

## Jurnal Modul 9

Nama : Farhan Kurniawan

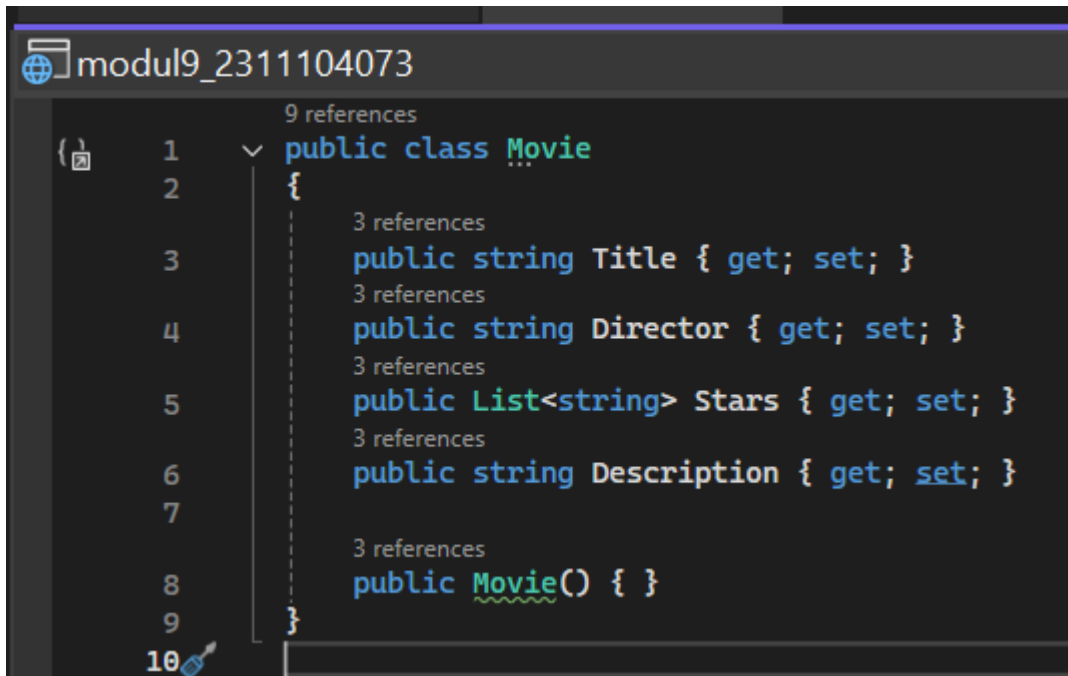
NIM : 2311104073

Kelas : SE-07-02

Link Github : [https://github.com/farkurr/KPL\\_FARHANKURNIAWAN\\_2311104073\\_SE-07-02.git](https://github.com/farkurr/KPL_FARHANKURNIAWAN_2311104073_SE-07-02.git)

### Penjelasan Source Kode

#### 1. Movie.cs



```
modul9_2311104073
9 references
1 public class Movie
2 {
3     3 references
4     public string Title { get; set; }
5     3 references
6     public string Director { get; set; }
7     3 references
8     public List<string> Stars { get; set; }
9     3 references
10    public string Description { get; set; }
11
12    3 references
13    public Movie() { }
14 }
```

File Movie.cs berisi deklarasi class Movie yang merepresentasikan struktur data utama dari aplikasi Web API ini. Kelas ini memiliki empat properti yaitu Title, Director, Stars, dan Description, yang masing-masing bertipe data string, kecuali Stars yang bertipe List<string> karena mewakili daftar nama aktor/aktris yang terlibat dalam film. Konstruktor default (Movie()) disediakan untuk memungkinkan pembuatan objek Movie tanpa harus langsung mengisi nilainya. Kelas ini digunakan sebagai model utama dalam proses pertukaran data (request dan response) pada endpoint API yang tersedia.

## 2. MoviesController.cs

```
modul9_2311104073

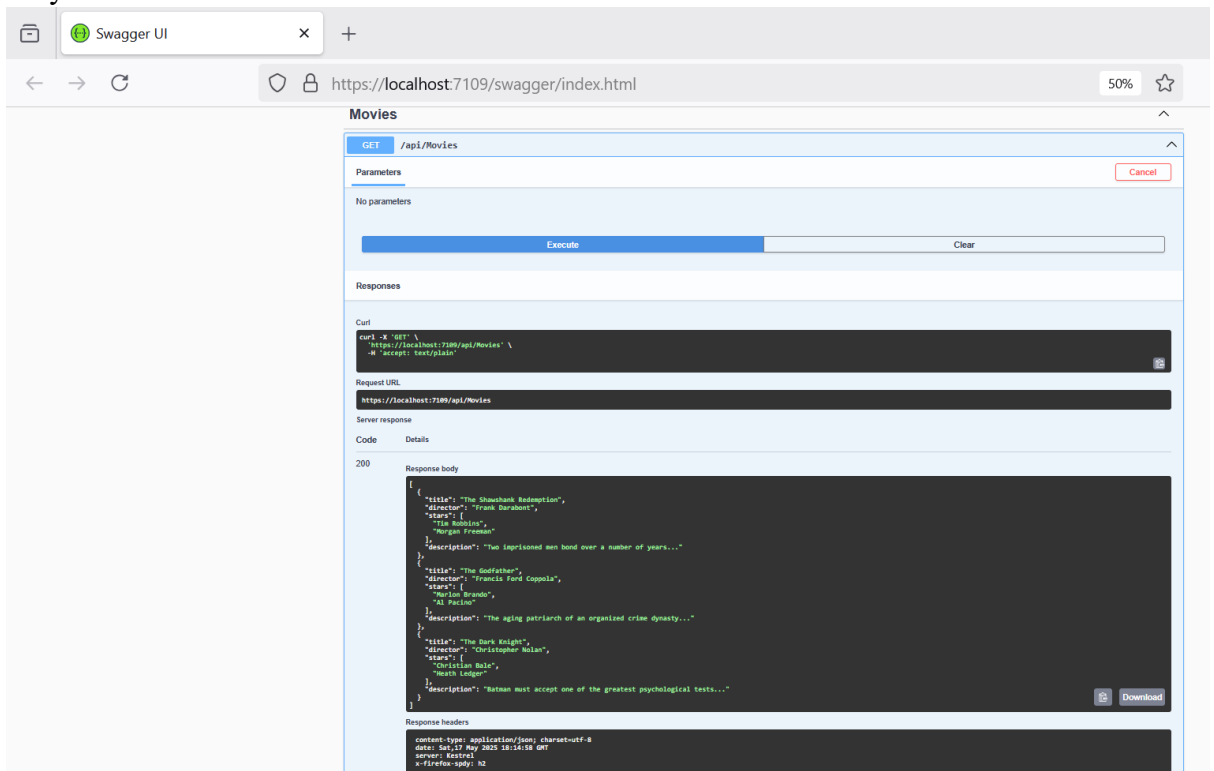
1 using Microsoft.AspNetCore.Mvc;
2
3 [Route("api/[controller]")]
4 [ApiController]
5 public class MoviesController : ControllerBase
6 {
7     private static List<Movie> movies = new List<Movie>
8     {
9         new Movie
10         {
11             Title = "The Shawshank Redemption",
12             Director = "Frank Darabont",
13             Stars = new List<string> { "Tim Robbins", "Morgan Freeman" },
14             Description = "Two imprisoned men bond over a number of years..."
15         },
16         new Movie
17         {
18             Title = "The Godfather",
19             Director = "Francis Ford Coppola",
20             Stars = new List<string> { "Marlon Brando", "Al Pacino" },
21             Description = "The aging patriarch of an organized crime dynasty..."
22         },
23         new Movie
24         {
25             Title = "The Dark Knight",
26             Director = "Christopher Nolan",
27             Stars = new List<string> { "Christian Bale", "Heath Ledger" },
28             Description = "Batman must accept one of the greatest psychological tests..."
29         }
30     };
31
32 [HttpGet]
33 public ActionResult<List<Movie>> GetAll() => movies;
34
35 [HttpGet("{id}")]
36 public ActionResult<Movie> Get(int id)
37 {
38     if (id < 0 || id >= movies.Count) return NotFound();
39     return movies[id];
40 }
41
42 [HttpPost]
43 public ActionResult Add(Movie movie)
44 {
45     movies.Add(movie);
46     return Ok();
47 }
48
49 [HttpDelete("{id}")]
50 public ActionResult Delete(int id)
51 {
52     if (id < 0 || id >= movies.Count) return NotFound();
53     movies.RemoveAt(id);
54     return Ok();
55 }
56 }
```

File MoviesController.cs merupakan implementasi controller untuk mengatur jalannya proses HTTP request dan response dari API yang berkaitan dengan data film. Controller ini ditandai dengan atribut [ApiController] dan rute [Route("api/[controller]")] yang secara otomatis menetapkan URL ke /api/Movies. Di dalamnya terdapat list statis movies yang menyimpan data tiga film terbaik dari IMDB secara default. Controller ini menyediakan empat endpoint utama: GET untuk menampilkan semua film, GET/{id} untuk mengambil satu film berdasarkan indeks,

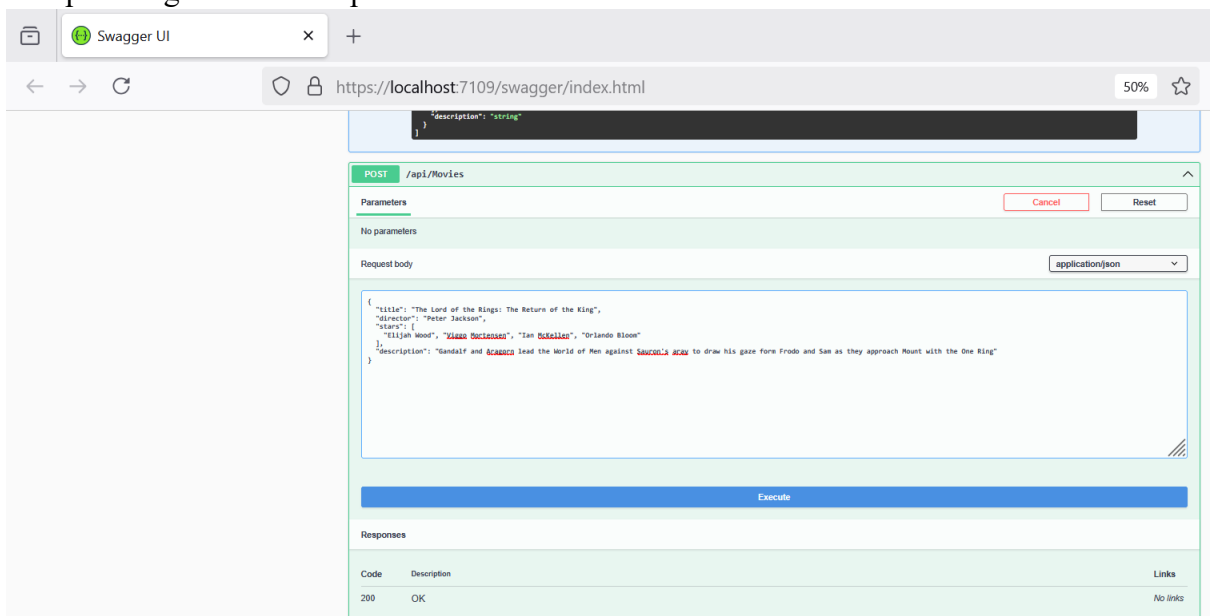
POST untuk menambahkan film baru ke dalam list, dan DELETE/{id} untuk menghapus film berdasarkan indeks. Karena list bersifat statis, seluruh data disimpan sementara selama aplikasi berjalan tanpa menggunakan database

## Demonstrasi Web API

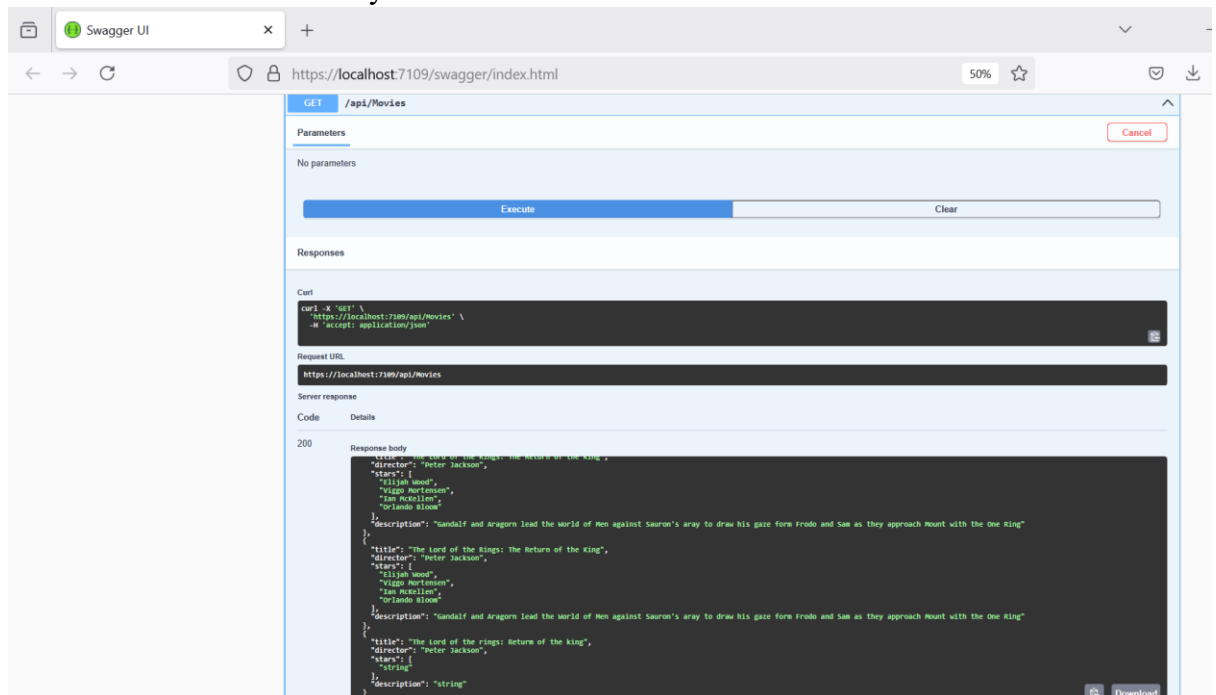
1. Mencoba “GET /api/Movies” saat baru dijalankan yang mengeluarkan list film dari TOP 3 IMDB seperti pada tampilan berikut pada saat dicoba dengan menekan tombol “Try it out” dan tombol “Execute”



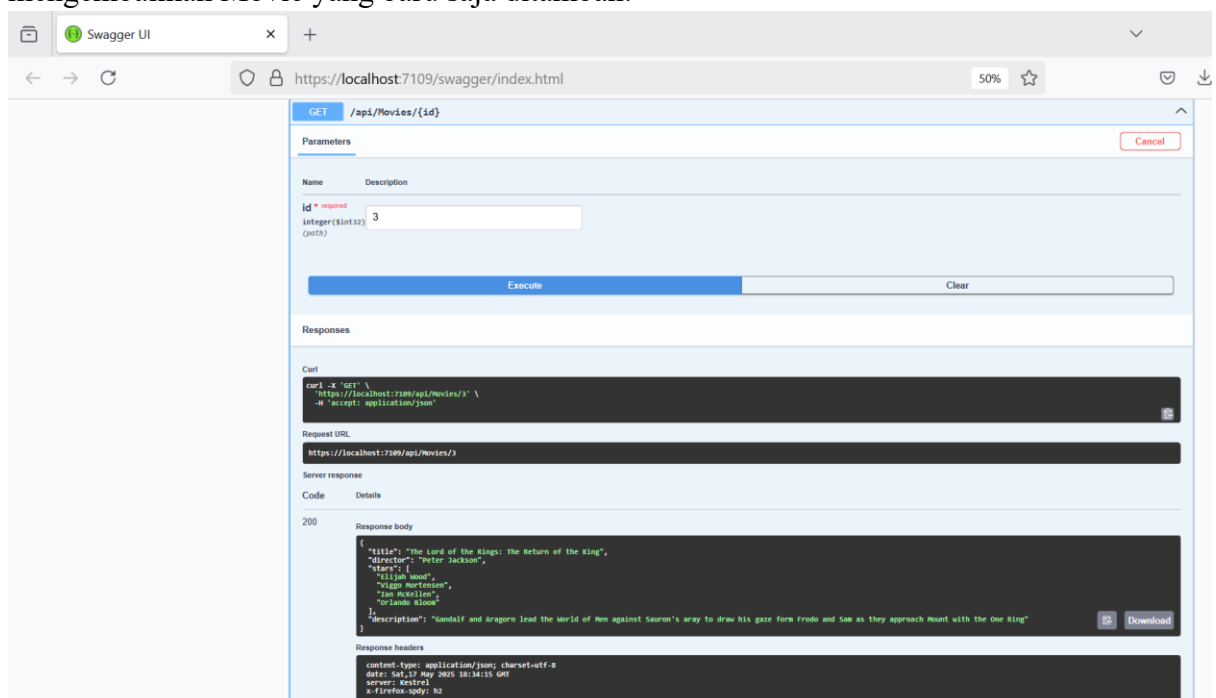
2. Menambahkan Movie baru yaitu urutan ke-4 pada TOP IMDB list dengan memanggil API pada bagian “POST /api/Movies”



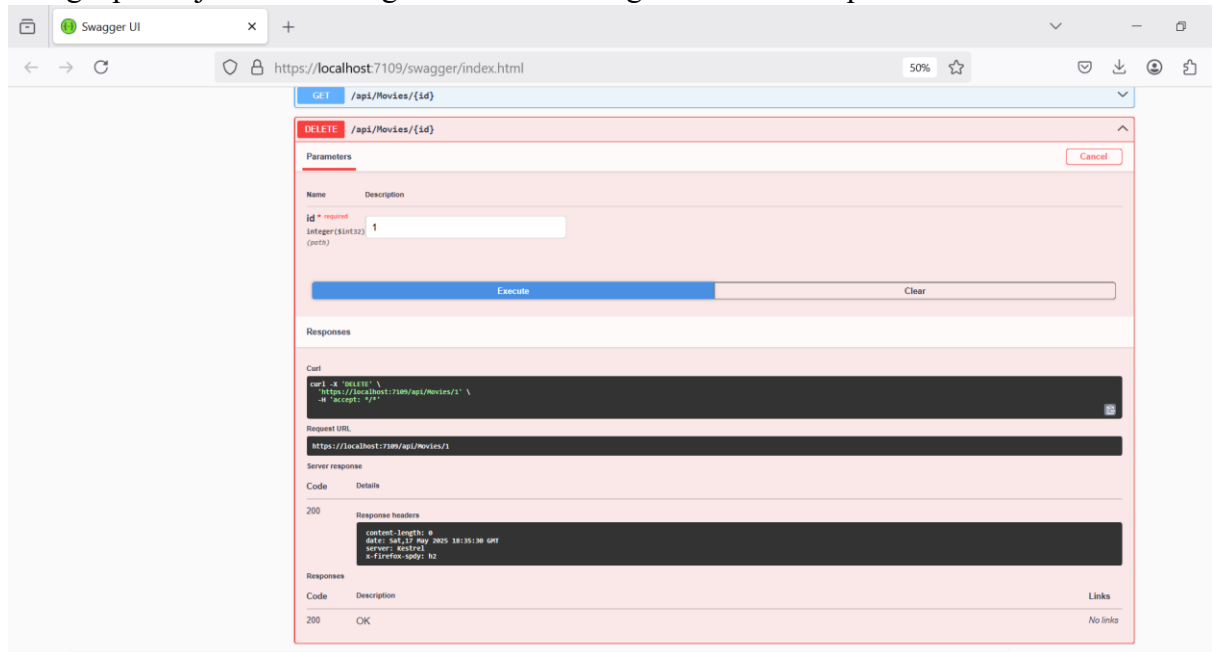
3. Cek list/array dari semua Movie lagi dengan “GET /api/Movies”, pastikan Movie yang baru ditambahkan sebelumnya sudah ada:



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



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



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

