

Nama : Naufal Aflakh Wijayanto

Kelas : SE063

NIM : 2211104073

JURNAL MODUL 9

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.

BUKTI Pengerjaan

Swagger UI

https://localhost:7152/swagger/index.html

 Swagger
Support by SMARTBEAR

Select a definition

modul9_2211104073 v1

modul9_2211104073

1.0

OAS 3.0

<https://localhost:7152/swagger/v1/swagger.json>

WeatherForecast

^

GET

/WeatherForecast



^

Schemas

^

WeatherForecast >

83°F
Sunny

 Search 

10:04 AM
6/8/2025

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.

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

Movie.cs

```
using System;
using System.Collections.Generic;

namespace modul9_2211104073.Models
{
    public class Movie
    {
        public string Title { get; set; }
        public string Director { get; set; }
        public List<string> Stars { get; set; }
        public string Description { get; set; }

        // Constructor (Opsional, jika tidak diperlukan, bisa dihilangkan)
        public Movie() {}
    }
}
```

Penjelasan kode:

Kode di atas adalah sebuah kelas bernama `Movie` dalam namespace `modul9_2211104073.Models` yang ditulis menggunakan bahasa pemrograman C#. Kelas ini merepresentasikan data sebuah film dengan beberapa properti, yaitu `Title` (judul film), `Director` (sutradara), `Stars` (daftar aktor/aktris dalam bentuk list string), dan `Description` (deskripsi film). Properti-properti tersebut menggunakan auto-implemented properties untuk menyederhanakan deklarasi getter dan setter. Selain itu, terdapat konstruktor default (`public Movie() { }`) yang bersifat opsional dan dapat digunakan untuk menginisialisasi objek tanpa parameter. Kelas ini biasanya digunakan dalam konteks pemrograman berbasis objek untuk menyimpan dan mengelola data film secara terstruktur.

- 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>):

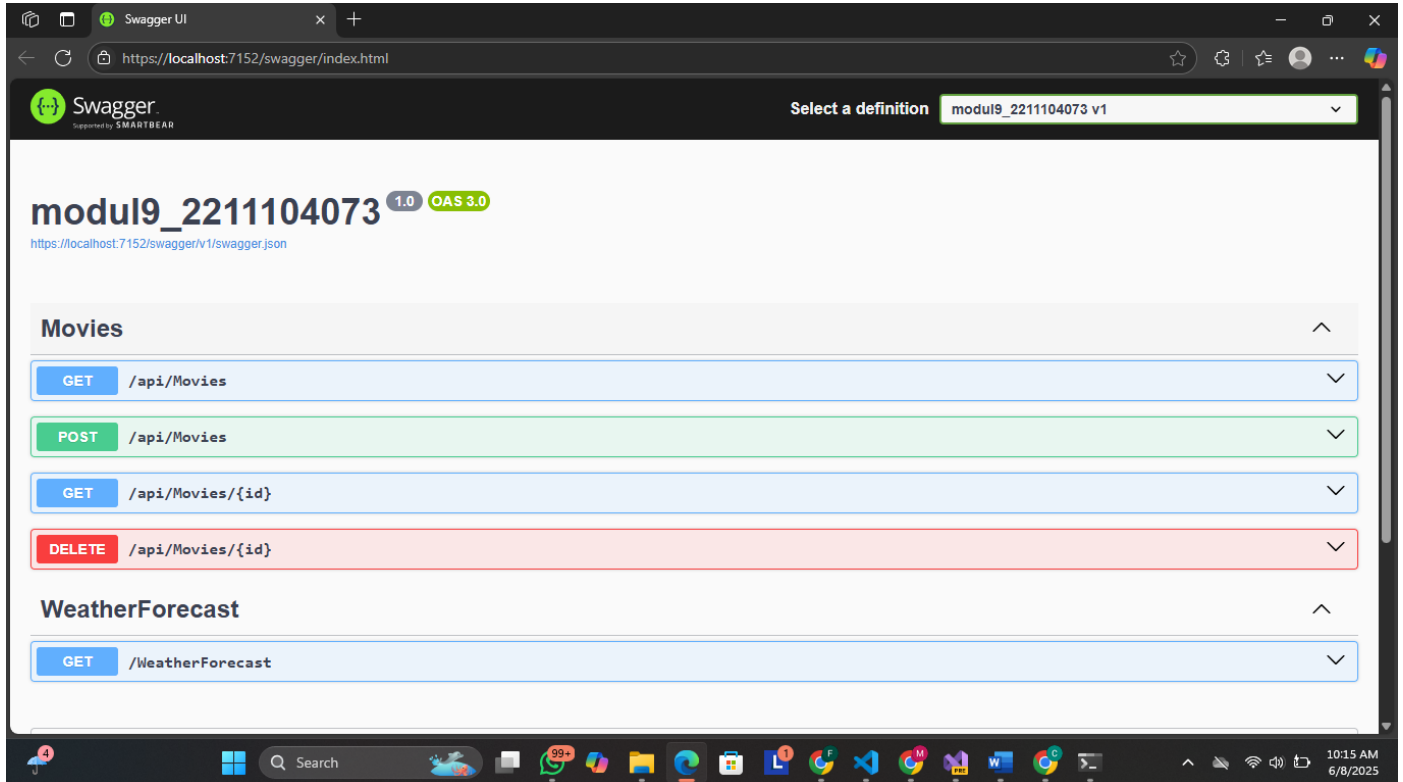
Movies		^
GET	/api/Movies	▼
POST	/api/Movies	▼
GET	/api/Movies/{id}	▼
DELETE	/api/Movies/{id}	▼

- 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. Implementasi 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.

BUKTI Pengerjaan



MoviesController.cs

```
using Microsoft.AspNetCore.Mvc;
using modul9_2211104073.Models; // Referensikan kelas Movie yang telah dibuat
using System.Collections.Generic;

namespace modul9_2211104073.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class MoviesController : ControllerBase
    {
        // Static list of Movies (menyimpan data film sementara di memori)
        private static List<Movie> movies = new List<Movie>
        {
            new Movie { Title = "Inception", Director = "Christopher Nolan", Stars = new List<string> {
                "Leonardo DiCaprio", "Joseph Gordon-Levitt" }, Description = "A thief who steals corporate secrets..." },
        }
```

```

        Title = "The Shawshank Redemption",
        Director = "Frank Darabont",
        Stars = new List<string> { "Tim Robbins", "Morgan Freeman" },
        Description = "Two imprisoned men bond over a number of years, finding solace and eventual
redemption through acts of common decency."
    },
    new Movie
    {
        Title = "The Godfather",
        Director = "Francis Ford Coppola",
        Stars = new List<string> { "Marlon Brando", "Al Pacino" },
        Description = "The aging patriarch of an organized crime dynasty transfers control of his
clandestine empire to his reluctant son."
    },
    new Movie
    {
        Title = "The Dark Knight",
        Director = "Christopher Nolan",
        Stars = new List<string> { "Christian Bale", "Heath Ledger" },
        Description = "When the menace known as the Joker emerges from his mysterious past, he
wreaks havoc and chaos on the people of Gotham."
    }
};

// GET /api/Movies: Mengembalikan semua movie
[HttpGet]
public IActionResult GetMovies()
{
    return Ok(Movies); // Mengembalikan seluruh list film
}

// GET /api/Movies/{id}: Mengembalikan movie berdasarkan ID
[HttpGet("{id}")]
public IActionResult GetMovie(int id)
{
    // Periksa apakah ID valid
    if (id < 0 || id >= Movies.Count)
    {
        return NotFound(); // Jika ID tidak ditemukan, kembalikan 404 NotFound
    }

    return Ok(Movies[id]); // Mengembalikan movie berdasarkan ID
}

```

```

// POST /api/Movies: Menambahkan movie baru
[HttpPost]
public IActionResult AddMovie([FromBody] Movie newMovie)
{
    if (newMovie == null)
    {
        return BadRequest("Movie data is required."); // Memastikan data valid
    }

    Movies.Add(newMovie); // Menambahkan movie ke dalam list
    return CreatedAtAction(nameof(GetMovie), new { id = Movies.Count - 1 }, newMovie); //
Mengembalikan HTTP 201 Created
}

// DELETE /api/Movies/{id}: Menghapus movie berdasarkan ID
[HttpDelete("{id}")]
public IActionResult DeleteMovie(int id)
{
    // Memeriksa apakah ID valid
    if (id < 0 || id >= Movies.Count)
    {
        return NotFound(); // Jika ID tidak ditemukan, kembalikan 404 NotFound
    }

    Movies.RemoveAt(id); // Menghapus movie berdasarkan ID
    return NoContent(); // Mengembalikan HTTP 204 No Content (berhasil menghapus)
}
}
}

```

Penjelasan kode:

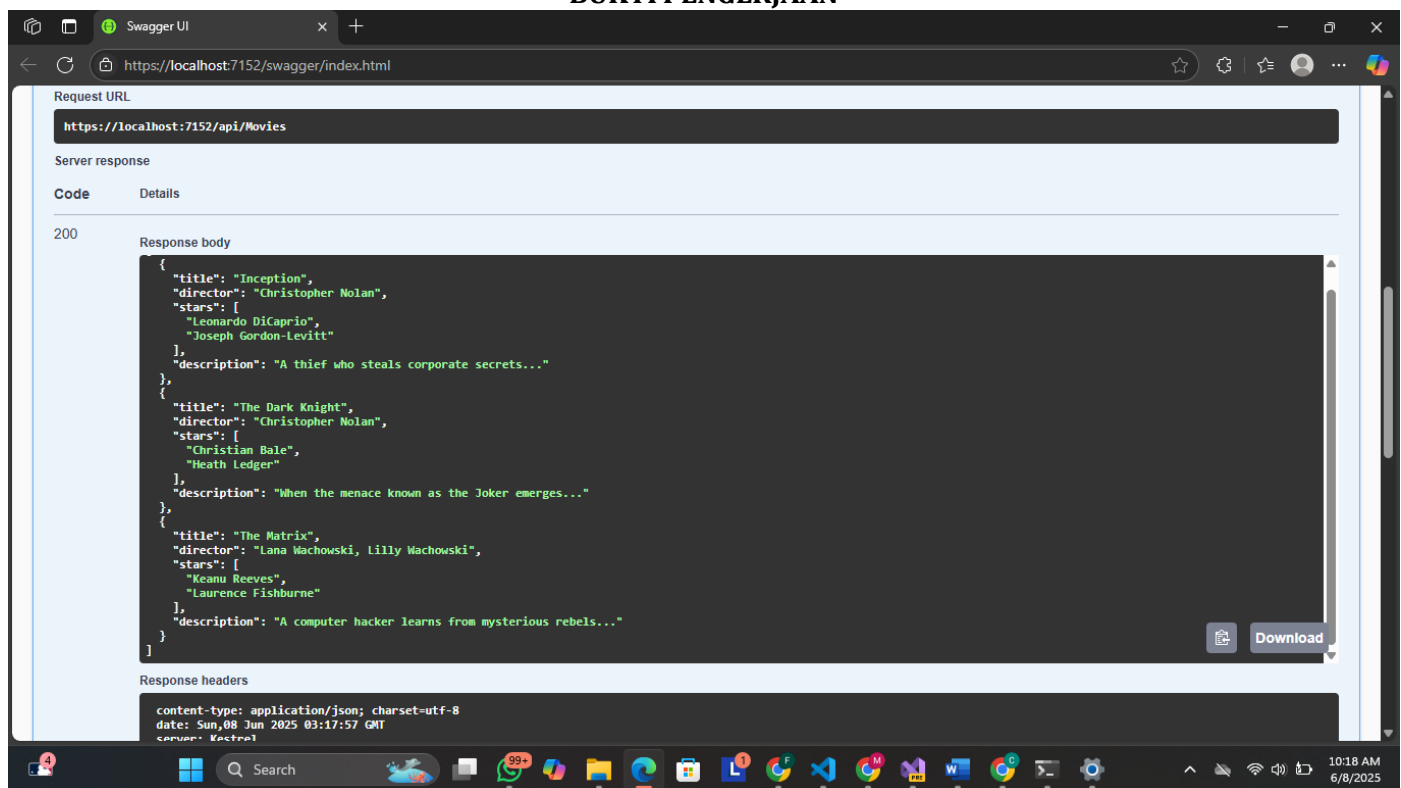
Kode di atas merupakan implementasi dari sebuah *Web API controller* bernama *MoviesController* dalam framework ASP.NET Core. Controller ini menangani operasi CRUD sederhana untuk objek *Movie* yang sudah didefinisikan sebelumnya. Data film disimpan dalam list statis *movies* yang bersifat sementara dan hanya ada selama aplikasi berjalan. Terdapat empat endpoint utama: `[HttpGet]` untuk mengambil seluruh daftar film, `[HttpGet("{id}")]` untuk mengambil film berdasarkan index, `[HttpPost]` untuk menambahkan film baru, dan `[HttpDelete("{id}")]` untuk menghapus film berdasarkan index. Setiap metode memanfaatkan atribut routing dan anotasi seperti *ApiController* untuk memudahkan pengelolaan HTTP request dan response, serta menghasilkan status kode HTTP yang sesuai seperti 200 OK, 201 Created, 404 Not Found, dan 204 No Content. Kode ini cocok digunakan sebagai dasar untuk API layanan film sederhana.

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:

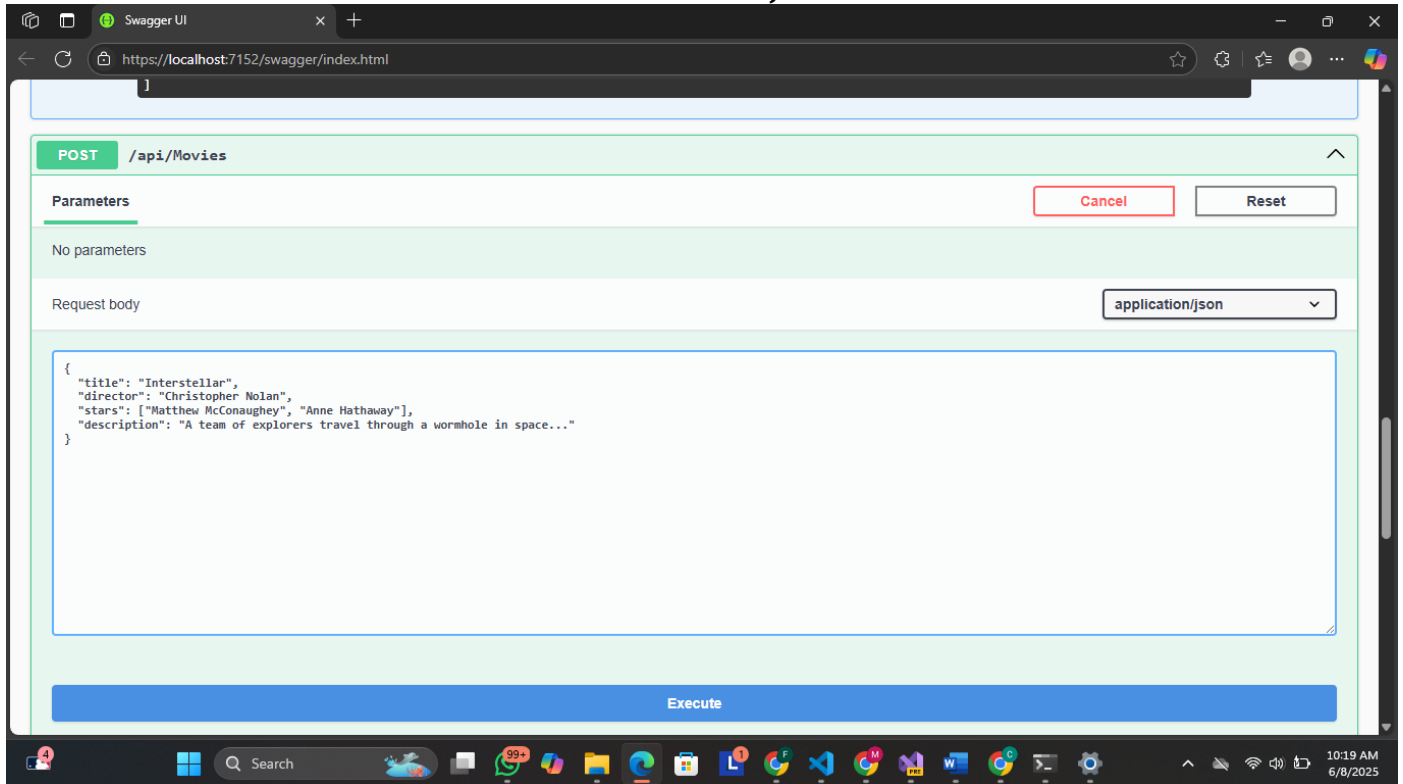
- A. Mencoba “GET /api/Movies” saat baru dijalankan yang mengeluarkan list film dari TOP 3 IMDB seperti pada tampilan berikut pada saat dicoba dengan menekan tombol “Try it out” dan tombol “Execute”

BUKTI Pengerjaan



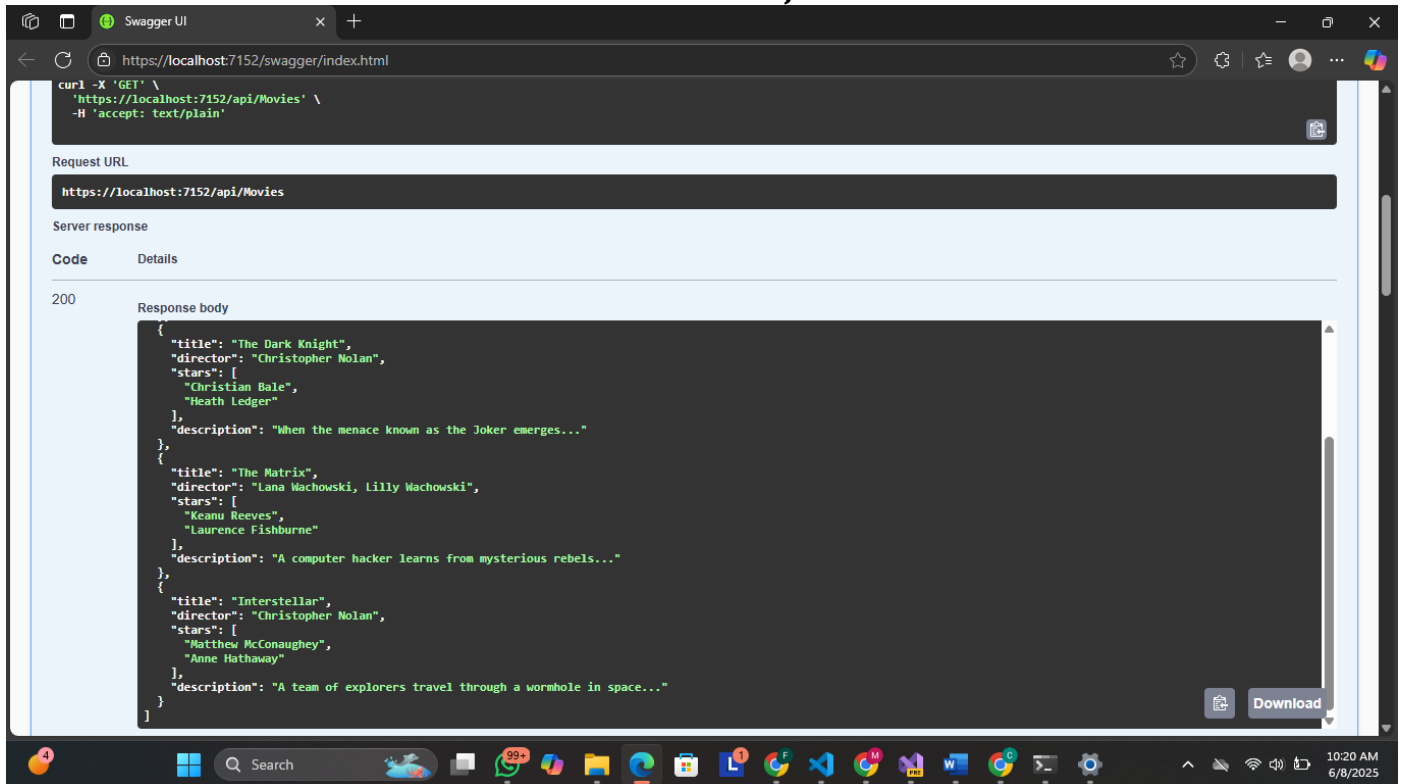
- B. Menambahkan Movie baru yaitu urutan ke-4 pada TOP IMDB list dengan memanggil API pada bagian "POST /api/Movies"

BUKTI Pengerjaan



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

BUKTI Pengerjaan



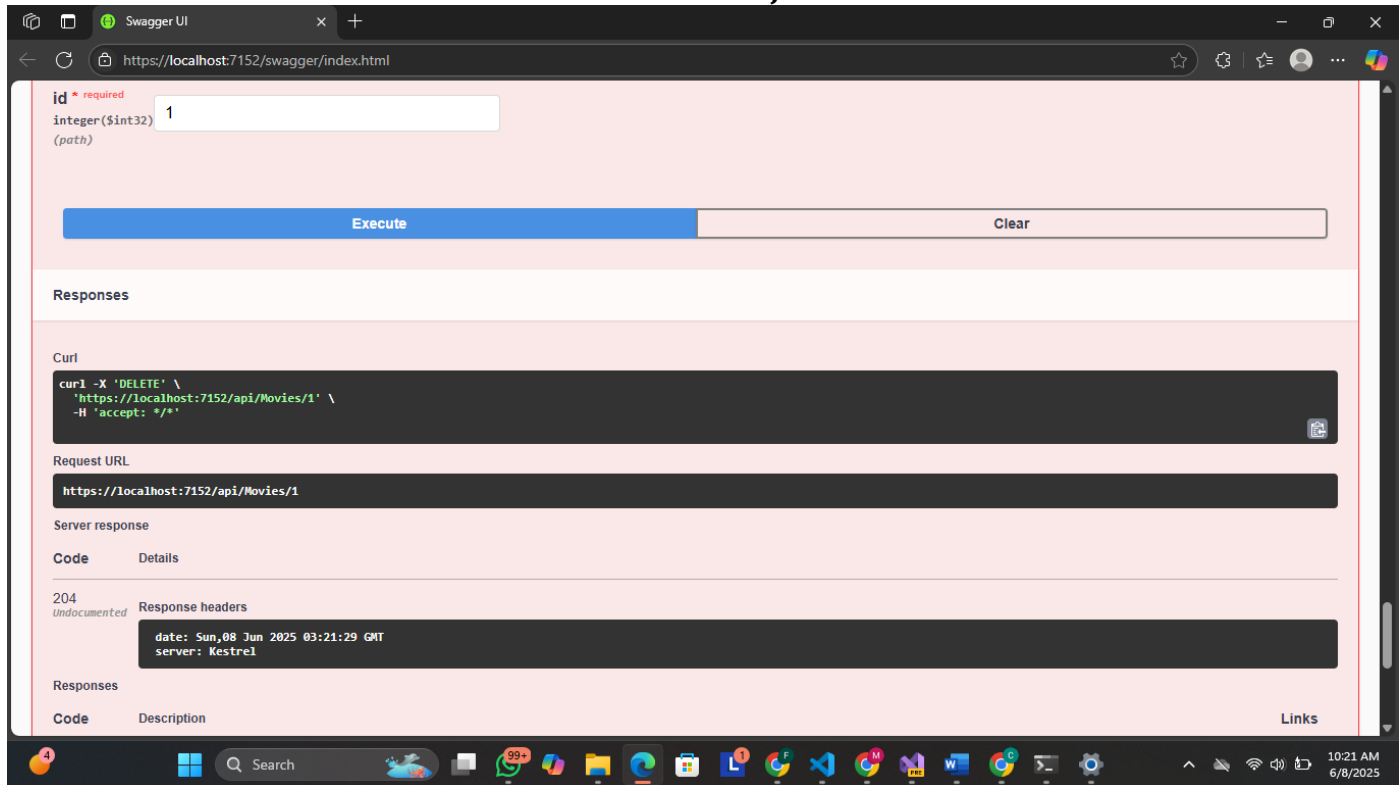
- D. Mencoba meminta Movie dengan index 3, “GET /api/Movies/3” yang seharusnya mengembalikan Movie yang baru saja ditambah:

BUKTI Pengerjaan

The screenshot shows the Swagger UI interface in a web browser. The URL bar indicates the page is at `https://localhost:7152/swagger/index.html`. In the top section, the 'id' parameter is set to '3' with a type of 'integer(\$int32)' and a '(path)' label. Below this, there are 'Execute' and 'Clear' buttons. The 'Responses' section is expanded, showing the 'Curl' command: `curl -X 'GET' \ 'https://localhost:7152/api/Movies/3' \ -H 'accept: text/plain'`. The 'Request URL' is `https://localhost:7152/api/Movies/3`. The 'Server response' section shows a '200' status code. The 'Response body' is a JSON object: `{ "title": "Interstellar", "director": "Christopher Nolan", "stars": ["Matthew McConaughey", "Anne Hathaway"], "description": "A team of explorers travel through a wormhole in space..." }`. A 'Download' button is visible next to the response body. The Windows taskbar at the bottom shows the time as 10:21 AM on 6/8/2025.

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

BUKTI Pengerjaan



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

BUKTI Pengerjaan

Swagger UI

https://localhost:7152/swagger/index.html

```
'https://localhost:7152/api/Movies' \
-H 'accept: text/plain'
```

Request URL

https://localhost:7152/api/Movies

Server response

Code

Details

200

Response body

```
{
  {
    "title": "Inception",
    "director": "Christopher Nolan",
    "stars": [
      "Leonardo DiCaprio",
      "Joseph Gordon-Levitt"
    ],
    "description": "A thief who steals corporate secrets..."
  },
  {
    "title": "The Matrix",
    "director": "Lana Wachowski, Lilly Wachowski",
    "stars": [
      "Keanu Reeves",
      "Laurence Fishburne"
    ],
    "description": "A computer hacker learns from mysterious rebels..."
  },
  {
    "title": "Interstellar",
    "director": "Christopher Nolan",
    "stars": [
      "Matthew McConaughey",
      "Anne Hathaway"
    ],
    "description": "A team of explorers travel through a wormhole in space..."
  }
}
```

Download

Response headers

4

Search

10:22 AM
6/8/2025