TUGAS JURNAL
KONSTRUKSI PERANGKAT LUNAK
MODUL IX
API DESIGN & CONSTRUCTIONN USING SWAGGER

Disusun Oleh:

Maria Nathasya Desfera Pangestu

2211104008

SE0601

Dosen Pengampu:

Yudha Islami Sulistya, S.Kom., M.Cs.

PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

Tugas Pendahuluan

1. MEMBUAT PROJECT WEB API

Berhubung cara membuat project web api berbeda-beda untuk setiap bahasa pemrograman, langkah-langkah berikut hanya berlaku apabila dilakukan dengan menggunakan .NET dan Visual Studio. Untuk IDE dan bahasa pemrograman lain, yang terpenting adalah nama project yang dibuat yaitu “modul8_NIM”.

A. Buka visual studio yang sudah terinstall dengan ASP.NET dan .NET 5.0 SDK atau setelahnya

B. Pilih New Project dan kemudian pilih ASP.NET Core Web API atau API (pastikan opsi

‘Enable OpenAPI support’ tercentang).

C. Pastikan untuk memilih .NET versi 5.0 atau yang lebih baru.

D. Masukkan nama projek “modul9_NIM”.

E. Langkah-langkah yang disertai gambar dapat dilihat pada link berikut ini (cukup dilihat

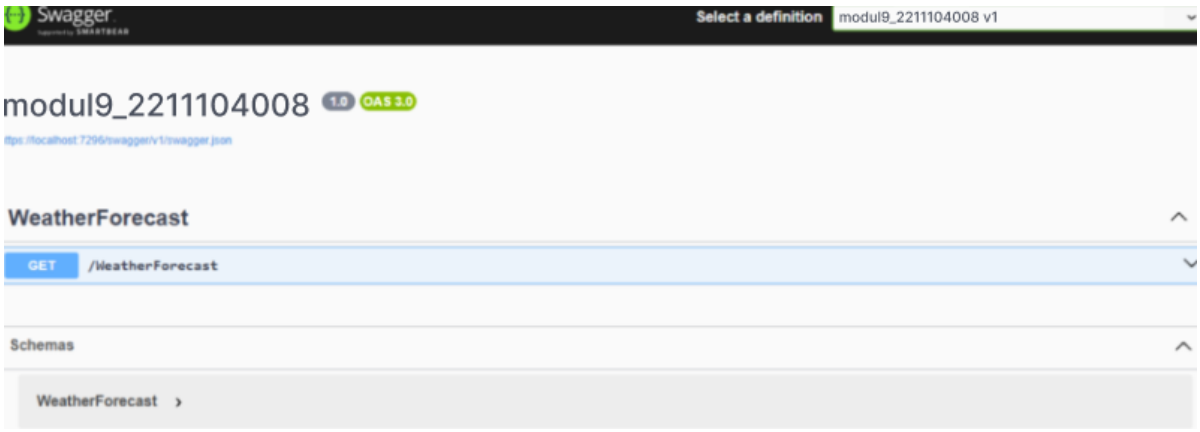
pada bagian “Create a Web API project”): https://docs.microsoft.com/en-us/aspnet/core/tutorials/min-web-api?view=aspnetcore-

6.0&tabs=visual-studio

F. Setelah project tersebut selesai dibuat, coba run programnya, dan tunggu sampai program

selesai di-compile.

Jawab



2. MELAKUKAN GIT COMMIT PADA PROJECT YANG DIBUAT

3. IMPLEMENTASI WEB API

Dari master/main branch dan class utama, buatlah program/aplikasi web API dari spesifikasi sebagai berikut ini:

A. API yang dibuat menggunakan data dari kelas Movie.

Movie
+ Title : string + Director : string + Stars : List<string> + Description: string
+ Movie()

B. API yang dibuat mempunyai lokasi sebagai berikut '/api/Movies, URL domain boleh dari port mana saja (port bebas). Dengan menggunakan swagger API tersebut dapat menerima
RESTful API dengan metoda sebagai berikut (halaman swagger dapat diakses pada https://localhost:<PORT>/swagger/index.html):

i. GET /api/Movies: mengembalikan output berupa list/array dari semua objek Movies
ii. GET /api/Movies/{id}: mengembalikan output berupa objek Movie untuk index "id"
iii. POST /api/Movies: menambahkan objek Movie baru
iv. DELETE /api/Movies/{id}: menghapus objek Movie pada index "id"

C. Secara default, program yang dibuat memiliki list film yang berasal dari TOP 3 film IMDB dari link: https://www.imdb.com/search/title/?groups=top_100&sort=user_rating,desc

D. Impementasi yang dibuat tidak menggunakan database, cukup disimpan sebagai suatu variable, dan gunakan "static" di variable tersebut yang menyimpan list/array dari objek- objek Movie.

E. Dalam pembuatan program/aplikasi ini, anda dapat mengasumsikan bahwa input dari user selalu benar dan sesuai dengan tipe data yang diharapkan.

Jwab

Source code Movies.cs

```
using System.Collections.Generic;

namespace modul9_2211104008
{
    1 reference
    public class Movie
    {
        0 references
        public string Title { get; set; }
        0 references
        public string Director { get; set; }
        0 references
        public List<string> Stars { get; set; }
        0 references
        public string Description { get; set; }

        0 references
        public Movie() { }
    }
}
```

MoviesController.cs

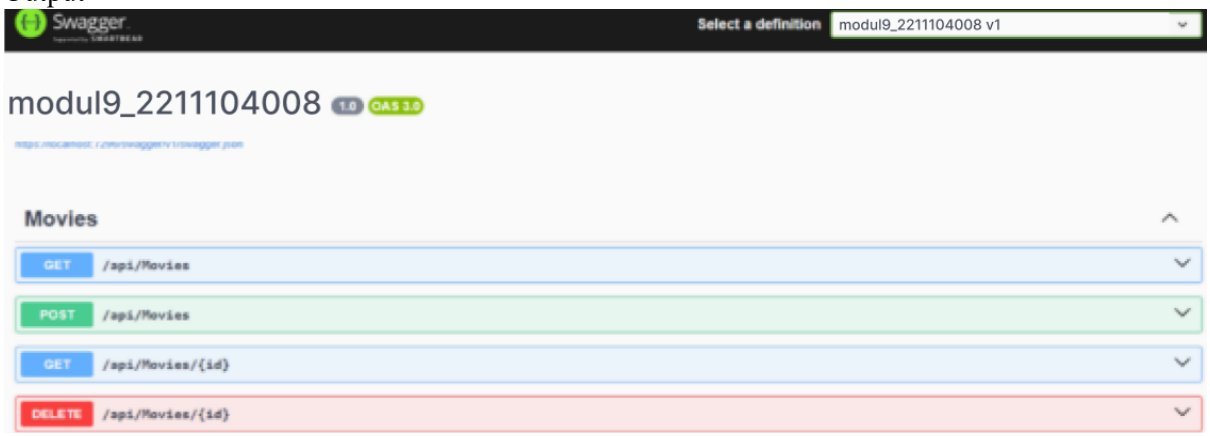
```
using Microsoft.AspNetCore.Mvc;
using System.Collections.Generic;

namespace modul9_2211104008.Controllers
{
    [ApiController]
    [Route("api/[controller]")]
    0 references
    public class MoviesController : ControllerBase
    {
        8 references
        private static List<Movie> movies = new List<Movie>
        {
            new Movie {
                Title = "The Shawshank Redemption",
                Director = "Frank Darabont",
                Stars = new List<string> { "Tim Robbins", "Morgan Freeman", "Bob Gunton" },
                Description = "Two imprisoned men bond over a number of years..."
            },
            new Movie {
                Title = "The Godfather",
                Director = "Francis Ford Coppola",
                Stars = new List<string> { "Marlon Brando", "Al Pacino", "James Caan" },
                Description = "The aging patriarch of an organized crime dynasty..."
            },
            new Movie {
                Title = "The Dark Knight",
                Director = "Christopher Nolan",
                Stars = new List<string> { "Christian Bale", "Heath Ledger", "Aaron Eckhart" },
                Description = "When the menace known as the Joker wreaks havoc..."
            }
        };

        // GET: api/Movies
        [HttpGet]
        0 references
        public ActionResult<List<Movie>> Get() => movies;
```

```
35
36          // GET: api/Movies/{id}
37          [HttpGet("{id}")]
            0 references
38          public ActionResult<Movie> Get(int id)
39          {
40              if (id < 0 || id >= movies.Count)
41                  return NotFound();
42              return movies[id];
43          }
44
45          // POST: api/Movies
46          [HttpPost]
            0 references
47          public ActionResult<List<Movie>> Post([FromBody] Movie newMovie)
48          {
49              movies.Add(newMovie);
50              return movies;
51          }
52
53          // DELETE: api/Movies/{id}
54          [HttpDelete("{id}")]
            0 references
55          public ActionResult<List<Movie>> Delete(int id)
56          {
57              if (id < 0 || id >= movies.Count)
58                  return NotFound();
59              movies.RemoveAt(id);
60              return movies;
61          }
62      }
63  }
```

Output



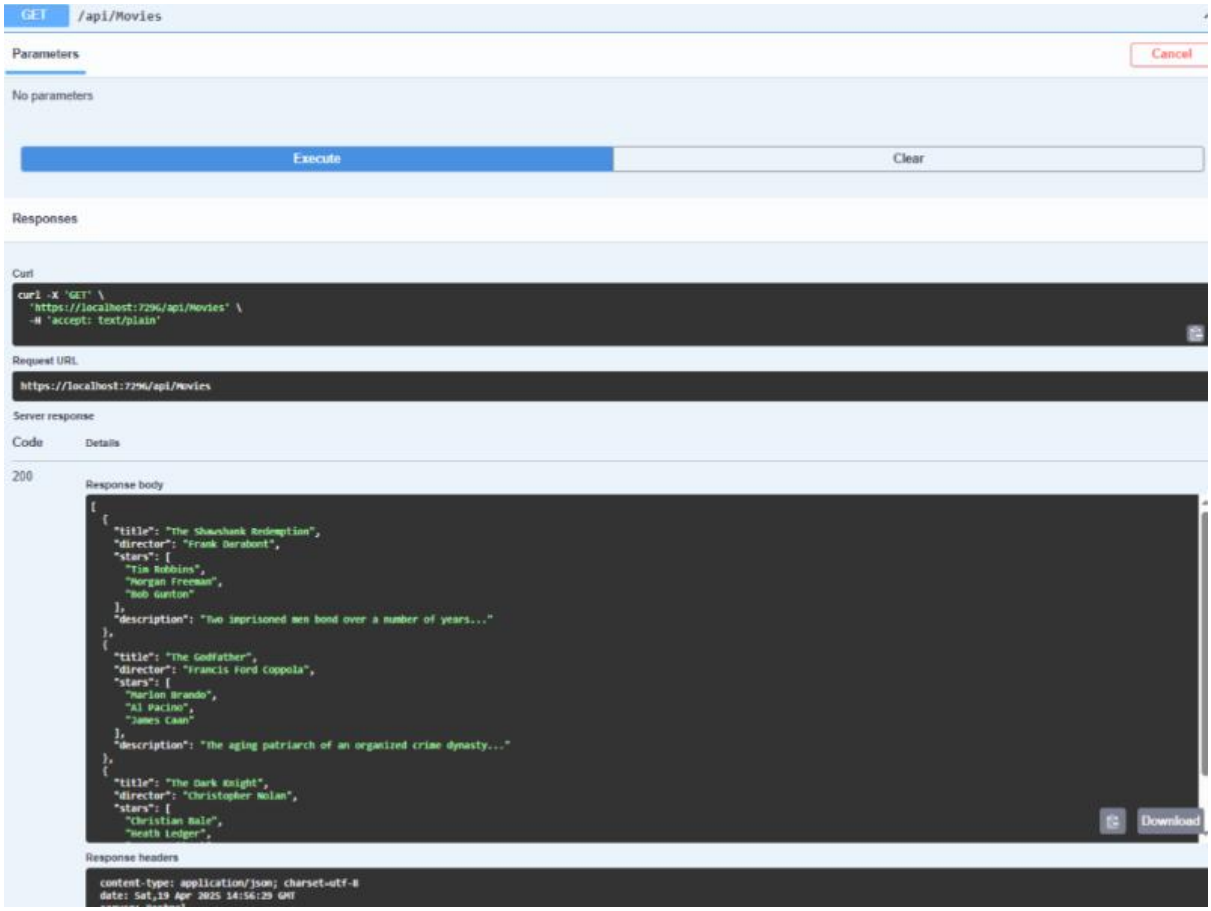
Penjelasan

Program ini adalah implementasi Web API sederhana dengan ASP.NET Core untuk mengelola informasi film. Data film disimpan sementara dalam movieList statis. MoviesController menangani permintaan pengguna melalui endpoint: GET /api/Movies (menampilkan semua film), GET /api/Movies/{id} (detail film), POST /api/Movies (menambah film), dan DELETE /api/Movies/{id} (menghapus film). Data bersifat sementara karena disimpan di memori dan akan hilang saat aplikasi dimatikan/dijalankan ulang. Atribut seperti [HttpGet], [HttpPost], dan [HttpDelete] digunakan untuk menentukan jenis permintaan HTTP.

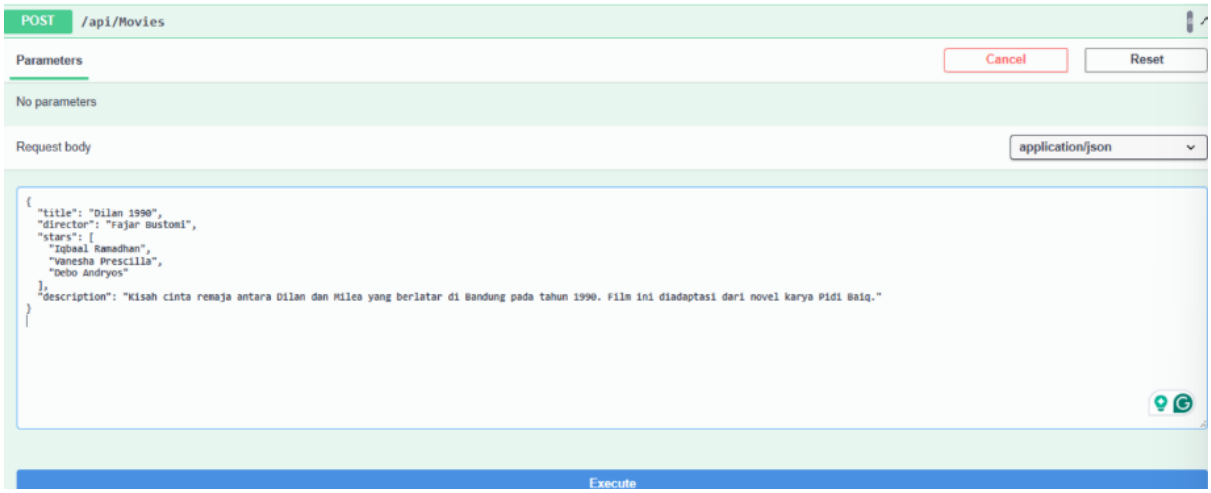
4. MENDEMONSTRASI WEB API

Beberapa skenario yang harus dicoba untuk memastikan jika program telah berjalan dengan baik. Buatlah dokumen yang berisi semua screenshot dari hasil uji coba scenario yang disebutkan pada list berikut ini:

A. Mencoba “GET /api/Movies” saat baru dijalankan yang mengeluarkan list film dari TOP 3 IMDB seperti pada tampilan berikut pada saat dicoba dengan menekan tombol “Try it out” dan tombol “Execute”

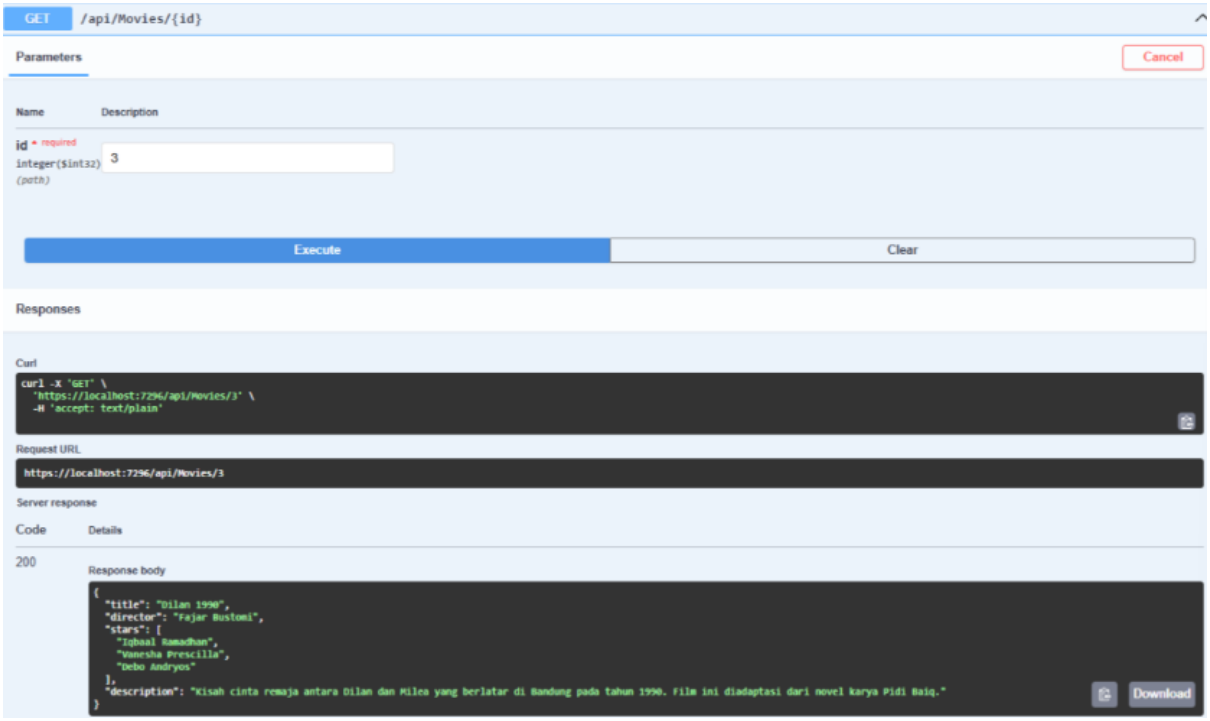


B. Menambahkan Movie baru yaitu urutan ke-4 pada TOP IMDB list dengan memanggil API pada bagian “POST /api/Movies”

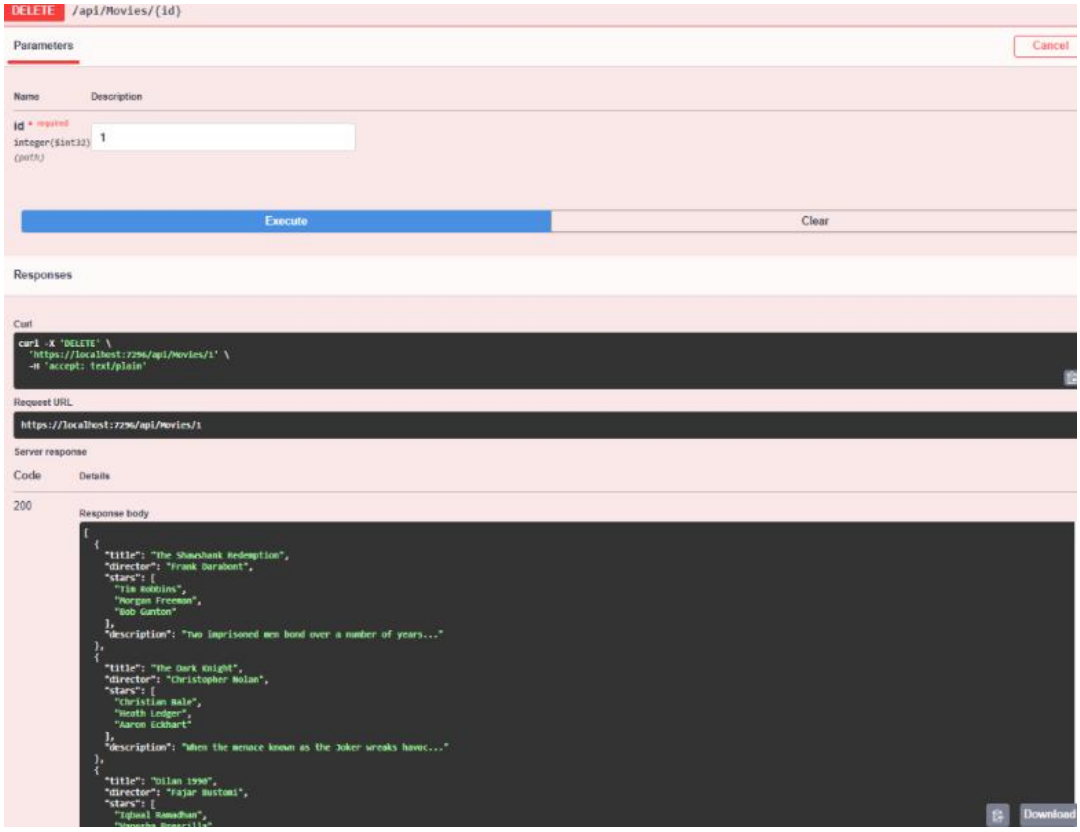


C. Cek list/array dari semua Movie lagi dengan “GET /api/Movies”, pastikan Movie yang baru ditambahkan sebelumnya sudah ada:

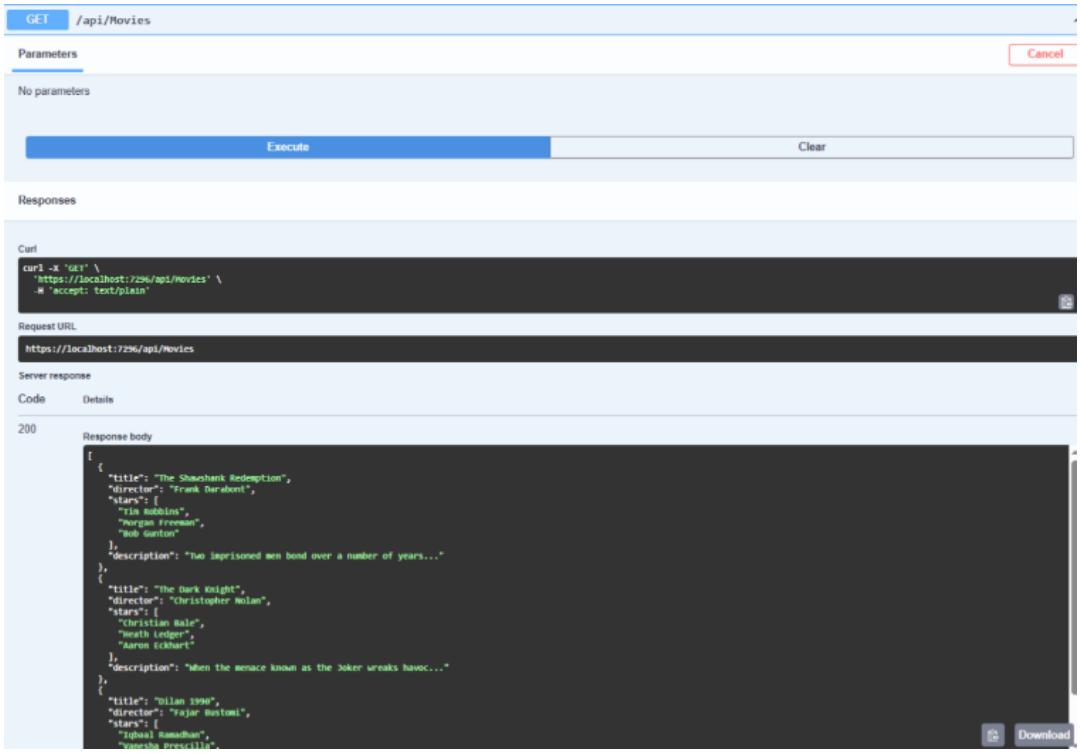
D. Mencoba meminta Movie dengan index 3, “GET /api/Movies/3” yang seharusnya mengembalikan Movie yang baru saja ditambah:



E. Menghapus objek Movie dengan index ke-1 dengan “DELETE /api/Movies/1”



F. Cek list/array dari semua Movie sekali lagi dengan “GET /api/Movies”, film dengan ranking kedua “Godfather” sudah tidak ada di list:



Penjelasan

Program Web API ASP.NET Core sederhana untuk mengelola data film (tiga film terbaik IMDb) yang disimpan sementara di movieList statis. MoviesController menyediakan endpoint GET (/api/Movies untuk semua film, /api/Movies/{id} untuk detail), POST (/api/Movies untuk menambah), dan DELETE (/api/Movies/{id} untuk menghapus). Data akan hilang saat aplikasi dimatikan/dijalankan ulang. Atribut HTTP ([HttpGet], dll.) digunakan untuk mendefinisikan jenis permintaan.