

TUGAS JURNAL
KONSTRUKSI PERANGKAT LUNAK

MODUL IX
API DESIGN DAN CONSTRUCTION USING SWAGGER



Disusun Oleh:

Aorinka Anendya Chazanah / 2211104013

S1 SE-06-01

Dosen Pengampu:

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

PROGRAM STUDI S1 SOFTWARE ENGINEERING

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

TUGAS JURNAL

1. 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>):

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.

Source Code

➤ File class Movie

```
1  namespace jurnal_modul9
2  {
3      9 references
4      public class Movie
5      {
6          3 references
7          public string Title { get; set; }
8          3 references
9          public string Director { get; set; }
10         3 references
11         public List<string> Stars { get; set; }
12         3 references
13         public string Description { get; set; }
14
15         3 references
16         public Movie() { }
17     }
18 }
```

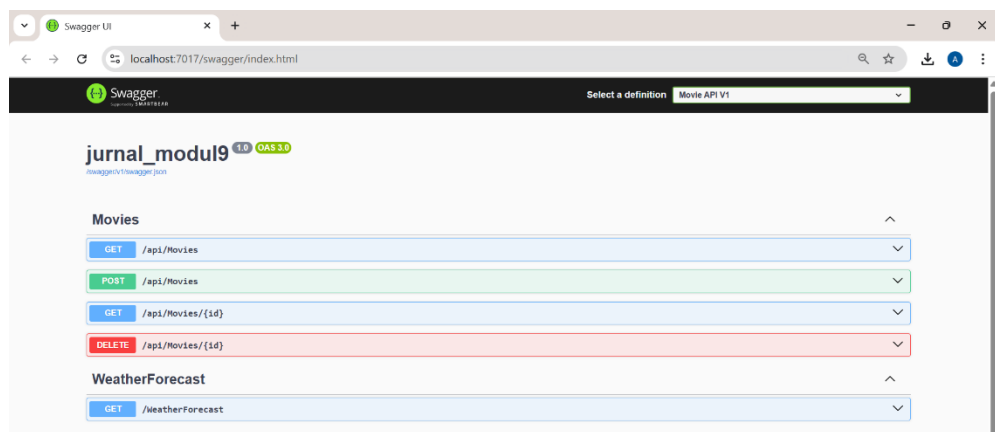
➤ File MoviesController.cs

```
1  using Microsoft.AspNetCore.Mvc;
2
3  namespace jurnal_modul9.Controllers
4  {
5      [ApiController]
6      [Route("api/[controller]")]
7      0 references
8      public class MoviesController : ControllerBase
9      {
10         private static List<Movie> Movies = new List<Movie>
11         {
12             new Movie {
13                 Title = "The Shawshank Redemption",
14                 Director = "Frank Darabont",
15                 Stars = new List<string> { "Tim Robbins", "Morgan Freeman" },
16                 Description = "Two imprisoned men bond over a number of years..."
17             },
18             new Movie {
19                 Title = "The Godfather",
20                 Director = "Francis Ford Coppola",
21                 Stars = new List<string> { "Marlon Brando", "Al Pacino" },
22                 Description = "The aging patriarch of an organized crime dynasty..."
23             },
24             new Movie {
25                 Title = "The Dark Knight",
26                 Director = "Christopher Nolan",
27                 Stars = new List<string> { "Christian Bale", "Heath Ledger" },
28                 Description = "When the menace known as the Joker wreaks havoc..."
29             }
30         };
31
32         [HttpGet]
33         0 references
34         public ActionResult<List<Movie>> GetAllMovies() => Movies;
35
36         [HttpGet("{id}")]
37         0 references
38         public ActionResult<Movie> GetMovieById(int id)
39         {
40             if (id < 0 || id >= Movies.Count)
41                 return NotFound();
42             return Movies[id];
43         }
44
45         [HttpPost]
46         0 references
47         public ActionResult AddMovie([FromBody] Movie movie)
48         {
49             Movies.Add(movie);
50             return Ok();
51         }
52
53         [HttpDelete("{id}")]
54         0 references
55         public ActionResult DeleteMovie(int id)
56         {
57             if (id < 0 || id >= Movies.Count)
58                 return NotFound();
59             Movies.RemoveAt(id);
60             return Ok();
61         }
62     }
63 }
```

➤ File Program.cs

```
1  var builder = WebApplication.CreateBuilder(args);
2
3  builder.Services.AddControllers();
4  builder.Services.AddEndpointsApiExplorer();
5  builder.Services.AddSwaggerGen();
6
7  var app = builder.Build();
8
9  app.UseSwagger();
10 app.UseSwaggerUI(c =>
11 {
12     c.SwaggerEndpoint("/swagger/v1/swagger.json", "Movie API V1");
13 });
14
15 app.UseHttpsRedirection();
16 app.UseAuthorization();
17 app.MapControllers();
18
19 app.Run();
20
```

➤ Hasil running program



➤ Penjelasan

Program API di atas merupakan aplikasi Web API berbasis C# yang dikembangkan menggunakan ASP.NET Core pada Visual Studio. API ini memanfaatkan atribut [ApiController] dan [Route] untuk menangani permintaan HTTP melalui endpoint /api/Movies. Seluruh data film disimpan dalam sebuah static List<Movie>, tanpa menggunakan database, sehingga data hanya tersimpan selama aplikasi berjalan. API ini menyediakan empat operasi utama: GET untuk mengambil semua data atau data berdasarkan indeks, POST untuk menambahkan data film baru, dan DELETE untuk menghapus film berdasarkan indeks tertentu. Selain itu, Swagger digunakan sebagai antarmuka dokumentasi dan pengujian endpoint API melalui halaman <https://localhost:<PORT>/swagger/index.html>.

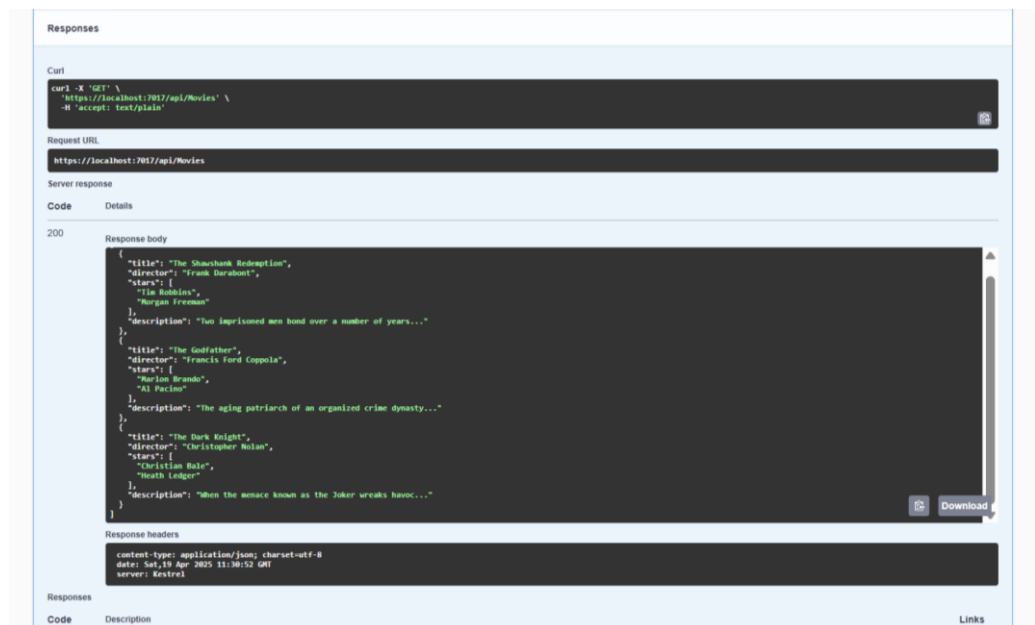
Dalam pengujian, terdapat beberapa skenario penting yang harus dilakukan untuk memastikan fungsionalitas API berjalan dengan baik. Pertama, memanggil GET /api/Movies untuk melihat daftar 3 film awal yang otomatis dimuat dari TOP 3 IMDb. Kedua, menambahkan film keempat melalui POST /api/Movies dengan data yang sesuai. Setelah penambahan, dilakukan GET /api/Movies kembali untuk memastikan film keempat telah tersimpan. Kemudian dilakukan GET /api/Movies/3 untuk mengambil data film baru berdasarkan index. Selanjutnya, DELETE /api/Movies/1 dijalankan untuk menghapus film "The Godfather", dan terakhir GET /api/Movies kembali

dipanggil untuk memastikan bahwa film tersebut sudah tidak ada di dalam daftar.

2. 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 skenario 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”



The screenshot displays a REST client interface with the following details:

- Request:**
 - Method: GET
 - URL: `https://localhost:7017/api/Movies`
 - Headers: `-H 'accept: text/plain'`
- Response:**
 - Status: 200
 - Body (JSON):

```
{
  "title": "The Shawshank Redemption",
  "director": "Frank Darabont",
  "stars": [
    "Tim Robbins",
    "Morgan Freeman"
  ],
  "description": "Two imprisoned men bond over a number of years..."
},
{
  "title": "The Godfather",
  "director": "Francis Ford Coppola",
  "stars": [
    "Marlon Brando",
    "Al Pacino"
  ],
  "description": "The aging patriarch of an organized crime dynasty..."
},
{
  "title": "The Dark Knight",
  "director": "Christopher Nolan",
  "stars": [
    "Christian Bale",
    "Heath Ledger"
  ],
  "description": "When the menace known as the Joker wreaks havoc..."
}
]
```
 - Headers:

```
content-type: application/json; charset=utf-8
date: Sat, 19 Apr 2025 11:30:52 GMT
server: Kestrel
```

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

POST /api/Movies

Parameters

No parameters

Request body

application/json

```
{
  "title": "Schindler's list",
  "director": "Steven Spielberg",
  "stars": [
    "Liam Neeson",
    "Ralph Fiennes",
    "Ben Kingsley"
  ],
  "description": "In German-occupied Poland during World War II, industrialist Oskar Schindler gradually becomes concerned for his Jewish workforce after witnessing their persecution by the Nazis."
}
```

Execute

Responses

Code	Description	Links
------	-------------	-------

Responses

Curl

```
curl -X 'POST' \
  https://localhost:7017/api/Movies \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
    "title": "Schindler\'s list",
    "director": "Steven Spielberg",
    "stars": [
      "Liam Neeson",
      "Ralph Fiennes",
      "Ben Kingsley"
    ],
    "description": "In German-occupied Poland during World War II, industrialist Oskar Schindler gradually becomes concerned for his Jewish workforce after witnessing their persecution by the Nazis."
  }'
```

Request URL

https://localhost:7017/api/Movies

Server response

Code	Details
200	Response headers content-length: 0 date: Sat, 19 Apr 2025 11:33:30 GMT server: Restful

Responses

Code	Description	Links
200	OK	No links

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

curl -X 'GET' \
 https://localhost:7017/api/Movies \
 -H 'accept: text/plain'

Request URL

https://localhost:7017/api/Movies

Server response

Code	Details
200	Response body { "description": "Two imprisoned men bond over a number of years..." }, { "title": "The godfather", "director": "Francis Ford Coppola", "stars": ["Marlon Brando", "Al Pacino"], "description": "The aging patriarch of an organized crime dynasty..." }, { "title": "The Dark Knight", "director": "Christopher Nolan", "stars": ["Christian Bale", "Heath Ledger"], "description": "When the menace known as the Joker wreaks havoc..." }, { "title": "Schindler's list", "director": "Steven Spielberg", "stars": ["Liam Neeson", "Ralph Fiennes", "Ben Kingsley"], "description": "In German-occupied Poland during World War II, industrialist Oskar Schindler gradually becomes concerned for his Jewish workforce after witnessing their persecution by the Nazis." }

Response headers

content-type: application/json; charset=utf-8
date: Sat, 19 Apr 2025 11:33:40 GMT
server: Restful

Responses

Code	Description	Links
200	OK	No links

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

GET /api/Movies/{id}

Parameters

Name	Description
id * required	integer(\$int32) (path)

Execute

Responses

Code	Description	Links
200	OK	No links

Media type: text/plain

Content Accept header:

Example Value | Schema

```
{  "title": "string",  "director": "string",  "stars": [    "string"  ],  "description": "string"}
```

DELETE /api/Movies/{id}

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

Parameters

Name	Description
id * required	integer(\$int32) (path)

Execute

Clear

Responses

Curl

```
curl -X 'DELETE' \  https://localhost:7017/api/Movies/1 \  -H 'accept: */*' \
```

Request URL

https://localhost:7017/api/Movies/1

Server response

Code	Details
200	Response headers content-length: 0 date: Sat, 19 Apr 2025 11:36:26 GMT server: Kestrel

Responses

Code	Description	Links
200	OK	No links

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

curl -X 'GET' \ https://localhost:7017/api/Movies \ -H 'accept: text/plain'

Request URL

https://localhost:7017/api/Movies

Server response

Code	Details
200	Response body [{ "title": "The Shawshank Redemption", "director": "Frank Darabont", "stars": ["Tim Robbins", "Morgan Freeman"], "description": "Two imprisoned men bond over a number of years..." }, { "title": "The Dark Knight", "director": "Christopher Nolan", "stars": ["Christian Bale", "Heath Ledger"], "description": "When the menace known as the Joker wreaks havoc..." }, { "title": "Schindler's List", "director": "Steven Spielberg", "stars": ["Liam Neeson", "Ralph Fiennes", "Ben Kingsley"], "description": "In German-occupied Poland during World War II, industrialist Oskar Schindler gradually becomes concerned for his Jewish workforce after witnessing their persecution..." }]

Response headers

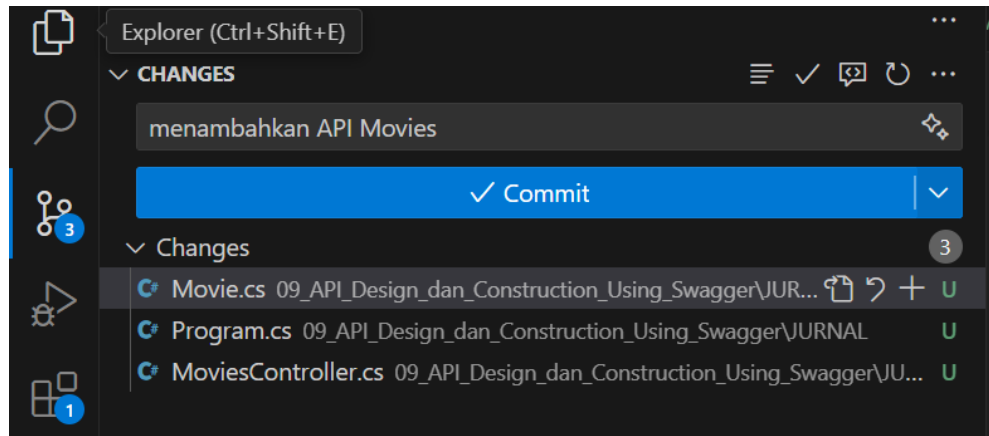
content-type: application/json; charset=utf-8
date: Sat, 19 Apr 2025 11:36:52 GMT
server: Kestrel

Responses

Code	Description	Links
200	OK	No links

3. MELAKUKAN COMMIT

A. Lakukan commit dengan pesan “menambahkan API Movies”.



B. Lakukan push ke github ke branch yang dibuat di bagian sebelumnya.

