

TUGAS JURNAL MODUL 9



Disusun Oleh :

Pradana Argo Pangestu

2311104079

Kelas :SE-07-02

Dosen :

Yudha Islami Sulistya

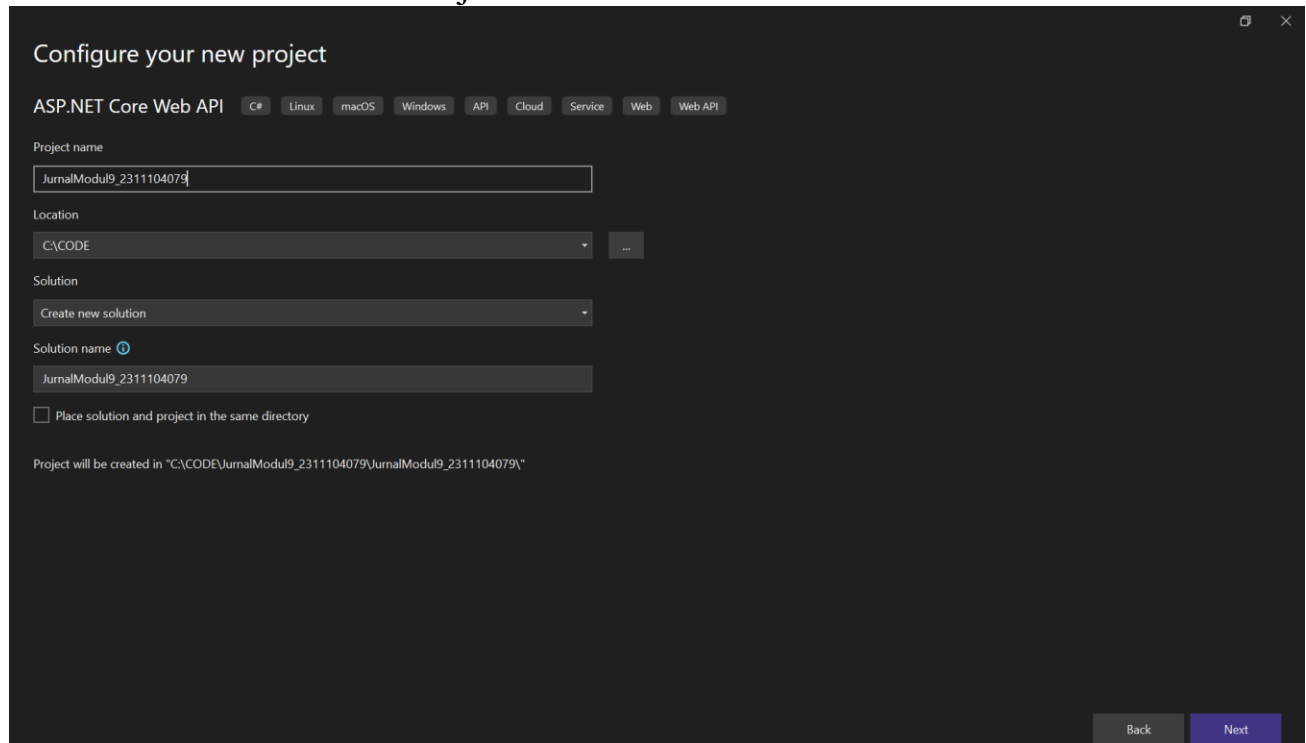
**PROGRAM STUDI SOFTWARE ENGINEERING
DIREKTORAT KAMPUS PURWOKERTO
TELKOM UNIVERSITY PURWOKERTO
2025**

I. Link Github

https://github.com/Pradana123/KPL_PradanaAP_2311104079/tree/main/09_API_Design_dan_Construction_Using_Swagger

I. Ss Output :

1. Membuat New Project



The screenshot shows the 'Configure your new project' window in Visual Studio. The 'Project name' field contains 'JurnalModul9_2311104079'. The 'Location' field shows 'C:\CODE'. The 'Solution' dropdown is set to 'Create new solution'. The 'Solution name' field also contains 'JurnalModul9_2311104079'. There is an unchecked checkbox for 'Place solution and project in the same directory'. At the bottom, it states 'Project will be created in "C:\CODE\JurnalModul9_2311104079\JurnalModul9_2311104079\"'. Navigation buttons 'Back' and 'Next' are at the bottom right.

2. Membuat Package Model yang dimana terdapat Class Movie.cs

a. Membuat File JSON "bank_transfer_config.json".

```
1. using System.Collections.Generic;
2.
3. namespace TJmodul9_2311104079.Models
4. {
5.     public class Movie
6.     {
7.         public string Title { get; set; } public string Director { get; set; }
8.         public List<string> Stars { get; set; } public string Description { get; set; }
9.     }
10. }
11. }
```

Penjelasan Singkat :

Bagian **model** dalam kode ini mendefinisikan sebuah **kelas bernama Movie** yang digunakan untuk merepresentasikan data sebuah film. Kelas ini berada di dalam **namespace TJmodul9_2311104079.Models** dan memiliki empat properti utama:

1. Title dan Director yang bertipe **string**, digunakan untuk menyimpan judul film dan nama sutradaranya.
2. Stars yang bertipe **List<string>**, digunakan untuk menyimpan daftar nama para pemeran dalam film.
3. Description yang juga bertipe **string**, digunakan untuk memberikan ringkasan atau penjelasan tentang isi film.

Model Movie ini memang sederhana, tetapi sangat penting karena menjadi struktur acuan untuk pertukaran data di dalam aplikasi, khususnya saat menerima atau mengirim data melalui **API**.

3. Menambah Class di Package Controller dengan nama MovieController.cs

```
1. using Microsoft.AspNetCore.Mvc; using TJmodul9_2311104079.Models; using
   System.Collections.Generic;
2.
3. namespace TJmodul9_2311104079.Controllers
4. {
5.     [ApiController] [Route("api/[controller]")]
6.     public class MoviesController : ControllerBase
7.     {
8.         private static List<movie> Movies = new List<movie>
9.         {
10.             new Movie
11.             {
12.                 Title = "The Shawshank Redemption", Director = "Frank Darabont",
13.                 Stars = new List<string> { "Tim Robbins", "Morgan Freeman" },
14.                 Description = "Two imprisoned men bond over a number of years, finding solace
15.                     and eventual redemption through acts of common decency."
16.             },
17.             new Movie
18.             {
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 transfers
23.                     control of his clandestine empire to his reluctant son."
24.             },
25.             new Movie
26.             {
27.                 Title = "The Dark Knight", Director = "Christopher Nolan",
28.                 Stars = new List<string> { "Christian Bale", "Heath Ledger" },
29.                 Description = "When the menace known as the Joker emerges, Batman must accept
30.                     one of the greatest psychological and physical tests."
31.             }
32.         };
33.         [HttpGet]
34.         public ActionResult<IEnumerable<movie>> GetMovies()
35.         {
36.             return Ok(Movies);
37.         }
38.         [HttpGet("{id}")]
```

```
36. public ActionResult<movie> GetMovie(int id)
37. {
38.     if (id < 0 || id >= Movies.Count)
39.     {
40.         return NotFound();
41.     }
42.     return Ok(Movies[id]);
43. }
44. [HttpPost]
45. public ActionResult<movie> AddMovie([FromBody] Movie movie)
46. {
47.     Movies.Add(movie);
48.     return CreatedAtAction(nameof(GetMovie), new { id = Movies.Count - 1 }, movie);
49. }
50. }
51. [HttpDelete("{id}")]
52. public IActionResult DeleteMovie(int id)
53. {
54.     if (id < 0 || id >= Movies.Count)
55.     {
56.         return NotFound();
57.     }
58.     Movies.RemoveAt(id); return NoContent();
59. }
60. }
61. }
```

Penjelasan Singkat:

Bagian **model** dalam kode ini berisi **kelas Movie** yang digunakan untuk merepresentasikan data sebuah film. Kelas ini berada di dalam **namespace TJmodul9_2311104079.Models** dan memiliki empat properti utama:

1. **Title** dan **Director** bertipe *string*, digunakan untuk menyimpan judul film dan nama sutradaranya.
2. **Stars** bertipe *List<string>*, yang memungkinkan penyimpanan beberapa nama pemeran dalam satu film.
3. **Description** juga bertipe *string*, digunakan untuk menuliskan ringkasan cerita atau informasi tambahan tentang film tersebut.

Meskipun sederhana, model ini sangat penting karena menjadi acuan utama dalam pertukaran data antar bagian aplikasi, khususnya saat menerima atau mengirim data melalui **API**.

4. Class Program

```
1. var builder = WebApplication.CreateBuilder(args);
2.
```

```
3. // Add services to the container. builder.Services.AddControllers();  
   builder.Services.AddEndpointsApiExplorer();  
   builder.Services.AddSwaggerGen();  
4.  
5. builder.Services.AddControllers();  
6. // Learn more about configuring Swagger/OpenAPI at  
7. https://aka.ms/aspnetcore/swashbuckle  
   builder.Services.AddEndpointsApiExplorer();  
   builder.Services.AddSwaggerGen();  
8. var app = builder.Build();  
9. // Configure the HTTP request pipeline. if  
   (app.Environment.IsDevelopment())  
10. {  
11.     app.UseSwagger(); app.UseSwaggerUI();  
12. }  
13. app.UseAuthorization(); app.MapControllers(); app.Run();
```

Penjelasan Singkat:

File **Program.cs** merupakan **titik awal (entry point)** dari aplikasi ASP.NET Core. File ini digunakan untuk mengatur **konfigurasi layanan** dan **alur HTTP aplikasi**.

1. Baris `WebApplication.CreateBuilder(args)` digunakan untuk membuat objek builder, yang menyiapkan segala kebutuhan aplikasi.
2. Kemudian, `builder.Services.AddControllers()` menambahkan layanan controller agar framework bisa mengenali dan menjalankan endpoint dari controller yang telah dibuat.
3. Layanan `AddSwaggerGen()` diaktifkan untuk menambahkan fitur **Swagger**, yaitu dokumentasi otomatis dan antarmuka web yang bisa digunakan untuk menguji API.

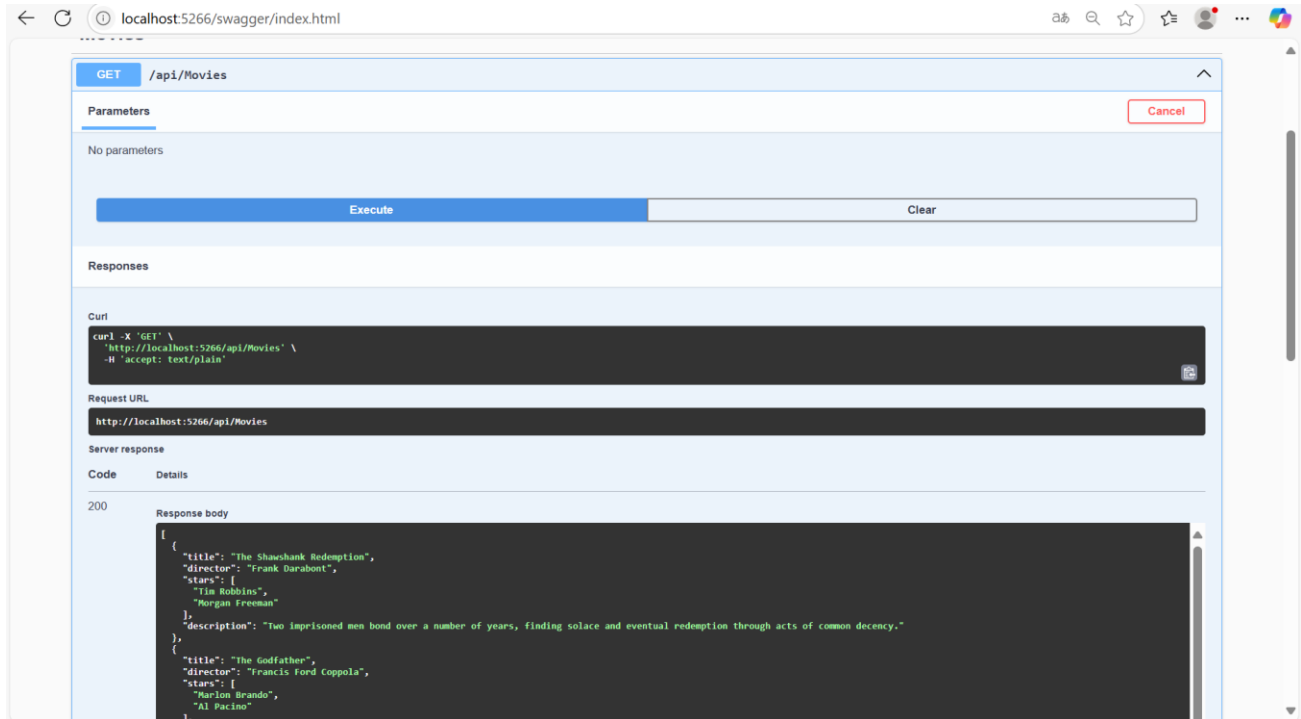
Setelah konfigurasi selesai, aplikasi dibangun dengan `builder.Build()`.

4. Pada bagian `if (app.Environment.IsDevelopment())`, Swagger dan Swagger UI hanya diaktifkan jika aplikasi berjalan dalam **mode pengembangan**, sehingga API bisa diuji langsung di browser.
5. Middleware `app.UseAuthorization()` disiapkan untuk memproses otorisasi, walaupun belum digunakan dalam API ini.
6. `app.MapControllers()` digunakan untuk **memetakan semua rute controller**, agar bisa merespons permintaan HTTP.
7. Terakhir, `app.Run()` menjalankan aplikasi sehingga bisa diakses melalui browser atau alat seperti **Postman**.

I. Hasil Running

1. Hasil Running

- **Get**



- **Post**

POST /api/Movies

Parameters Cancel Reset

No parameters

Request body application/json

```
{  "title": "Final Destination",  "director": "Pradana Argo Pangestu",  "stars": [    "Pradana Argo"  ],  "description": "Film mematkan"}
```

Execute Clear

Responses

Curl

```
curl -X 'POST' \  'http://localhost:5266/api/Movies' \  -H 'accept: text/plain' \  -H 'Content-Type: application/json' \  -d '{  "title": "Final Destination",  "director": "Pradana Argo Pangestu",  "stars": [    "Pradana Argo"  ]}'
```

- **Get Cari Berdasarkan ID**

localhost:5266/swagger/index.html

GET /api/Movies/{id}

Parameters

Name Description

id * required
integer(int32) 3
(path)

Execute Clear

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:5266/api/Movies/3' \
  -H 'accept: text/plain'
```

Request URL

http://localhost:5266/api/Movies/3

Server response

Code Details

200

Response body

```
{
  "title": "Final Destination",
  "director": "Pradana Argo Pangestu",
  "stars": [
    "Pradana Argo"
  ],
  "description": "film mematkan"
}
```

Download

• Delete

DELETE /api/Movies/{id}

Parameters

Name Description

id * required
integer(int32) 1
(path)

Execute Clear

Responses

Curl

```
curl -X 'DELETE' \
  'http://localhost:5266/api/Movies/1' \
  -H 'accept: */*'
```

Request URL

http://localhost:5266/api/Movies/1

Server response

Code Details

204
Undocumented

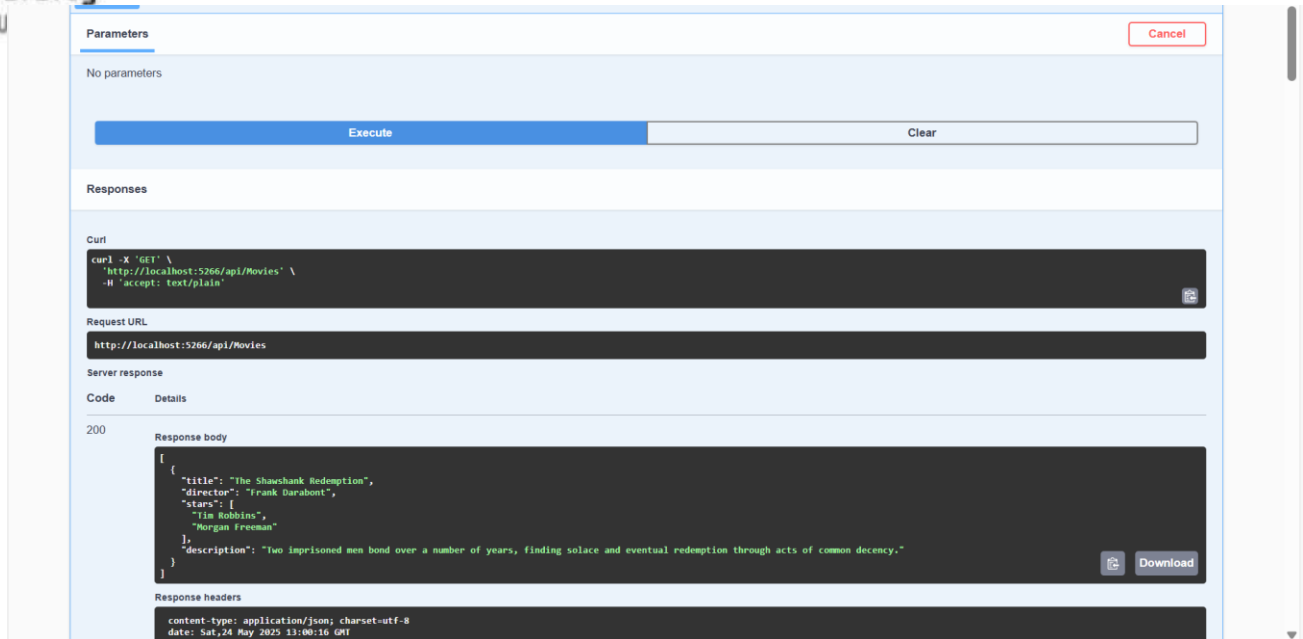
Response headers

```
date: Sat, 24 May 2025 12:59:11 GMT
server: Kestrel
```

Responses

Code Description Links

200 OK No links



II. Kesimpulan

Praktikum ini merupakan **penerapan dasar RESTful API** menggunakan **ASP.NET Core**, yang dibangun dari tiga komponen utama: **model**, **controller**, dan **konfigurasi aplikasi**.

- **Model Movie** digunakan untuk merepresentasikan data film secara sederhana namun lengkap. Model ini mencakup properti seperti **judul**, **sutradara**, **daftar pemeran**, dan **deskripsi film**.
- **Controller MoviesController** bertugas menangani permintaan HTTP, seperti:
 - Menampilkan semua data film,
 - Mengambil data film berdasarkan indeks,
 - Menambahkan film baru,
 - Menghapus film berdasarkan indeks.

Semua data film disimpan dalam **list statis**, tanpa menggunakan database.

- **File Program.cs** mengatur konfigurasi aplikasi. Di dalamnya, layanan controller diaktifkan, **Swagger** dihidupkan untuk dokumentasi dan pengujian API, dan alur pemrosesan permintaan diatur agar API dapat berjalan dengan baik.

Secara keseluruhan, proyek ini memberikan **dasar yang kuat** untuk memahami cara membangun dan mengelola **Web API sederhana** di ASP.NET Core. Ke depannya, proyek ini bisa dikembangkan lebih lanjut dengan menambahkan fitur seperti **penyimpanan data ke database** dan **autentikasi pengguna**.

