

LAPORAN PRAKTIKUM MATA KULIAH

PEMROGRAMAN WEB

TUGAS 14 – CRUD



DISUSUN OLEH:

L0122068 – Gardasvara Mistortoify

PROGRAM STUDI INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI DAN SAINS DATA

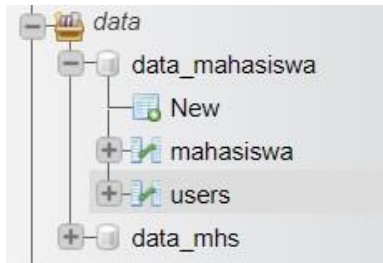
UNIVERSITAS SEBELAS MARET

2024

BAB I

ANALISIS SOURCE CODE

1. Analisis Database Tabel Baru “cinema” untuk movies, genre, dan movies_genre



Pertama buat pada phpmyadmin sebuah database cinema. kemudian untuk membuat tabel yakni movies dan genre, bisa dengan menggunakan “**php artisan make:model ...**” kemudian diatur untuk tabelnya pada models dan migrationnya seperti dibawah ini

```

public function up(): void
{
    Schema::create('movies', function (Blueprint $table) {
        $table->id();
        $table->string('title');
        $table->string('image')->nullable();
        $table->decimal('rating', 3, 1)->nullable();
        $table->string('production');
        $table->string('director');
        $table->date('release_date');
        $table->string('age_restriction');
        $table->integer('duration');
        $table->text('synopsis');
        $table->enum('status', ['available', 'unavailable', 'coming_soon']);
        $table->timestamps();
    });
}

```

Nantinya akan dijalankan command “**php artisan migrate**” sehingga apabila dicek di phpmyadmin, database cinema akan terdapat tabel movies dengan struktur saat diatur pada models

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1 id	bigint(20)		UNSIGNED	No	None		AUTO_INCREMENT	Change Drop More
<input type="checkbox"/>	2 title	varchar(255)	utf8mb4_unicode_ci		No	None			Change Drop More
<input type="checkbox"/>	3 image	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop More
<input type="checkbox"/>	4 rating	decimal(3,1)			Yes	NULL			Change Drop More
<input type="checkbox"/>	5 production	varchar(255)	utf8mb4_unicode_ci		No	None			Change Drop More
<input type="checkbox"/>	6 director	varchar(255)	utf8mb4_unicode_ci		No	None			Change Drop More
<input type="checkbox"/>	7 release_date	date			No	None			Change Drop More
<input type="checkbox"/>	8 age_restriction	varchar(255)	utf8mb4_unicode_ci		No	None			Change Drop More
<input type="checkbox"/>	9 duration	int(11)			No	None			Change Drop More
<input type="checkbox"/>	10 synopsis	text	utf8mb4_unicode_ci		No	None			Change Drop More
<input type="checkbox"/>	11 status	enum('available', 'unavailable', 'coming_soon')	utf8mb4_unicode_ci		No	None			Change Drop More
<input type="checkbox"/>	12 created_at	timestamp			Yes	NULL			Change Drop More
<input type="checkbox"/>	13 updated_at	timestamp			Yes	NULL			Change Drop More

Kemudian begitu juga dengan genre yang datanya akan di insert manual di phpmyadmin sehingga terisi seperti dibawah ini

				id	name	created_at	updated_at
<input type="checkbox"/>	Edit	Copy	Delete	1	Crime	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete	2	Thriller	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete	3	Action	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete	4	Comedy	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete	5	Mystery	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete	6	Adventure	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete	7	Horror	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete	8	Drama	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete	9	Romance	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete	10	Sci-Fi	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete	11	Fantasy	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete	12	Family	NULL	NULL
<input type="checkbox"/>	Edit	Copy	Delete	13	Biography	NULL	NULL

2. Analisis Kode PHP

a. Models/Movie

```
namespace App\Models;

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

class Movie extends Model
{
    use HasFactory;

    protected $fillable = [
        'title',
        'image',
        'rating',
        'production',
        'director',
        'release_date',
        'age_restriction',
        'duration',
        'synopsis',
        'status'
    ];

    protected $casts = [
        'release_date' => 'date',
        'rating' => 'string',
    ];

    public function genres()
    {
        return $this->belongsToMany(Genre::class, 'movie_genre');
    }

    public function getFormattedDurationAttribute()
    {
        $hours = intval($this->duration, 60);
        $minutes = $this->duration % 60;
        return sprintf('%dh %dm', $hours, $minutes);
    }
}
```

Model Movie dalam proyek ini berfungsi sebagai representasi dari entitas film dalam database. Model ini menggunakan trait HasFactory untuk memanfaatkan fitur factory

dalam pembuatan model selama pengujian. Atribut yang dapat diisi (fillable) meliputi berbagai properti film seperti title, image, rating, production, director, release_date, age_restriction, duration, synopsis, dan status. Atribut fillable memastikan bahwa hanya properti-properti ini yang dapat diisi secara massal untuk menghindari assignment data yang tidak diinginkan.

Konversi tipe data dilakukan pada atribut release_date dan rating menggunakan properti casts. Release_date dikonversi menjadi tipe date, sementara rating dikonversi menjadi string. Hal ini memastikan bahwa data yang disimpan dan diambil dari database memiliki tipe data yang sesuai.

Relasi antara model Movie dan Genre didefinisikan menggunakan metode genres. Relasi ini menggunakan hubungan many-to-many yang diimplementasikan melalui tabel pivot 'movie_genre'. Metode genres memungkinkan pengambilan data genre yang terkait dengan sebuah film.

Metode getFormattedDurationAttribute berfungsi untuk mengubah durasi film yang disimpan dalam bentuk menit menjadi format yang lebih mudah dibaca, yaitu jam dan menit. Metode ini memanfaatkan integer division dan modulus untuk menghitung jam dan menit dari total durasi dalam menit.

b. Models/Genre

```
namespace App\Models;

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

class Genre extends Model
{
    use HasFactory;

    protected $fillable = ['name'];

    public function movies()
    {
        return $this->belongsToMany(Movie::class, 'movie_genre');
    }
}
```

Model Genre dalam proyek ini berfungsi sebagai representasi dari entitas genre dalam database. Model ini menggunakan trait HasFactory untuk memanfaatkan fitur factory

dalam pembuatan model selama pengujian. Atribut yang dapat diisi (fillable) hanya terdiri dari satu properti yaitu 'name', yang memastikan bahwa hanya nama genre yang dapat diisi secara massal.

Relasi antara model Genre dan Movie didefinisikan menggunakan metode movies. Relasi ini menggunakan hubungan many-to-many yang diimplementasikan melalui tabel pivot 'movie_genre'. Metode movies memungkinkan pengambilan data film yang terkait dengan sebuah genre.

Model Genre ini berperan penting dalam pengelolaan data genre dalam aplikasi, memastikan setiap genre dapat dikaitkan dengan satu atau lebih film. Hal ini memungkinkan pengelompokan dan pencarian film berdasarkan genre, serta memudahkan dalam pengelolaan kategori film di dalam aplikasi.

c. Controllers/MovieController

```
class MovieController extends Controller
{
    public function index()
    {
        $movies = Movie::all();
        return view('movies.index', compact('movies'));
    }

    public function create()
    {
        $genres = Genre::all();
        return view('movies.create', compact('genres'));
    }

    public function store(Request $request)
    {
        $data = $request->validate([
            'title' => 'required|string|max:255',
            'image' => 'nullable|string|max:255',
            'image_file' => 'nullable|image|mimes:jpeg,png,jpg,gif|max:2048',
            'rating' => 'nullable|numeric|between:0,10',
            'production' => 'required|string|max:255',
            'director' => 'required|string|max:255',
            'release_date' => 'required|date',
            'age_restriction' => 'required|string|max:20',
            'duration' => 'required|integer|min:1',
            'synopsis' => 'required|string',
            'status' => 'required|in:available,unavailable,coming_soon',
            'genres' => 'required|array',
            'genres.*' => 'exists:genres,id',
        ]);

        if ($request->hasFile('image_file')) {
            $imagePath = $request->file('image_file')->store('movies', 'public');
            $data['image'] = $imagePath;
        } elseif ($request->image) {
            $data['image'] = $request->image;
        }

        if ($request->status == 'coming_soon' && empty($request->rating)) {
            $data['rating'] = 'Not Rated';
        }
    }
}
```

```

public function update(Request $request, Movie $movie)
{
    $data = $request->validate([
        'title' => 'required|string|max:255',
        'image' => 'nullable|string|max:255',
        'image_file' => 'nullable|image|mimes:jpeg,png,jpg,gif|max:2048',
        'rating' => 'nullable|numeric|between:0,10',
        'production' => 'required|string|max:255',
        'director' => 'required|string|max:255',
        'release_date' => 'required|date',
        'age_restriction' => 'required|string|max:20',
        'duration' => 'required|integer|min:1',
        'synopsis' => 'required|string',
        'status' => 'required|in:available,unavailable,coming_soon',
        'genres' => 'required|array',
        'genres.*' => 'exists:genres,id',
    ]);

    if ($request->hasFile('image_file')) {
        if ($movie->image && Storage::disk('public')->exists($movie->image)) {
            Storage::disk('public')->delete($movie->image);
        }
        $imagePath = $request->file('image_file')->store('movies', 'public');
        $data['image'] = $imagePath;
    }

    $data['rating'] = $data['rating'] ?? 'Not Rated';

    $movie->update($data);
    $movie->genres()->sync($data['genres']);

    return redirect()->route('movies.index')->with('success', 'Movie updated successfully.');
```

```

public function destroy(Movie $movie)
{
    // Hapus gambar jika ada
    if ($movie->image && Storage::disk('public')->exists($movie->image)) {
        Storage::disk('public')->delete($movie->image);
    }
    $movie->delete();
    return redirect()->route('movies.index')->with('success', 'Movie deleted successfully.');
```

MovieController berfungsi sebagai pengelola logika bisnis terkait operasi CRUD (Create, Read, Update, Delete) pada entitas Movie.

Method index mengakses semua data film dari database dan menampilkannya di view 'movies.index' dengan menggunakan metode compact untuk mengirim data.

Method create mengakses semua genre dan menampilkannya di view 'movies.create', menyediakan data genre untuk ditampilkan dalam form pembuatan film baru.

Method store berfungsi untuk menyimpan data film baru. Pertama, data yang diterima dari form validasi menggunakan validate. Jika ada file gambar yang diunggah, gambar tersebut disimpan di direktori storage/app/public/movies dan path-nya disimpan di kolom image. Jika tidak ada rating yang diisi dan statusnya 'coming_soon', maka rating diatur ke 'Not Rated'. Data yang telah diproses kemudian disimpan dalam

database menggunakan `Movie::create`, dan genre yang terkait disinkronkan dengan menggunakan `genres()->sync`.

Method `show` digunakan untuk menampilkan detail sebuah film berdasarkan ID-nya, menampilkan data di view `'movies.show'`.

Method `edit` mengakses data film yang akan diedit beserta genre-nya, kemudian menampilkannya di view `'movies.edit'` untuk diedit oleh pengguna.

Method `update` meng-update data film yang sudah ada. Data yang diterima dari form divalidasi terlebih dahulu. Jika ada file gambar baru yang diunggah, gambar lama akan dihapus dan gambar baru disimpan, lalu path-nya diperbarui. Jika rating tidak diisi, maka rating diatur ke `'Not Rated'`. Data yang telah diproses kemudian di-update di database, dan genre yang terkait disinkronkan kembali.

Method `destroy` menghapus data film dari database. Jika film memiliki file gambar, file tersebut dihapus dari storage terlebih dahulu sebelum data film dihapus dari database.

Relasi `genres()` pada model `Movie` dan `Genre` mengindikasikan hubungan many-to-many antara film dan genre, memanfaatkan tabel pivot `'movie_genre'` untuk mengelola relasi tersebut.

Metode `getFormattedDurationAttribute` di model `Movie` memformat durasi film dari menit ke dalam format jam dan menit, memudahkan pembacaannya oleh pengguna.

d. Controllers/PageController

```
class PageController extends Controller
{
  public function home()
  {
    $bestAvailMovies = Movie::where('status', 'available')
      ->where('release_date', '>=', '2015-01-01')
      ->orderBy('rating', 'desc')
      ->take(4)
      ->get();

    $extraordinaryMovies = Movie::where('status', 'available')
      ->orderBy('release_date', 'desc')
      ->take(4)
      ->get();

    $shortestMovies = Movie::whereIn('status', ['available', 'unavailable'])
      ->orderBy('duration', 'asc')
      ->take(4)
      ->get();

    $longestMovies = Movie::whereIn('status', ['available', 'unavailable'])
      ->orderBy('duration', 'desc')
      ->take(4)
      ->get();

    $soldFashionedMovies = Movie::orderBy('release_date', 'asc')
      ->take(4)
      ->get();

    $topMovies = Movie::orderBy('rating', 'desc')
      ->take(4)
      ->get();

    $underratedMovies = Movie::whereIn('status', ['available', 'unavailable'])
      ->orderBy('rating')
      ->take(4)
      ->get();

    $comingSoon = Movie::where('status', 'coming_soon')
      ->take(4)
      ->get();

    return view('home', compact('bestAvailMovies', 'extraordinaryMovies', 'shortestMovies',
  })
}
```

PageController bertanggung jawab untuk menampilkan halaman utama dengan berbagai kategori film yang disusun berdasarkan kriteria tertentu.

Method home mengumpulkan berbagai kategori film dari database dan mengirimkannya ke view 'home' untuk ditampilkan. Berikut adalah penjelasan singkat mengenai setiap kategori yang dikumpulkan:

1. Best Available Movies:

- Mengambil film dengan status 'available' yang dirilis setelah atau pada 1 Januari 2015.
- Mengurutkan film berdasarkan rating tertinggi.
- Mengambil empat film terbaik.

2. Extraordinary Movies:

- Mengambil film dengan status 'available'.
- Mengurutkan film berdasarkan tanggal rilis terbaru.
- Mengambil empat film terbaru yang luar biasa.

3. Longest Movies:

- Mengambil film dengan status 'available' atau 'unavailable'.
- Mengurutkan film berdasarkan durasi terpanjang.
- Mengambil empat film dengan durasi terpanjang.

4. Old Fashioned Movies:

- Mengurutkan semua film berdasarkan tanggal rilis tertua.
- Mengambil empat film tertua.

5. Top Movies:

- Mengurutkan semua film berdasarkan rating tertinggi.
- Mengambil empat film terbaik berdasarkan rating.

6. Underrated Movies:

- Mengambil film dengan status 'available' atau 'unavailable'.
- Mengurutkan film berdasarkan rating terendah.
- Mengambil empat film dengan rating terendah.

7. Coming Soon:

- Mengambil film dengan status 'coming_soon'.
- Mengambil empat film yang akan datang.

8. Shortest Movies:

- Mengambil film dengan status 'available' atau 'unavailable'.
- Mengurutkan film berdasarkan durasi terpendek.
- Mengambil empat film dengan durasi terpendek.

Setelah semua kategori film dikumpulkan, metode ini mengirimkan data ke view 'home' menggunakan metode compact untuk menyederhanakan passing data.

e. Migrations/movies_table

```
return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('movies', function (Blueprint $table) {
            $table->id();
            $table->string('title');
            $table->string('image')->nullable();
            $table->decimal('rating', 3, 1)->nullable();
            $table->string('production');
            $table->string('director');
            $table->date('release_date');
            $table->string('age_restriction');
            $table->integer('duration');
            $table->text('synopsis');
            $table->enum('status', ['available', 'unavailable', 'coming_soon']);
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('movies');
    }
};
```

Migrasi ini bertujuan untuk membuat tabel movies dalam database yang akan menyimpan informasi tentang film. Berikut adalah penjelasan mengenai struktur tabel movies:

- id: Kolom ini berfungsi sebagai primary key dan akan menyimpan nilai unik yang dihasilkan secara otomatis untuk setiap film.
- title: Kolom ini menyimpan judul film dan bertipe string.
- image: Kolom ini menyimpan path ke gambar poster film dan bersifat nullable, artinya kolom ini boleh tidak diisi.
- rating: Kolom ini menyimpan rating film dalam format desimal dengan skala 3,1 dan bersifat nullable.
- production: Kolom ini menyimpan nama perusahaan produksi film dan bertipe string.

- director: Kolom ini menyimpan nama sutradara film dan bertipe string.
- release_date: Kolom ini menyimpan tanggal rilis film dan bertipe date.
- age_restriction: Kolom ini menyimpan batasan usia untuk film tersebut dan bertipe string.
- duration: Kolom ini menyimpan durasi film dalam menit dan bertipe integer.
- synopsis: Kolom ini menyimpan ringkasan cerita film dan bertipe text.
- status: Kolom ini menyimpan status ketersediaan film dan memiliki nilai enum yang dapat berupa 'available', 'unavailable', atau 'coming_soon'.
- timestamps: Kolom ini otomatis menyimpan informasi waktu pembuatan dan pembaruan data.

Bagian up dari migrasi ini berfungsi untuk membuat tabel ketika migrasi dijalankan, sedangkan bagian down berfungsi untuk menghapus tabel tersebut ketika migrasi dibatalkan.

f. Migrations/genre_table

```
return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('genres', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('genres');
    }
};
```

Migrasi ini bertujuan untuk membuat tabel genres dalam database yang akan menyimpan informasi tentang genre film. Berikut adalah penjelasan mengenai struktur tabel genres:

- id: Kolom ini berfungsi sebagai primary key dan akan menyimpan nilai unik yang dihasilkan secara otomatis untuk setiap genre.
- name: Kolom ini menyimpan nama genre dan bertipe string.
- timestamps: Kolom ini secara otomatis menyimpan informasi waktu pembuatan dan pembaruan data.

Bagian up dari migrasi ini berfungsi untuk membuat tabel ketika migrasi dijalankan, sedangkan bagian down berfungsi untuk menghapus tabel tersebut ketika migrasi dibatalkan.

Dengan demikian, tabel genres akan digunakan untuk menyimpan berbagai jenis genre film yang kemudian dapat dihubungkan dengan tabel movies melalui tabel pivot untuk mengelola hubungan many-to-many antara film dan genre.

g. Migrations/movie_genre_table

```
return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('movie_genre', function (Blueprint $table) {
            $table->id();
            $table->foreignId('movie_id')->constrained()->onDelete('cascade');
            $table->foreignId('genre_id')->constrained()->onDelete('cascade');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('movie_genre');
    }
};
```

Migrasi ini membuat tabel movie_genre yang bertujuan untuk menyimpan data hubungan antara film dan genre. Struktur tabel ini mencakup kolom id sebagai primary key, movie_id dan genre_id sebagai foreign key yang merujuk ke tabel

movies dan genres. Relasi ini memastikan bahwa setiap film dapat dikaitkan dengan beberapa genre dan sebaliknya. Ketika baris dalam tabel movies atau genres dihapus, baris yang sesuai dalam tabel movie_genre juga akan dihapus berkat fitur onDelete('cascade'). Tabel ini juga menyertakan kolom timestamps untuk mencatat waktu pembuatan dan pembaruan setiap hubungan.

h. Routes/web

```
<?php

use Illuminate\Support\Facades\Route;
use App\Http\Controllers\MovieController;
use App\Http\Controllers\PageController;

Route::get('/', [PageController::class, 'home'])->name('home');
Route::resource('movies', MovieController::class);
```

Pada baris pertama, file ini memuat kelas Route dari Laravel serta dua controller: MovieController dan PageController. Kedua controller ini menangani logika untuk berbagai rute dalam aplikasi.

Rute pertama adalah Route::get('/', [PageController::class, 'home'])->name('home');. Rute ini mendefinisikan bahwa permintaan GET ke URL root ('/') akan ditangani oleh metode home dalam PageController. Metode ini mengembalikan tampilan utama atau halaman beranda aplikasi dan memberi nama rute tersebut sebagai 'home'.

Rute kedua menggunakan Route::resource('movies', MovieController::class);. Rute ini mendefinisikan sekumpulan rute RESTful untuk resource 'movies'. Dengan satu baris ini, Laravel secara otomatis menghasilkan rute-rute standar untuk berbagai operasi CRUD (Create, Read, Update, Delete) pada model Movie. Rute-rute ini meliputi:

- GET /movies: Menampilkan daftar semua film.
- GET /movies/create: Menampilkan formulir untuk membuat film baru.
- POST /movies: Menyimpan film baru ke database.
- GET /movies/{movie}: Menampilkan detail film tertentu.

- GET /movies/{movie}/edit: Menampilkan formulir untuk mengedit film tertentu.
- PUT/PATCH /movies/{movie}: Memperbarui data film tertentu.
- DELETE /movies/{movie}: Menghapus film tertentu dari database.

Dengan menggunakan rute resource, pengelolaan film dalam aplikasi menjadi lebih terstruktur dan mengikuti pola konvensi yang disarankan oleh Laravel. Ini membuat kode lebih rapi dan mudah dipahami, serta memudahkan dalam pemeliharaan dan pengembangan lebih lanjut.

i. Views/Card

```
<div>
  <h2 class="h2-home">{{ $title }}</h2>
  <div class="row mb-5">
    @foreach($movies as $movie)
      <div class="col-md-3">
        <div class="card">
          title }}" class="card-img-top" width="100">
          <!-- title }}" -->
          <div class="card-body">
            <h5 class="card-title">
              {{ $movie->title }}
              <span class="float-right">
                <i class="fas fa-star text-warning"></i> {{ $movie->rating }}
              </span>
            </h5>
            <p class="card-text">
              @foreach($movie->genres as $genre)
                <span class="badge badge-{{ $badgeColor }}">{{ $genre->name }}</span>
              @endforeach
            </p>
            <p class="card-text">{{ $movie->synopsis }}</p>
          </div>
        </div>
      </div>
    @endforeach
  </div>
</div>
```

Bagian views ini bertanggung jawab untuk menampilkan daftar film dalam bentuk kartu pada halaman web. Tampilan ini dibangun menggunakan HTML yang dipadukan dengan sintaks Blade, template engine bawaan Laravel.

Dalam elemen <div>, judul halaman ditampilkan menggunakan elemen <h2> yang mengambil nilai dari variabel Blade \$title.

Selanjutnya, terdapat elemen <div> dengan kelas row yang mengatur tata letak grid dari kartu-kartu film, dengan setiap iterasi film dalam koleksi \$movies ditampilkan

dalam elemen `<div class="col-md-3">`, yang membagi halaman menjadi kolom-kolom sesuai dengan grid sistem Bootstrap.

Setiap film ditampilkan dalam sebuah kartu yang terdiri dari elemen-elemen HTML berikut:

- `<div class="card">`: Menyusun tampilan kartu untuk setiap film.
- `title }}" class="card-img-top" width="100">`: Menampilkan poster film menggunakan elemen ``. Gambar diambil dari folder storage dan dipastikan memiliki deskripsi alternatif (alt) yang sesuai dengan judul film.
- `<div class="card-body">`: Mengelompokkan isi kartu yang terdiri dari:
- `<h5 class="card-title">`: Menampilkan judul film dan rating di sebelah kanan menggunakan ikon bintang berwarna kuning.
- `<p class="card-text">`: Menampilkan genre-genre film dalam bentuk lencana (badge) Bootstrap. Setiap genre diambil dari relasi genre film dan diberi kelas CSS dinamis menggunakan variabel `$badgeColor`.
- `<p class="card-text">`: Menampilkan sinopsis film.

Blade digunakan untuk melakukan iterasi melalui koleksi film dan genre serta mengeluarkan data dinamis yang sesuai. Dengan penggunaan kelas Bootstrap seperti `card`, `card-body`, `card-img-top`, dan `badge`, tampilan menjadi responsif dan estetis. Seluruh kode ini disusun dalam struktur grid sehingga setiap film ditampilkan dalam kolom yang rapi dan mudah diakses.

j. Views/app (navbar)

```
<body>
<nav class="navbar navbar-expand-lg navbar-dark bg-dark">
  <a class="navbar-brand" href="{{ route('home') }}">
    
    Cinema
  </a>
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false"
    <span class="navbar-toggler-icon"></span>
  </button>
  <div class="collapse navbar-collapse" id="navbarNav">
    <ul class="navbar-nav">
      <li class="nav-item {{ Request::routeIs('home') ? 'active' : '' }}">
        <a class="nav-link" href="{{ route('home') }}">
          
          Home
        </a>
      </li>
      <li class="nav-item {{ Request::routeIs('movies.index') ? 'active' : '' }}">
        <a class="nav-link" href="{{ route('movies.index') }}">
          
          Movies
        </a>
      </li>
      <li class="nav-item {{ Request::routeIs('movies.create') ? 'active' : '' }}">
        <a class="nav-link" href="{{ route('movies.create') }}">
          
          Add Movies
        </a>
      </li>
    </ul>
  </div>
</body>
```

Bagian `<body>` dimulai dengan navbar yang dibangun menggunakan komponen Bootstrap. Navbar ini memiliki merek "Cinema" dengan ikon, tombol toggle untuk navigasi yang dapat dibuka pada layar kecil, dan tiga item navigasi (Home, Movies, Add Movies) yang diaktifkan berdasarkan rute saat ini.

Selanjutnya, terdapat elemen `<div class="container mt-4">` yang berfungsi sebagai kontainer utama untuk konten halaman, di mana konten dinamis akan dimasukkan menggunakan directive Blade `@yield('content')`.

Bagian bawah dokumen memuat skrip JavaScript untuk mendukung fitur interaktif Bootstrap, termasuk jQuery, Popper.js, dan Bootstrap's JavaScript bundle. Skrip ini diambil dari CDN untuk memastikan pemuatan cepat dan efisien.

k. Views/home (home)

```
@section('content')
<div class="container">
  <div class="jumbotron text-center custom-jumbotron">
    <h1 class="display-4">Cinema</h1>
    <p class="lead">Discover the best movies and enjoy the cinema experience.</p>
  </div>
</div>

<div class="container mt-5">
  <div class="row">
    <div class="col-md-12 text-center mb-4">
      <h1 class="text-white text-uppercase">Movies to Explore</h1>
    </div>
  </div>

  <div class="row mt-3">
    <div class="col-md-12">
      @component('components.movie-card', [
        'title' => 'Shortest-Length Movies to Savor',
        'movies' => $shortestMovies,
        'badgeColor' => 'danger'
      ])@endcomponent
    </div>
  </div>

  <div class="row mt-5">
    <div class="col-md-12">
      @component('components.movie-card', [
        'title' => 'Long-Duration Movies to Discover',
        'movies' => $longestMovies,
        'badgeColor' => 'success'
      ])@endcomponent
    </div>
  </div>

  <div class="row mt-5">
    <div class="col-md-12">
      @component('components.movie-card', [
        'title' => 'Even If It\'s Underrated, Maybe You Like These Movies?',
        'movies' => $underratedMovies,
        'badgeColor' => 'warning'
      ])@endcomponent
    </div>
  </div>
</div>
```

Halaman ini merupakan halaman utama aplikasi Cinema. Dengan menggunakan template Blade, halaman ini dibuat dinamis dengan berbagai komponen yang menampilkan informasi film-film tertentu.

Bagian atas halaman menampilkan jumbotron dengan judul "Cinema" dan deskripsi singkat "Discover the best movies and enjoy the cinema experience."

Selanjutnya, terdapat beberapa blok konten yang menampilkan daftar film dalam kategori-kategori tertentu:

- Shortest-Length Movies to Savor: Menampilkan daftar film dengan durasi terpendek untuk dinikmati.
- Long-Duration Movies to Discover: Menampilkan daftar film dengan durasi terpanjang untuk ditemukan.

- Even If It's Underrated, Maybe You Like These Movies?: Menampilkan daftar film yang mungkin diremehkan tapi mungkin Anda akan menyukainya.
- Extraordinary Smash Hit Movies: Menampilkan daftar film yang luar biasa dan sangat populer.
- Top Movies of All Time: Menampilkan daftar film teratas sepanjang masa.
- Old Fashioned Movies: Menampilkan daftar film klasik.
- Coming Soon Movies: Menampilkan daftar film yang akan datang.

Setiap blok konten tersebut menggunakan komponen movie-card yang diatur dengan judul kategori film, daftar film yang sesuai, dan warna badge untuk setiap genre film.

1. Views/index (list table)

```
@section('content')
<div class="container">
  <h1 style="color:snow; text-align: center; margin-bottom: 30px;">Movies List</h1>
  <div class="table-responsive">
    <table class="table table-bordered table-hover table-striped">
      <thead class="thead-light" style="text-align: center;">
        <tr>
          <th>Title</th>
          <th>Poster</th>
          <th>Rating</th>
          <th>Producer</th>
          <th>Director</th>
          <th>Release Date</th>
          <th>Age Restriction</th>
          <th>Duration</th>
          <th>Synopsis</th>
          <th>Genres</th>
          <th>Status</th>
          <th>Actions</th>
        </tr>
      </thead>
      <tbody>
        @foreach($movies as $movie)
          <tr style="color: snow;">
            <td>{{ $movie->title }}</td>
            <td>
              title }}" width="100">
            </td>
            <td>
              <i class="fas fa-star text-warning"></i>
              {{ $movie->rating ?? 'Not Rated' }}
            </td>
            <td>{{ $movie->production }}</td>
            <td>{{ $movie->director }}</td>
            <td>{{ $movie->release_date->format('M Y') }}</td>
            <td>{{ $movie->age_restriction }}</td>
            <td>{{ $movie->formatted_duration }}</td>
            <td>{{ Str::limit($movie->synopsis, 200) }}</td>
            <td>
              @foreach($movie->genres as $genre)
                <span class="badge badge-primary">{{ $genre->name }}</span>
              @endforeach
            </td>
          </tr>
        @endforeach
      </tbody>
    </table>
  </div>
</div>
```

Halaman ini menampilkan daftar film dalam bentuk tabel yang responsif. Setiap baris tabel menampilkan informasi tentang satu film, seperti judul, poster, rating, produser, sutradara, tanggal rilis, batasan usia, durasi, sinopsis, genre, status, dan aksi yang dapat dilakukan pengguna.

- Judul: Menampilkan judul film.
- Poster: Menampilkan poster film dengan lebar maksimum 100 piksel.
- Rating: Menampilkan rating film dengan ikon bintang dan nilai rating jika tersedia. Jika tidak ada rating, ditampilkan "Not Rated".
- Produser: Menampilkan nama produser film.
- Sutradara: Menampilkan nama sutradara film.
- Tanggal Rilis: Menampilkan tanggal rilis film dalam format bulan dan tahun.
- Batasan Usia: Menampilkan batasan usia film.
- Durasi: Menampilkan durasi film dalam format jam dan menit.
- Sinopsis: Menampilkan sinopsis film yang dipotong menjadi maksimum 200 karakter.
- Genre: Menampilkan genre film dalam bentuk badge.
- Status: Menampilkan status film, seperti "Available", "Unavailable", atau "Coming Soon", dengan warna badge yang sesuai.
- Aksi: Menampilkan tombol "Edit" dan "Delete" untuk setiap film. Tombol "Edit" mengarahkan pengguna ke halaman edit film, sedangkan tombol "Delete" menghapus film setelah konfirmasi.

Di bagian bawah halaman, terdapat script JavaScript yang menangani peristiwa klik pada tombol "Delete". Ketika pengguna mengklik tombol "Delete", akan muncul konfirmasi untuk memastikan pengguna benar-benar ingin menghapus film tersebut. Jika pengguna mengonfirmasi, formulir penghapusan akan disubmit.

m. Views/edit

```
<h1>Edit Movie</h1>
<form action="{{ route('movies.update', $movie) }}" method="POST">
    @csrf
    @method('PUT')
    <div class="form-group">
        <label for="title">Title</label>
        <input type="text" class="form-control" id="title" name="title" value="{{ $movie->title }}" required>
    </div>
    <div class="form-group">
        <label for="image">Image (URL or Upload File)</label>
        <input type="text" class="form-control" id="image" name="image" value="{{ old('image', $movie->image) }}">
        <input type="file" class="form-control-file mt-2" id="image_file" name="image_file" accept="image/*">
    </div>
    <div class="form-group">
        <label for="rating">Rating</label>
        <input type="number" class="form-control" id="rating" name="rating" step="0.1" min="0" max="10" value="{{ $movie->rating }}">
    </div>
    <div class="form-group">
        <label for="production">Produced By</label>
        <input type="text" class="form-control" id="production" name="production" value="{{ $movie->production }}">
    </div>
    <div class="form-group">
        <label for="director">Directed By</label>
        <input type="text" class="form-control" id="director" name="director" value="{{ $movie->director }}">
    </div>
    <div class="form-group">
        <label for="release_date">Release Date</label>
        <input type="date" class="form-control" id="release_date" name="release_date" value="{{ $movie->release_date }}">
    </div>
    <div class="form-group">
        <label for="age_restriction">Age Restriction</label>
        <input type="text" class="form-control" id="age_restriction" name="age_restriction" value="{{ $movie->age_restriction }}">
    </div>
    <div class="form-group">
        <label for="duration">Duration (Minutes)</label>
        <input type="number" class="form-control" id="duration" name="duration" value="{{ $movie->duration }}">
    </div>
</form>
```

Halaman ini adalah formulir edit untuk memperbarui informasi film yang ada dalam database. Pengguna diminta untuk mengisi atau memperbarui detail film seperti judul, gambar poster, rating, produser, sutradara, tanggal rilis, batasan usia, durasi, sinopsis, status, dan genre. Setelah mengisi formulir, pengguna dapat mengklik tombol "Update Movie" untuk menyimpan perubahan.

n. Views/create (add)

```
<div class="container mt-2 container-form">
  <h1>Add Movies</h1>
  <form action="{{ route('movies.store') }}" method="POST" enctype="multipart/form-data">
    @csrf
    <div class="form-group">
      <label for="title">Title</label>
      <input type="text" class="form-control" id="title" name="title" placeholder="Enter the Title of the Movie..." required>
    </div>
    <div class="form-group">
      <label for="image">Image (URL or Upload File)</label>
      <input type="text" class="form-control" id="image" name="image" value="{{ old('image') }}" placeholder="Enter Image URL.">
      <input type="file" class="form-control-file mt-2" id="image_file" name="image_file" accept="image/*">
    </div>
    <div class="form-group">
      <label for="rating">Rating (0-10)</label>
      <input type="number" class="form-control" id="rating" name="rating" step="0.1" min="0" max="10" placeholder="Enter the Rating">
    </div>
    <div class="form-group">
      <label for="production">Produced By</label>
      <input type="text" class="form-control" id="production" name="production" placeholder="Enter the Person Who Produced the Movie">
    </div>
    <div class="form-group">
      <label for="director">Directed By</label>
      <input type="text" class="form-control" id="director" name="director" placeholder="Enter the Person Who Directed the Movie">
    </div>
    <div class="form-group">
      <label for="release_date">Release Date</label>
      <input type="date" class="form-control" id="release_date" name="release_date" placeholder="Choose the Date When this Movie was Released">
    </div>
    <div class="form-group">
      <label for="age_restriction">Age Restriction</label>
      <input type="text" class="form-control" id="age_restriction" name="age_restriction" placeholder="Enter the Age Restriction for this Movie">
    </div>
    <div class="form-group">
      <label for="duration">Duration (Minutes)</label>
      <input type="number" class="form-control" id="duration" name="duration" placeholder="Enter the Duration of this Movie in Minutes">
    </div>
    <div class="form-group">
      <label for="synopsis">Synopsis</label>
      <textarea class="form-control" id="synopsis" name="synopsis" rows="3" placeholder="Write a Synopsis that Captures this Movie's Essence">
    </div>
  </form>
</div>
```

Halaman ini merupakan formulir untuk menambahkan film baru ke dalam database.

Halaman ini akan muncul saat user mengklik pada navbar “Add Movies” Pengguna diminta untuk mengisi detail film seperti judul, gambar poster, rating, produser, sutradara, tanggal rilis, batasan usia, durasi, sinopsis, status, dan genre. Setelah mengisi formulir, pengguna dapat mengklik tombol "Add Movie" untuk menyimpan informasi film baru tersebut.

o. css/style

```
.navbar-nav .nav-item {
  margin-right: 15px;
}

.nav-link .icon {
  margin-right: 5px;
  width: 24px;
  height: 24px;
}

.navbar-brand .icon {
  width: 24px;
  height: 24px;
  margin-right: 5px;
}

.custom-jumbotron {
  position: relative;
  background-image: url('/storage/background/background3.jpg');
  background-size: cover;
  background-position: center;
  color: white;
  padding: 100px 0;
  border-radius: 15px;
  text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.7);
}

.custom-jumbotron::before {
  content: '';
  position: absolute;
  top: 0;
  left: 0;
  width: 100%;
  height: 100%;
  background: rgba(0, 0, 0, 0.5);
  border-radius: 15px;
  z-index: 1;
}

.custom-jumbotron .display-4,
.custom-jumbotron .lead {
  position: relative;
  z-index: 2;
}

.custom-jumbotron .display-4 {
  font-size: 4rem;
  font-weight: bold;
}

.custom-jumbotron .lead {
  font-size: 1.5rem;
  margin-top: 20px;
}

body {
  background-image: url('/storage/background/background4.jpg');
  background-size: cover;
  background-attachment: fixed;
  background-position: center;
}

.container {
  max-width: 1750px;
  margin: auto;
}

.container-form {
  background-image: url('/storage/background/formbackground.jpg');
  background-size: cover;
  background-position: center;
  border-radius: 10px;
  padding: 30px;
  color: white;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
  max-width: 950px;
  margin-top: 100px;
  margin-bottom: 40px;
}

.container-form h1 {
  text-align: center;
  margin-bottom: 20px;
  color: #f1c40f;
  font-weight: bold;
}

.form-control, .form-control-file, .form-control:focus, .form-control-file:focus {
  background-color: rgba(255, 255, 255, 0.8);
  color: #000;
  border: 1px solid #ccc;
  border-radius: 5px;
}

.form-control::placeholder {
  color: #666;
}

.form-control-file::file-selector-button {
  color: #fff;
  background: #007bff;
  border: none;
  border-radius: 5px;
  padding: 5px 10px;
}

.form-control-file::file-selector-button:hover {
  background: #0056b3;
}
```

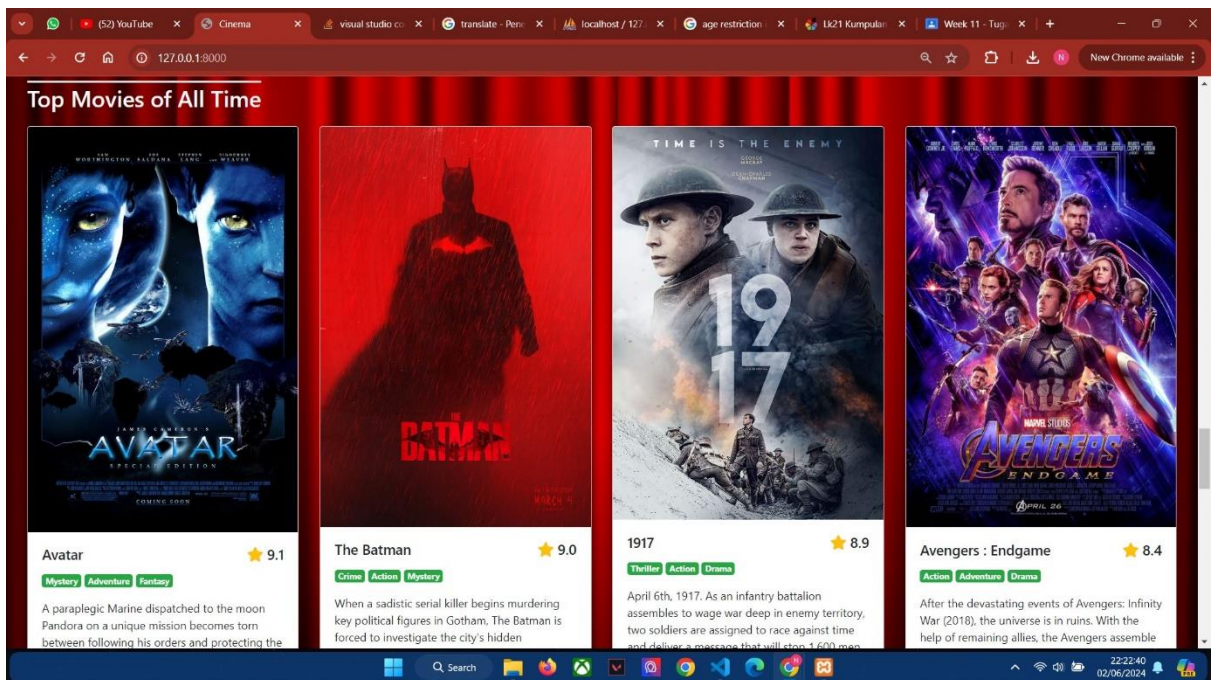
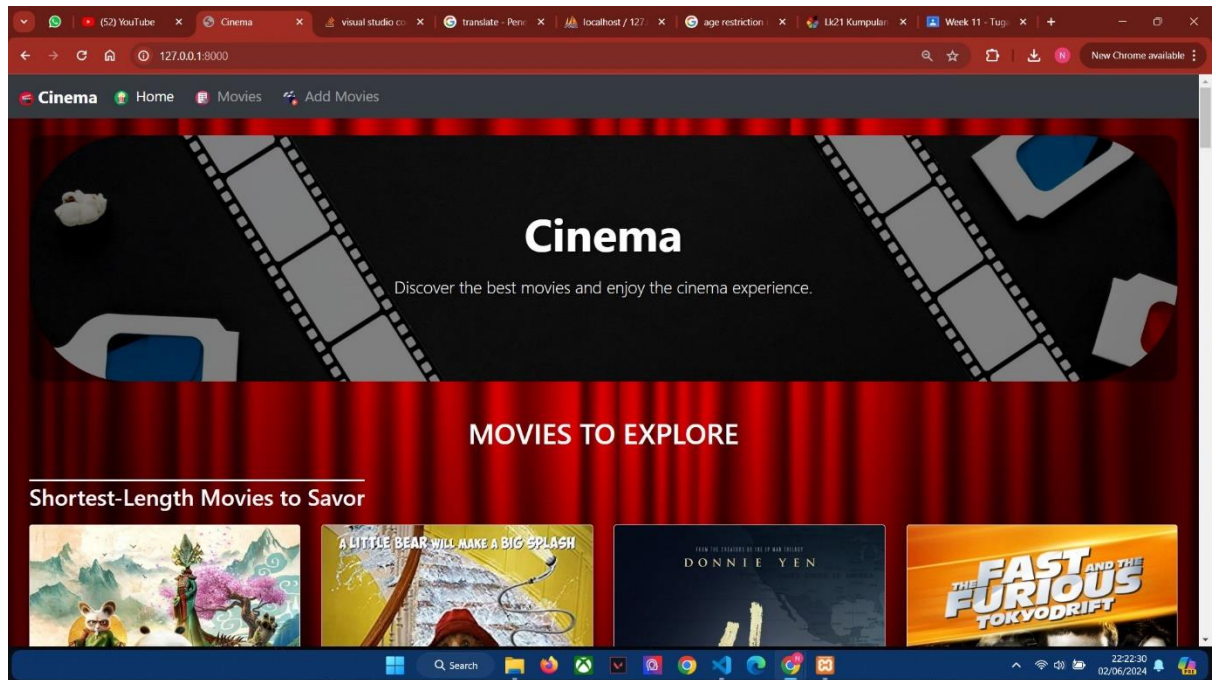
CSS ini mengatur tata letak dan tampilan elemen-elemen pada halaman web. Ini termasuk penyesuaian untuk navigasi, jumbotron, formulir, tombol, dan tabel. Warna, ukuran, dan gaya teks juga disesuaikan untuk meningkatkan estetika dan kegunaan halaman. Selain itu, terdapat transisi efek untuk beberapa elemen, seperti gambar pada kartu, untuk memberikan responsivitas visual saat interaksi pengguna.

BAB II

Analisis Output

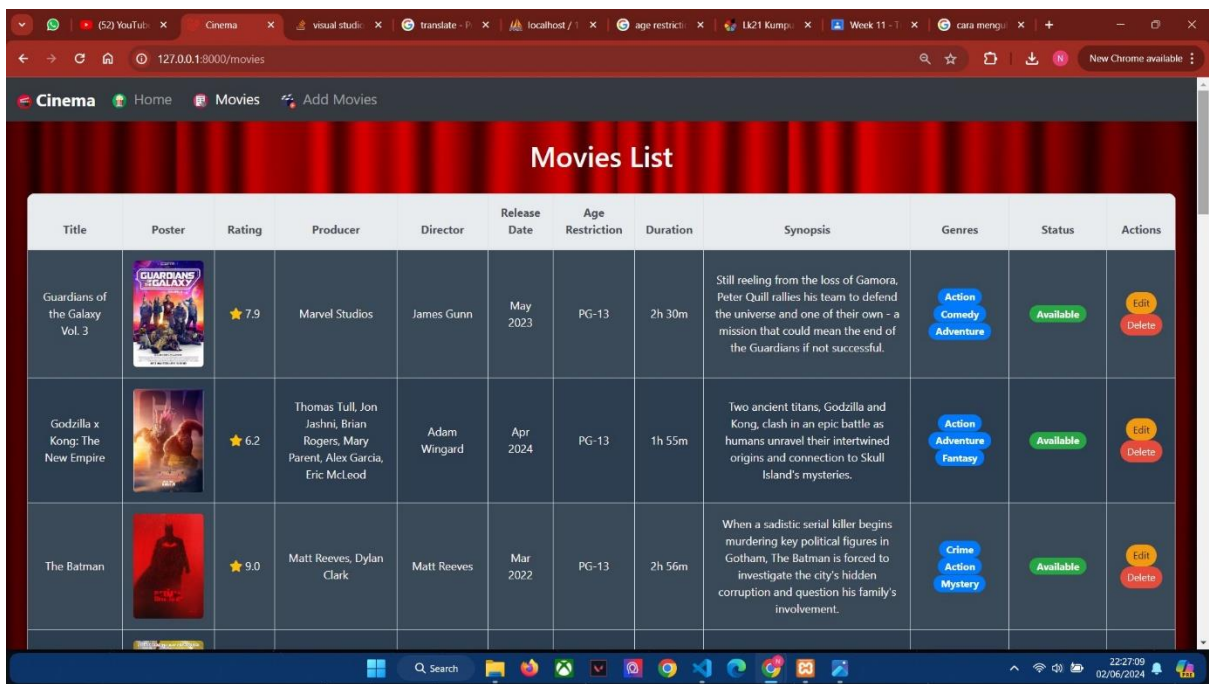
1. Tampilan Home




Berikut adalah tampilan dari home (page awal dari website)

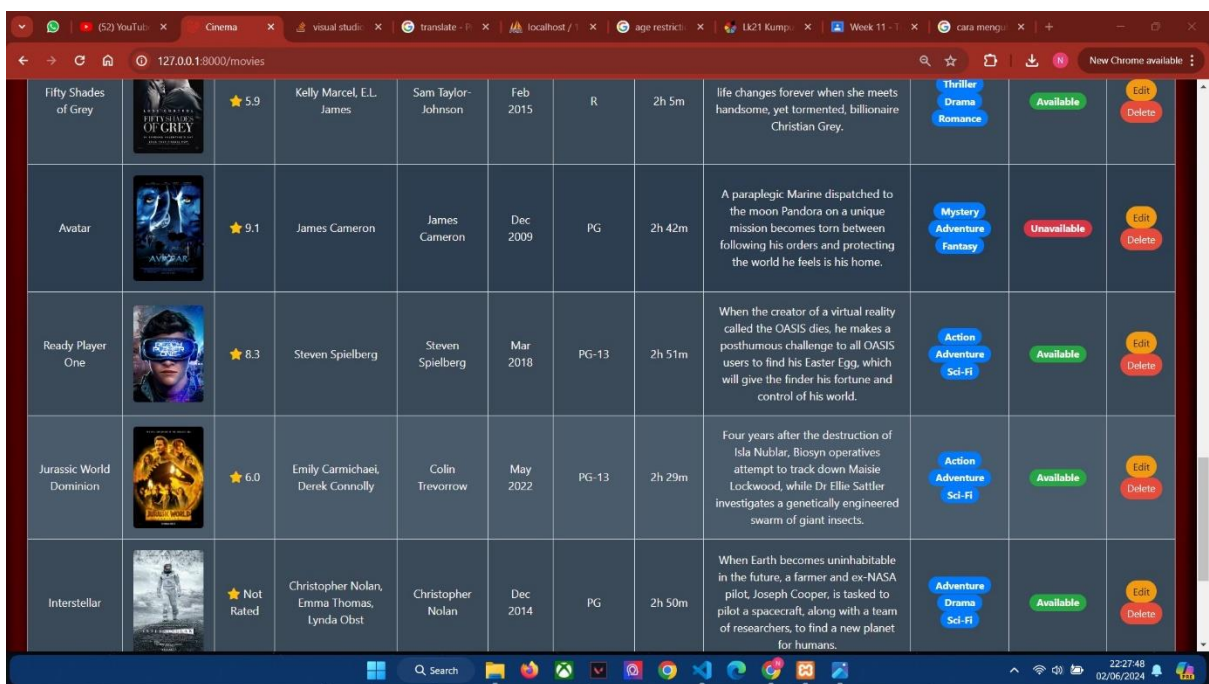







2. Tampilan Movies (List)

Pada halaman ini menampilkan list darai tiap film beserta attribut detail dari film tersebut mulai dari nama, producer, rating, durasi, dll. Terdapat juga 2 tombol pada kolom action yakni edit dan delete



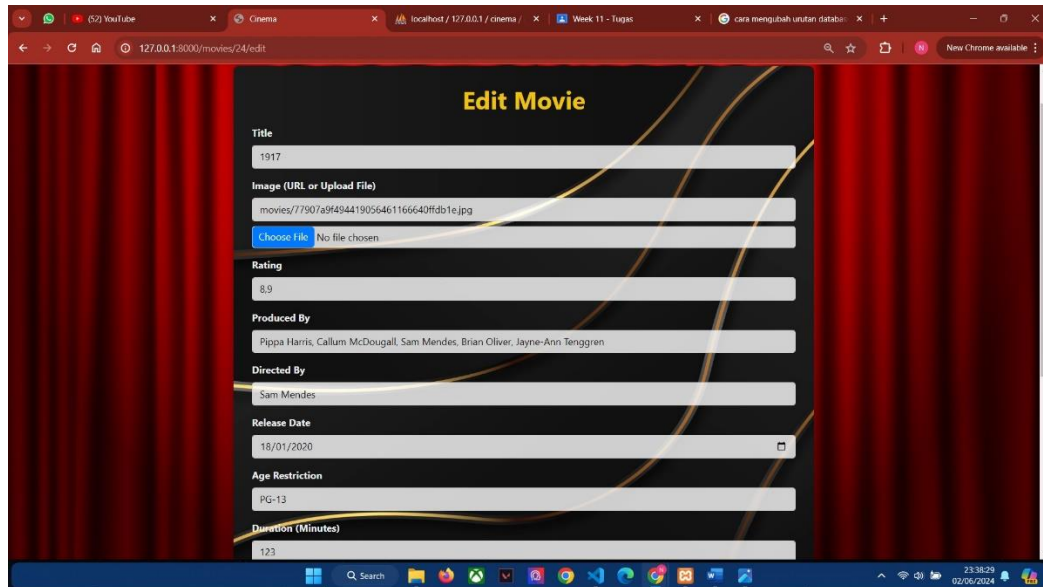
Movies List											
Title	Poster	Rating	Producer	Director	Release Date	Age Restriction	Duration	Synopsis	Genres	Status	Actions
Guardians of the Galaxy Vol. 3		★ 7.9	Marvel Studios	James Gunn	May 2023	PG-13	2h 30m	Still reeling from the loss of Gamora, Peter Quill rallies his team to defend the universe and one of their own - a mission that could mean the end of the Guardians if not successful.	Action Comedy Adventure	Available	Edit Delete
Godzilla x Kong: The New Empire		★ 6.2	Thomas Tull, Jon Jashni, Brian Rogers, Mary Parent, Alex Garcia, Eric McLeod	Adam Wingard	Apr 2024	PG-13	1h 55m	Two ancient titans, Godzilla and Kong, clash in an epic battle as humans unravel their intertwined origins and connection to Skull Island's mysteries.	Action Adventure Fantasy	Available	Edit Delete
The Batman		★ 9.0	Matt Reeves, Dylan Clark	Matt Reeves	Mar 2022	PG-13	2h 56m	When a sadistic serial killer begins murdering key political figures in Gotham, The Batman is forced to investigate the city's hidden corruption and question his family's involvement.	Crime Action Mystery	Available	Edit Delete



Fifty Shades of Grey		★ 5.9	Kelly Marcel, E.L. James	Sam Taylor-Johnson	Feb 2015	R	2h 5m	life changes forever when she meets handsome, yet tormented, billionaire Christian Grey.	Thriller Drama Romance	Available	Edit Delete
Avatar		★ 9.1	James Cameron	James Cameron	Dec 2009	PG	2h 42m	A paraplegic Marine dispatched to the moon Pandora on a unique mission becomes torn between following his orders and protecting the world he feels is his home.	Mystery Adventure Fantasy	Unavailable	Edit Delete
Ready Player One		★ 8.3	Steven Spielberg	Steven Spielberg	Mar 2018	PG-13	2h 51m	When the creator of a virtual reality called the OASIS dies, he makes a posthumous challenge to all OASIS users to find his Easter Egg, which will give the finder his fortune and control of his world.	Action Adventure Sci-Fi	Available	Edit Delete
Jurassic World Dominion		★ 6.0	Emily Carmichael, Derek Connolly	Colin Trevorrow	May 2022	PG-13	2h 29m	Four years after the destruction of Isla Nublar, Biosyn operatives attempt to track down Maisie Lockwood, while Dr Ellie Sattler investigates a genetically engineered swarm of giant insects.	Action Adventure Sci-Fi	Available	Edit Delete
Interstellar		★ Not Rated	Christopher Nolan, Emma Thomas, Lynda Obst	Christopher Nolan	Dec 2014	PG	2h 50m	When Earth becomes uninhabitable in the future, a farmer and ex-NASA pilot, Joseph Cooper, is tasked to pilot a spacecraft, along with a team of researchers, to find a new planet for humans.	Adventure Drama Sci-Fi	Available	Edit Delete

3. Tampilan Edit

Misalkan disini saya ingin mengedit data dari film 1917, maka akan menampilkan form yang sudah berisi dengan data lama, sehingga user nantinya bisa mengubah data mana yang ingin diubah tanpa harus mengulangi pengisian data dari awal. Perubahan ini juga otomatis akan tersimpan di database



Edit Movie

Title: 1917

Image (URL or Upload File): movies/7790/a9494419056461166640f0d01e.jpg
[Choose File](#) No file chosen

Rating: 8.9

Produced By: Pippa Harris, Callum McDougall, Sam Mendes, Brian Oliver, Jayne-Ann Tenggren

Directed By: Sam Mendes

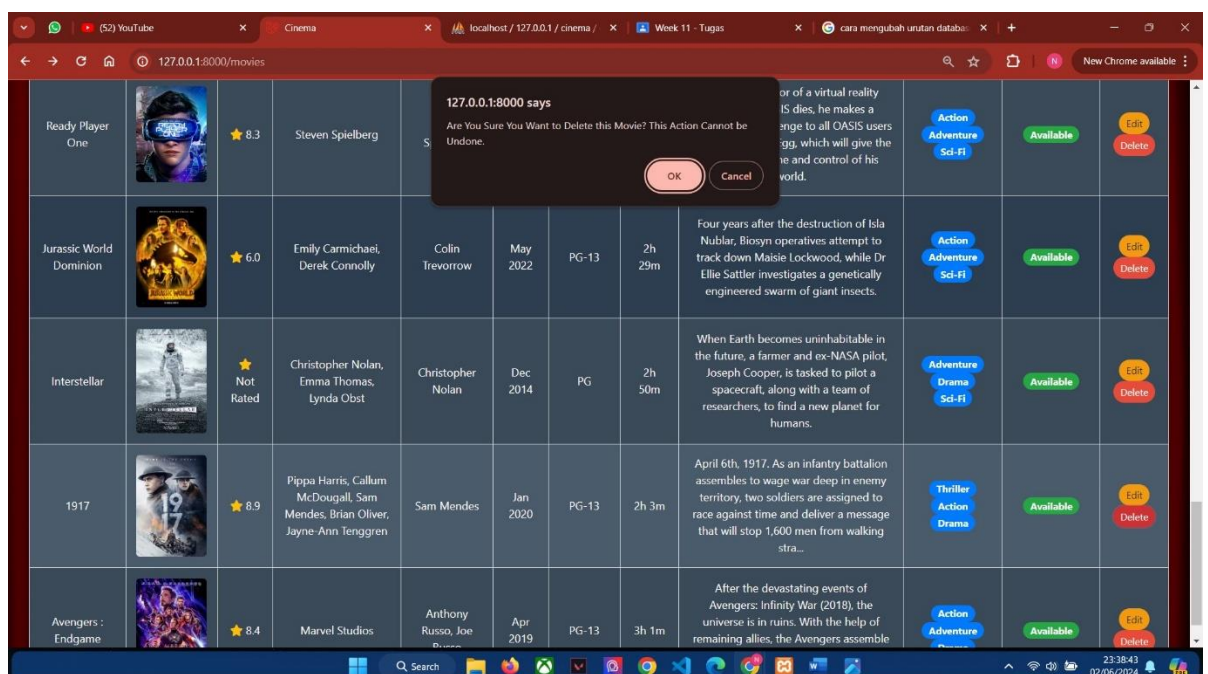
Release Date: 18/01/2020






Age Restriction: PG-13

Duration (Minutes): 123

4. Tampilan Delete

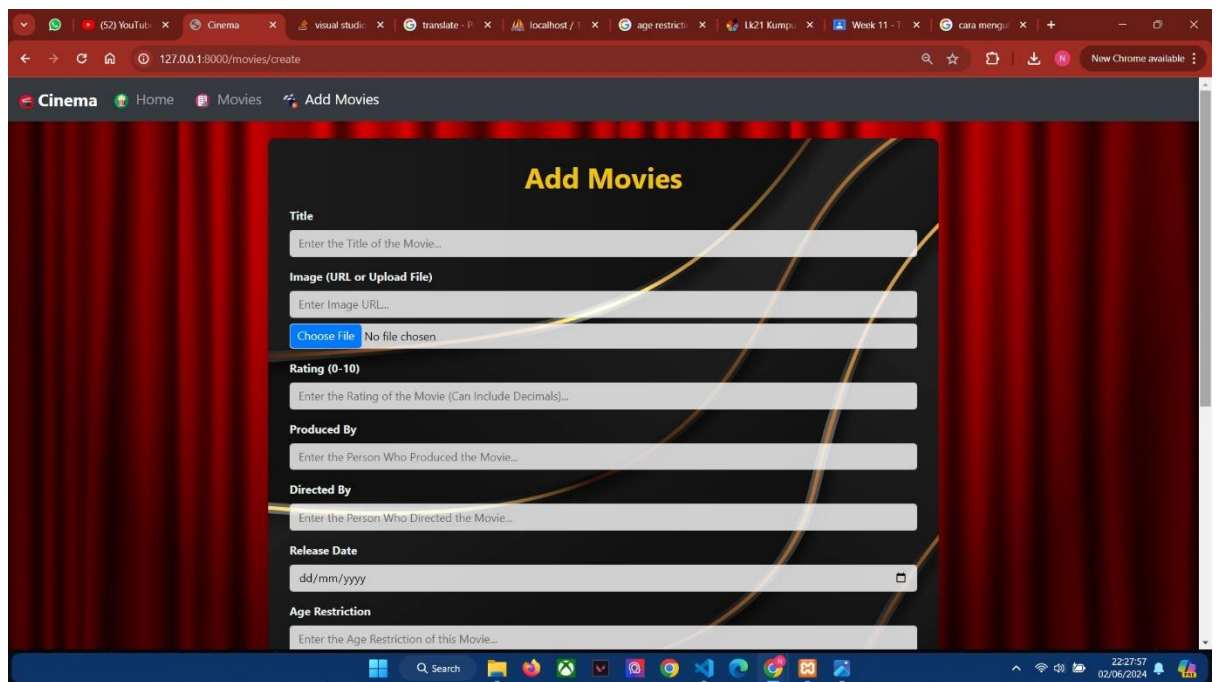
Jika diklik akan ditampilkan pesan konfirmasi seperti pada gambar untuk menghapus data yang ingin dihapus. Penghapusan ini juga otomatis akan tersimpan di database



Ready Player One		★ 8.3	Steven Spielberg						Four years after the destruction of Isla Nublar, Biosyn operatives attempt to track down Maisie Lockwood, while Dr Ellie Sattler investigates a genetically engineered swarm of giant insects.	Action Adventure Sci-Fi	Available	Edit Delete
Jurassic World Dominion		★ 6.0	Emily Carmichael, Derek Connolly	Colin Trevorrow	May 2022	PG-13	2h 29m		Four years after the destruction of Isla Nublar, Biosyn operatives attempt to track down Maisie Lockwood, while Dr Ellie Sattler investigates a genetically engineered swarm of giant insects.	Action Adventure Sci-Fi	Available	Edit Delete
Interstellar		★ Not Rated	Christopher Nolan, Emma Thomas, Lynda Obst	Christopher Nolan	Dec 2014	PG	2h 50m		When Earth becomes uninhabitable in the future, a farmer and ex-NASA pilot, Joseph Cooper, is tasked to pilot a spacecraft, along with a team of researchers, to find a new planet for humans.	Adventure Drama Sci-Fi	Available	Edit Delete
1917		★ 8.9	Pippa Harris, Callum McDougall, Sam Mendes, Brian Oliver, Jayne-Ann Tenggren	Sam Mendes	Jan 2020	PG-13	2h 3m		April 6th, 1917. As an infantry battalion assembles to wage war deep in enemy territory, two soldiers are assigned to race against time and deliver a message that will stop 1,600 men from walking straight into a deadly trap.	Thriller Action Drama	Available	Edit Delete
Avengers : Endgame		★ 8.4	Marvel Studios	Anthony Russo, Joe Russo	Apr 2019	PG-13	3h 1m		After the devastating events of Avengers: Infinity War (2018), the universe is in ruins. With the help of remaining allies, the Avengers assemble once more in order to reverse the recent events caused by Thanos' snap.	Action Adventure Sci-Fi	Available	Edit Delete

5. Tampilan Add Movies

Saat user mengklik Add Movies pada navbar akan ditampilkan form add movies seperti dibawah ini untuk menambahkan data film baru, beberapa kolom seperti title, directed by, release date, age restriction wajib ditambahkan (tidak boleh kosong). Setelah selesai mengisi akan ada tombol Add Movie untuk menambahkan data film yang kemudian jika user kembali ke halaman home ataupun list maka akan muncull film yang baru saja ditambahkan. Penambahan ini juag otomatis akan tersimpan di database



The screenshot shows a web browser window with the URL `127.0.0.1:8000/movies/create`. The page has a dark red background with a curtain-like texture. The navigation bar at the top includes 'Cinema', 'Home', 'Movies', and 'Add Movies'. The main content area is titled 'Add Movies' in yellow. The form contains the following fields:

- Title**: A text input field with the placeholder 'Enter the Title of the Movie...'.
- Image (URL or Upload File)**: A text input field with the placeholder 'Enter Image URL...'. Below it is a blue 'Choose File' button and the text 'No file chosen'.
- Rating (0-10)**: A text input field with the placeholder 'Enter the Rating of the Movie (Can Include Decimals)'.
- Produced By**: A text input field with the placeholder 'Enter the Person Who Produced the Movie...'.
- Directed By**: A text input field with the placeholder 'Enter the Person Who Directed the Movie...'.
- Release Date**: A text input field with the placeholder 'dd/mm/yyyy' and a calendar icon on the right.
- Age Restriction**: A text input field with the placeholder 'Enter the Age Restriction of this Movie...'.

The Windows taskbar at the bottom shows the time as 22:27:57 on 02/06/2024.