RAKTIKUM KONSTRUKSI PERANGKAT LUNAK
TUGAS JURNAL 09

Api Design & Construction Using Swagger

Telkom University

disusun Oleh:
Nita Fitrotul Mar'ah
2211104005
SE0601

Dosen Pengampu :
Yudha Islami Sulistya, S.Kom., M.Cs.

S1 REKAYASA PERANGKAT LUNAK
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO
2025

1. **MEMBUAT PROJECT WEB API**

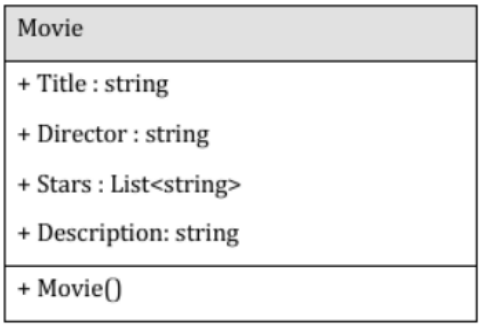
   Berhubung cara membuat project web api berbeda-beda untuk setiap bahasa pemrograman, langkah langkah berikut hanya berlaku apabila dilakukan dengan menggunakan .NET dan Visual Studio. Untuk
   IDE dan bahasa pemrograman lain, yang terpenting adalah nama project yang dibuat yaitu "modul8_NIM"

   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=aspnetcore6.0&tabs=visual-studio
   f. Setelah project tersebut selesai dibuat, coba run programnya, dan tunggu sampai program selesai di-compile.

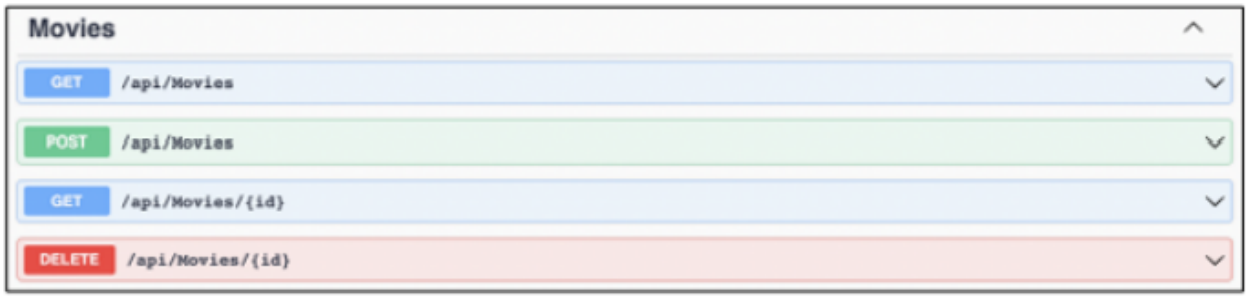
2. **IMPLEMENTASI WEB API**

   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



   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:/swagger/index.html):



- GET /api/Movies: mengembalikan output berupa list/array dari semua objek Movies
- GET /api/Movies/{id}: mengembalikan output berupa objek Movie untuk index "id"
- POST /api/Movies: menambahkan objek Movie baru
- 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.

Jawab:

- **Source code**

  Movie.cs

```
using System.Collections.Generic;

namespace modul9_2211104005
{
    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() { }
    }
}
```

Controllers/MoviesController.cs

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

namespace modul9_2211104005.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;

        // GET: api/Movies/{id}
        [HttpGet("{id}")]
        0 references
        public ActionResult<Movie> Get(int id)
        {
            if (id < 0 || id >= movies.Count)
                return NotFound();
            return movies[id];
        }

        // POST: api/Movies
        [HttpPost]
        0 references
        public ActionResult<List<Movie>> Post([FromBody] Movie newMovie)
        {
            movies.Add(newMovie);
            return movies;
        }

        // DELETE: api/Movies/{id}
        [HttpDelete("{id}")]
        0 references
        public ActionResult<List<Movie>> Delete(int id)
        {
            if (id < 0 || id >= movies.Count)
                return NotFound();
            movies.RemoveAt(id);
            return movies;
        }
    }
}
```

- **Screenshot hasil run**

- **Penjelasan**

  Program ini merupakan Web API sederhana yang dibangun dengan ASP.NET Core dan berfungsi untuk mengelola data film. Informasi film disimpan dalam sebuah list statis bernama `movieList`, yang berisi tiga film teratas dari daftar IMDB. API ini menggunakan `MoviesController` sebagai pengendali utama untuk menangani berbagai permintaan pengguna melalui sejumlah endpoint seperti GET, POST, dan DELETE.

  Endpoint `GET /api/Movies` digunakan untuk menampilkan seluruh data film, sementara `GET /api/Movies/{id}` digunakan untuk menampilkan data film berdasarkan indeks tertentu. Untuk menambahkan film baru, digunakan endpoint `POST /api/Movies`, sedangkan penghapusan film berdasarkan indeks dilakukan melalui endpoint `DELETE /api/Movies/{id}`.

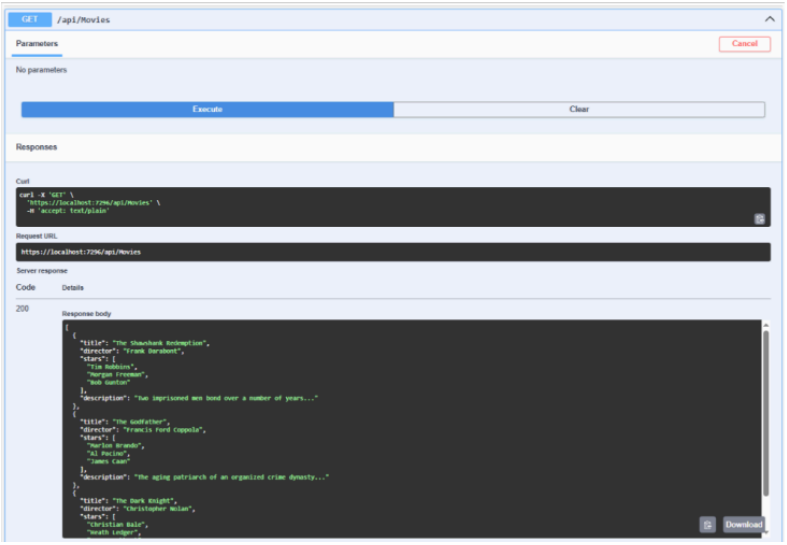
  Semua data disimpan hanya di dalam memori (menggunakan list statis), sehingga ketika aplikasi dimatikan atau dijalankan ulang, data akan kembali ke kondisi awal. Atribut seperti `[HttpGet]`, `[HttpPost]`, dan `[HttpDelete]` digunakan untuk menentukan tipe permintaan HTTP yang ditangani oleh masing-masing metode dalam controller.

3. 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:

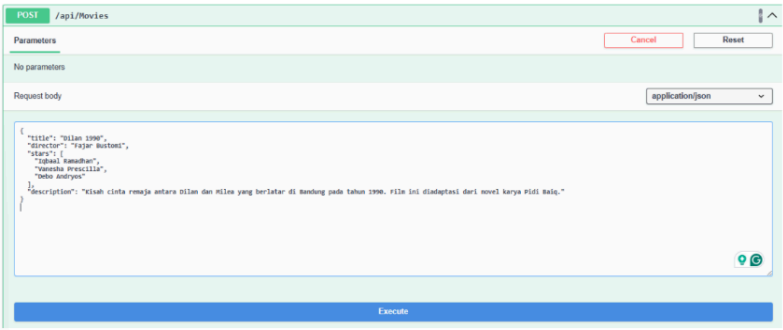
a. Mencoba “GET /api/Movies” saat baru dijalankan yang mengeluarkan list film dari TOP 3 IMDB seperti pada tampilan berikut pada saat dicoba dengan menekan tombol “Try it out” dan tombol “Execute”

Hasil:



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

Hasil:

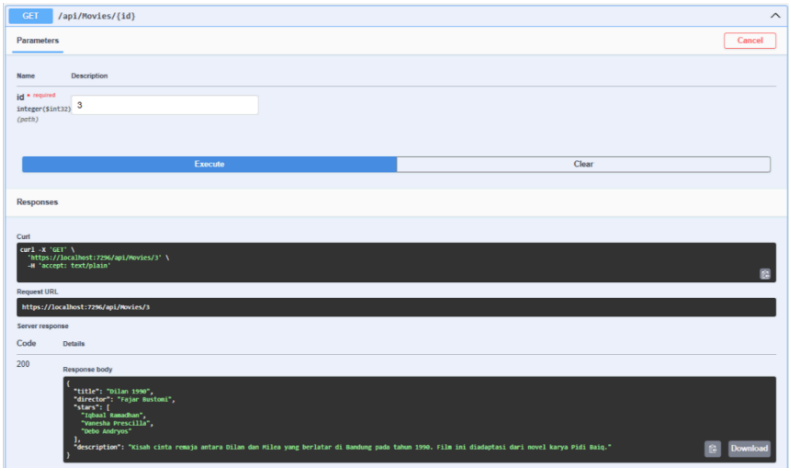


c. Cek list/array dari semua Movie lagi dengan “GET /api/Movies”, pastikan Movie yang baru ditambahkan sebelumnya sudah ada:

Hasil:

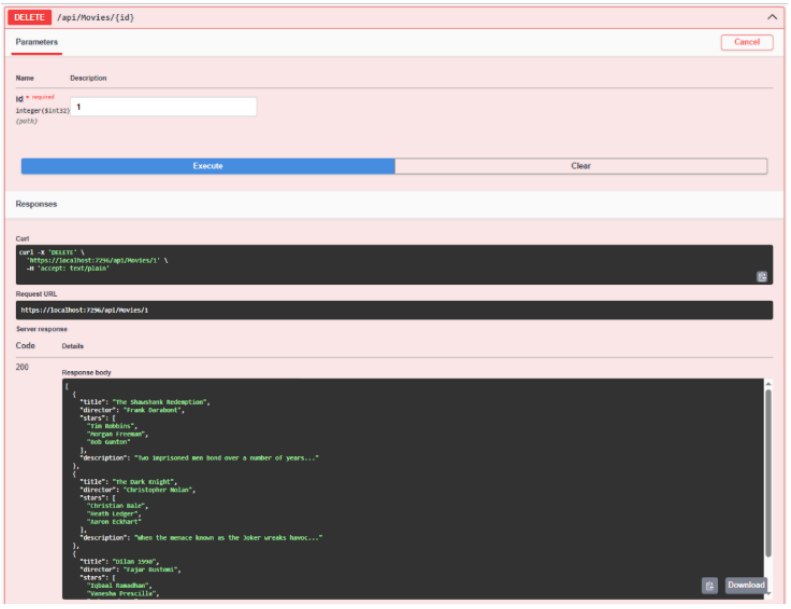
d. Mencoba meminta Movie dengan index 3, “GET /api/Movies/3” yang seharusnya mengembalikan Movie yang baru saja ditambah:

Hasil:



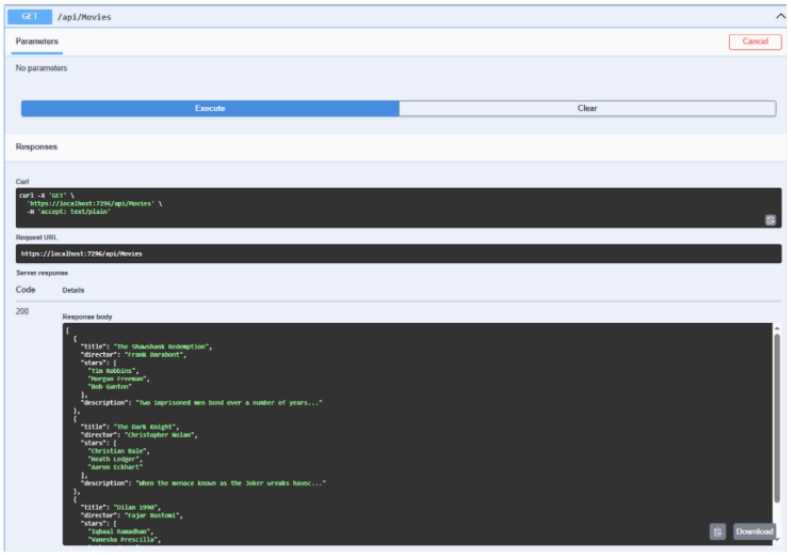
e. Menghapus objek Movie dengan index ke-1 dengan “DELETE /api/Movies/1”

Hasil:



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

Hasil:



Penjelasan

Endpoint `POST /api/Movies` digunakan untuk menambahkan film baru ke dalam koleksi, sedangkan `DELETE /api/Movies/{id}` digunakan untuk menghapus film berdasarkan indeks yang diberikan. Seluruh data hanya disimpan di memori melalui list statis, sehingga data tidak bersifat permanen dan akan kembali ke kondisi awal setiap kali aplikasi dimatikan atau dijalankan ulang. Atribut seperti `[HttpGet]`, `[HttpPost]`, dan `[HttpDelete]` digunakan untuk menetapkan jenis permintaan HTTP yang akan diproses oleh masing-masing metode di controller.