

# **PRAKTIKUM PEMROGRAMAN WEB**

Laporan Praktikum Jobsheet 3



Dosen Pengampu: Dr. Ika Parma Dewi, M.Pd.T.

Oleh :

Rian Septiawan 23076052

**PRODI PENDIDIKAN TEKNIK INFORMATIKA**

**DEPARTEMEN ELEKTRONIKA**

**FAKULTAS TEKNIK**

**UNIVERSITAS NEGERI PADANG**

**2025**

<b>FT UNP Padang</b>	<b>Lembaran : JOBSHEET 3</b>
<b>Jurusan : Pendidikan Teknik Informatika</b>	<b>Matakuliah: Pratikum Pemrograman Web</b>
<b>Waktu : 3 x 50 Menit</b>	<b>Topik : Database dan Halaman Data Poliklinik</b>
<b>Kode : TIK1.61.4318</b>	<b>Judul : Membuat Database dan Halaman Data Poliklinik</b>

<b>Fakultas</b>	<b>Proram Studi</b>	<b>Kode MK</b>	<b>Waktu</b>
Teknik	Pendidikan Teknik Informatika	TIK1.61.4318	2 x 170 Menit

## A. TUJUAN PRAKTIKUM

Setelah mengikuti perkuliahan ini diharapkan mahasiswa mampu :

1. Memahami dan menjelaskan konsep Framework Laravel.
2. Dapat melakukan instanlasi composer.
3. Dapat melakukan instalasi Laravel.
4. Dapat membuat projek dengan Laravel.

## B. ALAT DAN BAHAN

1. PC/Laptop
2. Browser Internet (Internet Explorer/Mozilla Firefox/Google Chrome)
3. XAMPP/Laragon
4. Text Editor (Visual Studio Code)

## C. TEORI SINGKAT

### 1. Pengertian Database

Database adalah kumpulan data yang terorganisir secara sistematis sehingga dapat diakses, dikelola, dan diperbarui dengan mudah. Database digunakan untuk menyimpan berbagai jenis informasi, seperti data pengguna, transaksi, atau inventaris.

### 2. Komponen Database

- a. Data: Informasi yang disimpan dalam database.
- b. DBMS (Database Management System): Perangkat lunak yang mengelola database, seperti MySQL, PostgreSQL, dan Oracle.
- c. Query: Perintah untuk mengambil atau memanipulasi data dalam database, sering menggunakan SQL (Structured Query Language).

### 3. Jenis Database

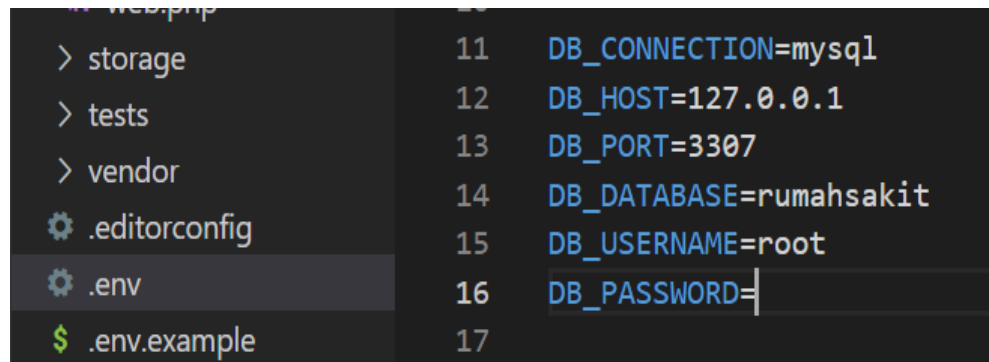
- a. Relasional: Menggunakan tabel dengan hubungan antar data (contoh: MySQL, PostgreSQL).

- b. Non-Relasional (NoSQL): Menyimpan data dalam format fleksibel seperti dokumen atau key-value (contoh: MongoDB, Redis).
4. Keunggulan Database
- 1) Memudahkan pencarian dan pengelolaan data.
  - 2) Mengurangi redundansi dan inkonsistensi data.
  - 3) Meningkatkan keamanan dan integritas data.
5. Konsep Penting dalam Database
- 1) Primary Key: Identitas unik setiap data dalam tabel.
  - 2) Foreign Key: Kunci yang menghubungkan satu tabel dengan tabel lain.
  - 3) Normalization: Proses untuk menghindari duplikasi data dan meningkatkan efisiensi penyimpanan.

## D. LANGKAH KERJA


### 4.1. Membuat Tabel di Database

1. Terlebih dahulu kita beri nama database kita, misalnya nama databasenya “jobsheet”, buka halaman **.env** kemudian ubah nama DB\_DATABASE



```
11 DB_CONNECTION=mysql
12 DB_HOST=127.0.0.1
13 DB_PORT=3307
14 DB_DATABASE=rumahsakit
15 DB_USERNAME=root
16 DB_PASSWORD=
17
```

2. Buat nama database yang telah dibuat kedalam database **phpMyAdmin**



3. Kemudian buka kembali VSCode pergi ke terminal, ketik **php artisan make:model nama\_tabel -m** untuk membuat tabel baru sekaligus models. Karna yang akan kita buat nama tabelnya poliklinik, maka ganti **nama\_tabel** dengan **poliklinik** seperti pada gambar.
4. Kemudian buka folder “migration” dalam folder “database”, terlihat file database

```
PS C:\Users\ryans\Documents\PROJECT DEVELOP\A Github> cd "C:\laragon\www\rumah-sakit"
PS C:\laragon\www\rumah-sakit> php artisan make:model poliklinik -m

INFO Model [C:\laragon\www\rumah-sakit\app\Models\poliklinik.php] created successfully.

INFO Migration [C:\laragon\www\rumah-sakit\database\Migrations\2025_02_24_023416_create_polikliniks_table.php] created successfully
```

5. yang baru dibuat

```
▼ migrations
  2014_10_12_000000_create_users_table.php
  2014_10_12_100000_create_password_resets_table.php
  2019_08_19_000000_create_failed_jobs_table.php
  2019_12_14_000001_create_personal_access_tokens_table.php
  2025_02_24_023416_create_polikliniks_table.php
```

6. Buat seperti ini pada file tersebut

```

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       * @return void
13       */
14     public function up()
15     {
16         Schema::create('poliklinik', function (Blueprint $table) {
17             $table->id()->unique();
18             $table->string('nama_poliklinik');
19             $table->timestamps();
20         });
21     }
22
23     /**
24      * Reverse the migrations.
25      *
26      * @return void
27      */
28     public function down()
29     {
30         Schema::dropIfExists('poliklinik');
31     }
32 };
33

```

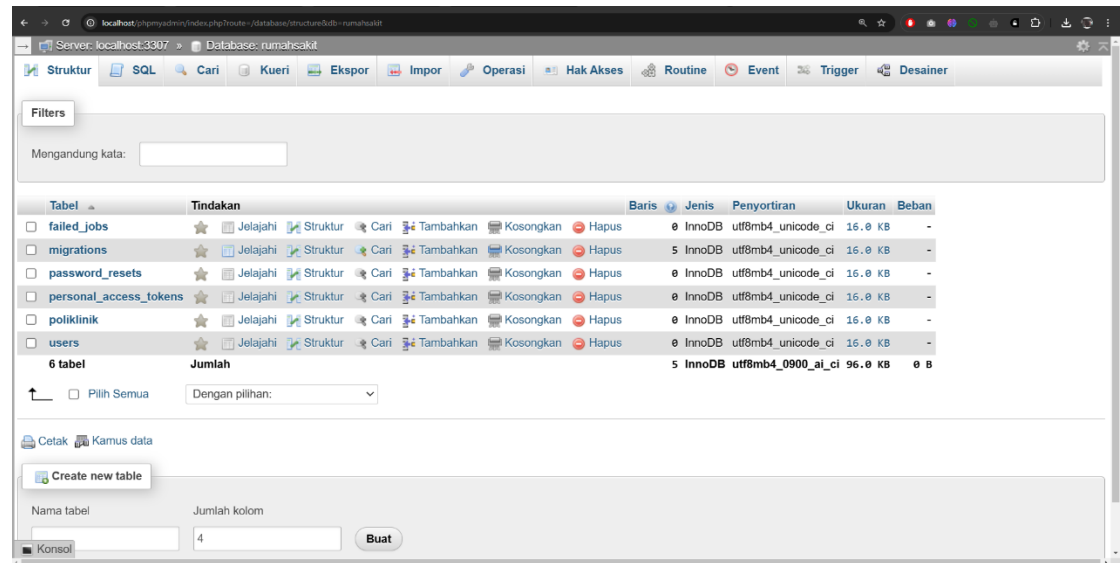
7. Setelah itu, jalankan **php artisan migrate**

```

PS C:\laragon\www\rumah-sakit> php artisan make:model poliklinik -m
INFO Model [C:\laragon\www\rumah-sakit\app\Models\poliklinik.php] created successfully.
INFO Migration [C:\laragon\www\rumah-sakit\database\Migrations\2025_02_24_023416_create_polikliniks_table.php] created successfully.
PS C:\laragon\www\rumah-sakit> php artisan migrate
INFO Preparing database.
Creating migration table ..... 88ms DONE
INFO Running migrations.
2014_10_12_000000_create_users_table ..... 67ms DONE
2014_10_12_100000_create_password_resets_table ..... 55ms DONE
2019_08_19_000000_create_failed_jobs_table ..... 53ms DONE
2019_12_14_000001_create_personal_access_tokens_table ..... 77ms DONE
2025_02_24_023416_create_polikliniks_table ..... 47ms DONE
PS C:\laragon\www\rumah-sakit>

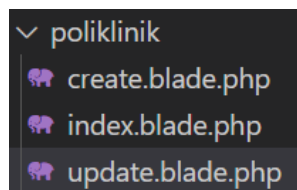
```

Maka database otomatis masuk ke tabel yang telah dibuat



## 4.2. Buat Halaman Data Poliklinik

1. Buat folder “poliklinik” pada folder “views” berisi 3 file seperti berikut



2. Isi halaman **index.blade.php**

```

1. @extends('layout.admin')
2.
3. @section('title', 'Poliklinik')
4.
5. @section('content')
6. <!-- Page Heading -->
7. <h1 class="h3 mb-2 text-gray-800">Data Poliklinik</h1>
8.
9. <!-- DataTales Example -->
10. <div class="card shadow mb-4">
11.   <div class="card-header py-3">
12.     <a href="{ route('poliklinik.create') }" class="btn btn-primary btn-sm"><i class="fas fa-plus"></i> Tambah</a>
13.   </div>
14.   <div class="card-body">
15.     <div class="table-responsive">
16.       <table class="table table-bordered" id="dataTable" width="100%"
17.         cellspacing="0">
18.         <thead>
19.           <tr>
20.             <th>No</th>
21.             <th>Nama Poliklinik</th>

```

```
22.         </tr>
23.     </thead>
24.     <tbody>
25.         @php $no = 1; @endphp
26.         @foreach ($poliklinik as $item)
27.             <tr>
28.                 <td>{{ $no++ }}</td>
29.                 <td>{{ $item->nama_poliklinik }}</td>
30.                 <td>
31.                     <a href="{{ route('poliklinik.edit', $item->id) }}"
class="btn btn-warning btn-sm">Edit</a>
32.                     <form action="{{ route('poliklinik.destroy', $item-
>id) }}" method="POST" class="d-inline delete-form">
33.                         @csrf
34.                         @method('DELETE')
35.                         <button type="button" class="btn btn-danger btn-sm
btn-delete">Hapus</button>
36.                     </form>
37.                 </td>
38.             </tr>
39.         @endforeach
40.         @include('sweetalert::alert')
41.     </tbody>
42. </table>
43. </div>
44. </div>
45. </div>
46.
47. <!-- Sertakan SweetAlert2 -->
48. <script src="https://cdn.jsdelivr.net/npm/sweetalert2@10"></script>
49. <!-- Sertakan file JavaScript khusus -->
50. <script src="{{ asset('js/sweetalert.js') }}"></script>
51. @endsection
```

### 3. Isi halaman **create.blade.php**

```
@extends('layout.admin')
@section('title', 'Tambah Poliklinik')
@section('content')
<div class="card shadow mb-4">
    <div class="card-header py-3">
        <h6 class="m-0 font-weight-bold text-primary">Tambah Poliklinik Baru</h6>
    </div>
    <div class="card-body">
        <form action="{{ route('poliklinik.add') }}" method="POST">
            @csrf
            <div class="form-group">
                <label for="nama_poliklinik">Nama Poliklinik</label>
                <input type="text" name="nama_poliklinik" id="nama_poliklinik" class="form-control" required>
            </div>
            <div class="text-center">
                <button type="submit" class="btn btn-primary">Simpan</button>
                <a href="{{ route('poliklinik.index') }}" class="btn btn-secondary">Batal</a>
            </div>
        </form>
    </div>
</div>
@endsection
@include('sweetalert::alert')
```

### 4. Isi halaman **update.blade.php**

```
@extends('layout.admin')
@section('title', 'Edit Poliklinik')
@section('content')
<div class="card shadow mb-4">
    <div class="card-header py-3">
        <h6 class="m-0 font-weight-bold text-primary">Edit Poliklinik</h6>
    </div>
    <div class="card-body">
        <form action="{{ route('poliklinik.update', $poliklinik->id) }}" method="POST">
            @method('PUT')
            @csrf
            <div class="form-group">
                <label for="nama_poliklinik">Nama Poliklinik</label>
                <input type="text" name="nama_poliklinik" id="nama_poliklinik" class="form-control"
required value="{{ $poliklinik->nama_poliklinik }}">
            </div>
            <div class="text-center">
                <button type="submit" class="btn btn-primary">Update</button>
                <a href="{{ route('poliklinik.index') }}" class="btn btn-secondary">Batal</a>
            </div>
        </form>
    </div>
</div>
@endsection
@include('sweetalert::alert')
```



#### 4.3. Buat Model Dan Poliklinik Kontroler

1. Buka **poliklinik** pada folder **Models** di dalam folder **app**

Kemudian ketik ini pada file tersebut

```
app > Models > poliklinik.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 poliklinik extends Model
9  {
10     use HasFactory;
11     protected $table = 'poliklinik';
12     protected $fillable = ['nama_poliklinik',];
13 }
14
```

2. Buat Controllernya dengan nama **PoliklinikController**  
**php artisan make:controller PoliklinikController --resource**

```
<?php

namespace App\Http\Controllers;

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

class PoliklinikController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        $poliklinik = Poliklinik::latest()->get();
        return view('poliklinik.index', [
            'poliklinik' => $poliklinik
        ]);
    }

    /**
     * Show the form for creating a new resource.
     */
}
```

```
*
* @return \Illuminate\Http\Response
*/
public function create()
{
    return view('poliklinik.create');
}

/**
 * Store a newly created resource in storage.
 *
 * @param \Illuminate\Http\Request $request
 * @return \Illuminate\Http\Response
 */
public function store(Request $request)
{
    // Validasi data
    $validatedData = $request->validate([
        'nama_poliklinik' => 'required|max:255',
    ]);

    // Simpan data ke dalam database
    Poliklinik::create($validatedData);

    return redirect()->route('poliklinik.index')->with('success', 'Data
berhasil disimpan!');
    return redirect()->back()->withInput()->withErrors(['error' => 'Gagal
menyimpan data. Silakan coba lagi.']);
}

/**
 * Display the specified resource.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function show($id)
{
    //
}

/**
 * Show the form for editing the specified resource.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function edit($id)
{
    $poliklinik = Poliklinik::findOrFail($id);
    return view('poliklinik.update', ['poliklinik' => $poliklinik]);
}
```

```
}

/**
 * Update the specified resource in storage.
 *
 * @param \Illuminate\Http\Request $request
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function update(Request $request, $id)
{
    // Validasi data
    $validatedData = $request->validate([
        'nama_poliklinik' => 'required|max:255',
    ]);

    // Cari poliklinik berdasarkan ID
    $poliklinik = Poliklinik::findOrFail($id);

    // Update data poliklinik
    $poliklinik->nama_poliklinik = $validatedData['nama_poliklinik'];
    $poliklinik->save();

    // Redirect ke halaman index dengan pesan sukses
    return redirect()->route('poliklinik.index')->with('success', 'Data
berhasil diperbarui!');
    return redirect()->back()->withInput()->withErrors(['error' => 'Gagal
memperbarui data. Silakan coba lagi.']);
}

/**
 * Remove the specified resource from storage.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function destroy($id)
{
    $poliklinik = Poliklinik::findOrFail($id);

    // Hapus data poliklinik dari database
    $poliklinik->delete();

    return redirect()->route('poliklinik.index')->with('success', 'Data
berhasil dihapus!');
}
}
```

3. Buka file **web.php**, kemudian tambahkan source code ini

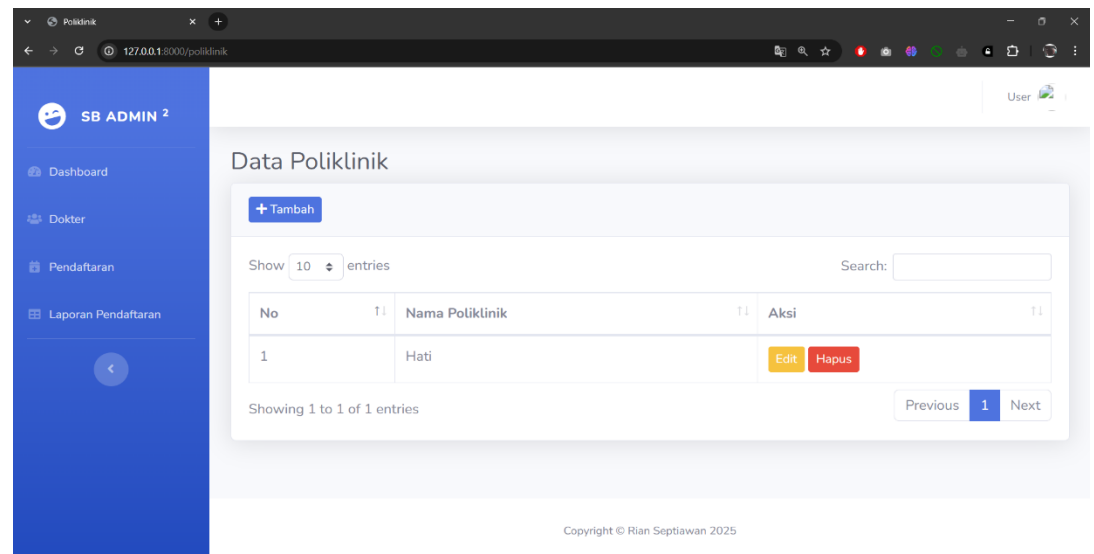
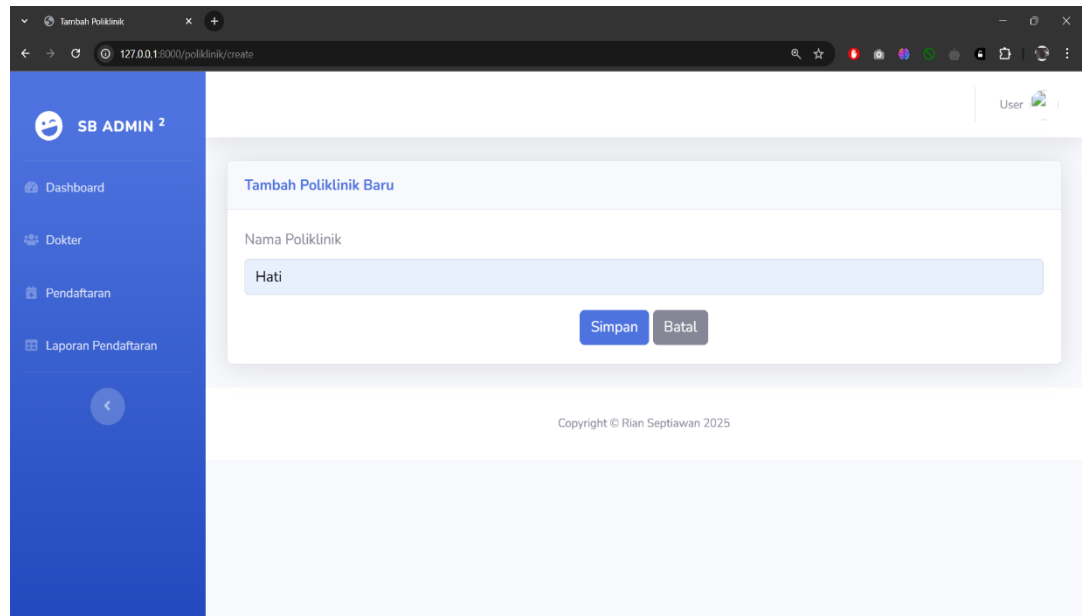
```
routes > web.php > ...
61 //Poliklinik
62 Route::get('/poliklinik/create', [PoliklinikController::class, 'create'])->name('poliklinik.create');
63 Route::post('/poliklinik/add', [PoliklinikController::class, 'add'])->name('poliklinik.add');
64 Route::get('/poliklinik', [PoliklinikController::class, 'index'])->name('poliklinik.index');
65 Route::get('/poliklinik/edit/{id}', [PoliklinikController::class, 'edit'])->name('poliklinik.edit');
66 Route::put('/poliklinik/update/{id}', [PoliklinikController::class, 'update'])->name('poliklinik.update');
67 Route::delete('/poliklinik/{id}', [PoliklinikController::class, 'destroy'])->name('poliklinik.destroy');
68
```

4. Tambahkan ini di bagian **layout** yang bagian menu **Data Poliklinik**

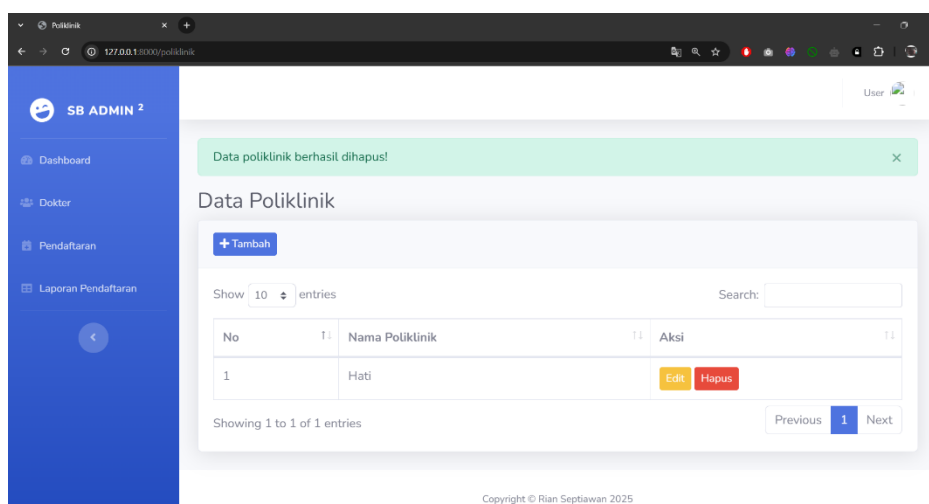
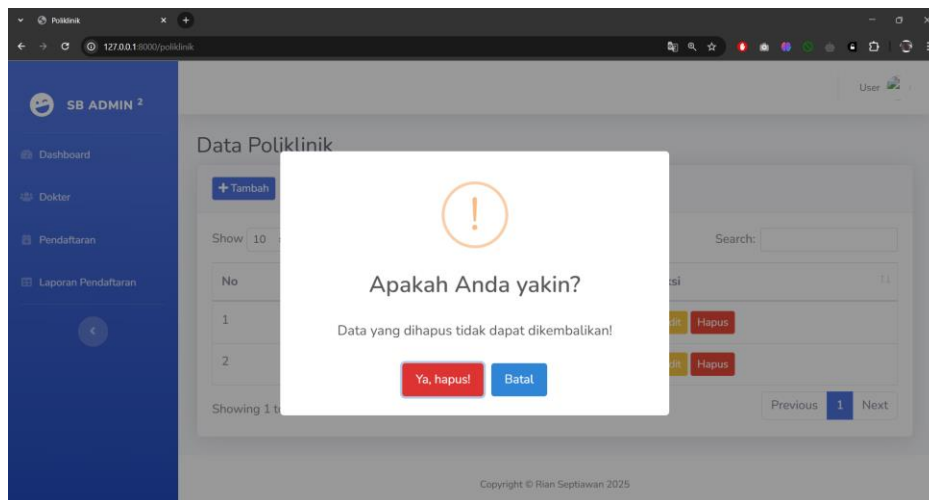
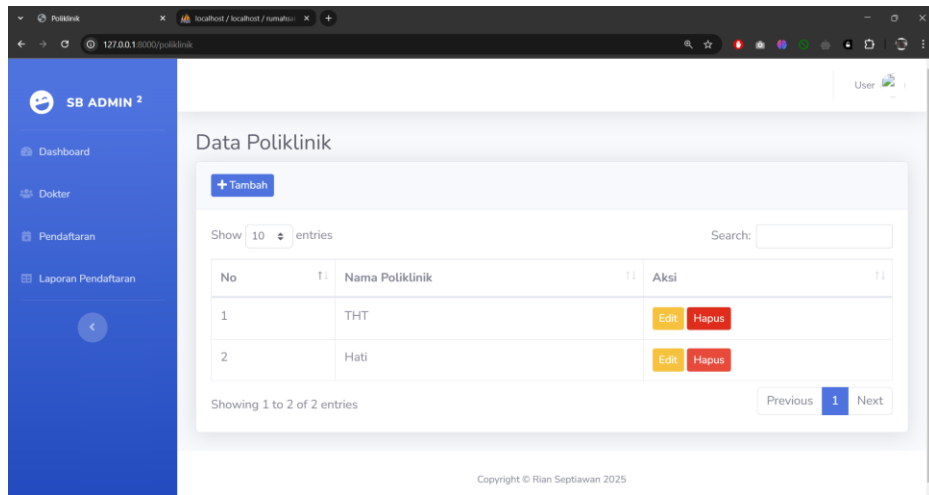
```
<a class="collapse-item" href="{{ route('poliklinik.index') }}">Data Poliklinik</a>
```

Atau buka link <http://127.0.0.1:8000/poliklinik> setelah menjalankan perintah

**php artisan serve**



5. Periksa apakah membuat, mengubah, dan menghapus data berfungsi dengan baik!



#### **E. LEMBAR KERJA**

1. Lakukan praktikum sesuai langkah kerja yang diberikan
2. Buat laporannya

## F. DAFTAR PUSTAKA

- [1] Deitel. 2001 Internet & WWW How to Programing, Prentice Hall
- [2]. Budi Raharjo. 2012. Modul Prmograman WEB. Modula: Bandung.
- [3]. Bunafit Nugroho. 2004. PHP & MySQL Dengan Editor Dreamweaver MX. Andi Yogyakarta
- [4] Sklar, David, and Adam Trachtenberg. PHP Cookbook. O'Reilly, 2008.
- [5] Welling, Luke. PHP and MySQL Web Development. Developers Library, 2012
- [6] Yank, Kevin, and Cameron Adams. Simply Javascript. Sitepoint, 2010
- [7] Manual PHP. 2014
- [8] Niederst, Jennifer. Learning Web Design (4th edition). O'Reilly. 2013