttons for each section and a 'Kirim Data' (Submit Data) button at the bottom.

### 3. Hasil dan Pembahasan

Dari hasil praktikum yang dilakukan, diperoleh hasil sebagai berikut:

- Pada Praktikum 1, form berhasil mengirim data dari view ke controller dan menampilkannya kembali melalui view hasil.
- Pada Praktikum 2, validasi berhasil dijalankan. Ketika data yang dimasukkan tidak sesuai, Laravel menampilkan pesan error secara otomatis.
- Pada Praktikum 3, multi-step form berjalan dengan baik menggunakan session, memungkinkan pengguna melanjutkan form ke langkah berikutnya tanpa kehilangan data sebelumnya.

Hal ini menunjukkan bahwa Laravel menyediakan sistem validasi dan manajemen data form yang efisien, serta memudahkan developer membangun form dinamis yang terstruktur dengan baik.

---

## 4. Kesimpulan

Dari praktikum ini dapat disimpulkan bahwa:

- Laravel menyediakan mekanisme request–response handling yang sederhana namun kuat untuk menangani input pengguna.
  - Fitur validasi Laravel membantu memastikan keamanan dan keabsahan data sebelum diproses lebih lanjut.
  - Dengan penggunaan session, developer dapat membangun form bertahap (multi-step form) tanpa kehilangan data antar halaman.
  - Implementasi form di Laravel dapat meningkatkan efisiensi dan kualitas aplikasi web yang melibatkan interaksi pengguna.
- 

## 5. Referensi

- <https://www.youtube.com/watch?v=MFh0Fd7BsjE>
  - <https://www.codepolitan.com/tutorial/laravel-form-validation>
  - <https://www.petanikode.com/laravel-pemula/>
-