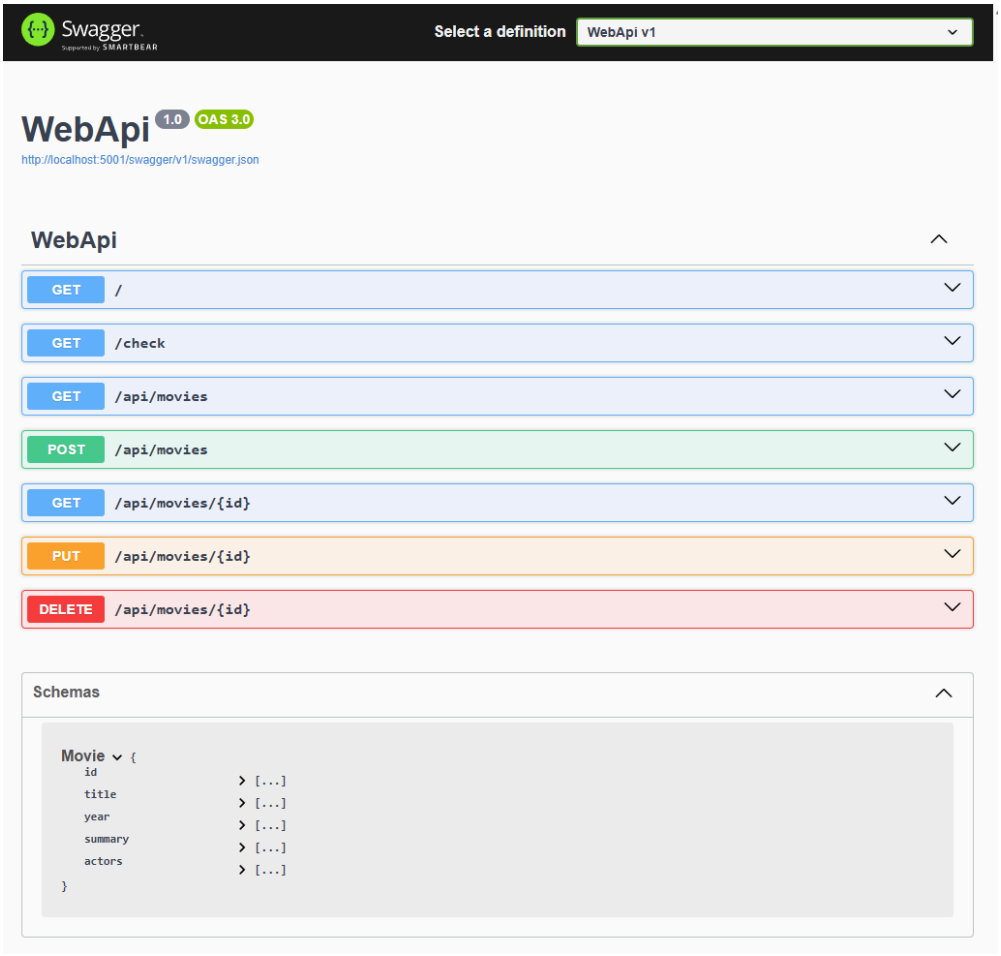
7.4-5 MovieService mit Swagger (Zusatzaufgabe)

Installieren Sie die beiden Nuget-Package mit folgenden .NET CLI Commands:

```
dotnet add package Microsoft.AspNetCore.OpenApi --version 8.0.8
dotnet add package Swashbuckle.AspNetCore
```

Integrieren Sie OpenAPI anhand der Dokumentation OpenAPI support in minimal API apps in Ihre Anwendung. Sie ist per Default über den Endpunkt **http://localhost:5001/swagger** erreichbar.

💡 Generate OpenAPI documents | Microsoft Learn



Program.cs

```
using MongoDB.Driver;
using WebApi;
using Microsoft.AspNetCore.OpenApi;

var builder = WebApplication.CreateBuilder(args);

var movieDatabaseCollection = builder.Configuration.GetSection("DatabaseSettings");
builder.Services.Configure<DatabaseSettings>(movieDatabaseCollection);
builder.Services.AddSingleton<IMovieService, MongoMovieService>();
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen();

var app = builder.Build();

if (app.Environment.IsDevelopment())
{
   app.UseSwagger();
   app.UseSwaggerUI();
}

app.UseHttpsRedirection();

app.MapGet("/", () => "Minimal API Version 1.0");

app.MapGet("/check", (Microsoft.Extensions.Options.IOptions<DatabaseSettings> options) =>
{
   try
   {
      var mongoClient = new MongoClient(options.Value.ConnectionString);
      var dbs = mongoClient.ListDatabaseNames().ToList();

      return $"Zugriff auf MongoDB ok. Vorhandene DBs: " + String.Join(", ", dbs);
   }
   catch
   {
      return "Leider hat Zugriff auf MongoDB nicht geklappt.";
   }
});

// Get all Movies
// Gibt alle vorhandenen Movie-Objekte mit Statuscode 200 OK zurück.
app.MapGet("/api/movies", (IMovieService movieService) =>
{
   IEnumerable<Movie> movies = movieService.Get();
   return Results.Ok(movies);
})
.WithName("Get all movies")
```

```
.WithOpenApi();

// Get Movie by id
// Gibt das gewünschte Movie-Objekt mit Statuscode 200 OK zurück.
// Bei ungültiger id wird Statuscode 404 not found zurückgegeben.
app.MapGet("/api/movies/{id}", (IMovieService movieService, string id) =>
{
    var movie = movieService.Get(id);
    return movie != null
        ? Results.Ok(movie)
        : Results.NotFound();
})
.WithName("Get movie by ID")
.WithOpenApi();

// Insert Movie
// Wenn das übergebene Objekt eingefügt werden konnte,
// wird es mit Statuscode 200 zurückgegeben.
// Bei Fehler wird Statuscode 409 Conflict zurückgegeben.
app.MapPost("/api/movies", (IMovieService movieService, Movie movie) =>
{
    if (String.IsNullOrWhiteSpace(movie.Id) || String.IsNullOrWhiteSpace(movie.Title))
    {
        return Results.Conflict("ID and Title can't be null.");
    }
    else if (movieService.Get(movie.Id) == null)
    {
        movieService.Create(movie);
        return Results.Ok(movie);
    }
    else
    {
        return Results.Conflict($"Movie with ID {movie.Id} already exists.");
    }
})
.WithName("Create new movie")
.WithOpenApi();

// Update Movie
// Gibt das aktualisierte Movie-Objekt zurück.
// Bei ungültiger id wird Statuscode 404 not found zurückgegeben.
app.MapPut("/api/movies/{id}", (IMovieService movieService, string id, Movie movie) =>
{
    if (movieService.Get(id) != null)
    {
        movieService.Update(id, movie);
        return Results.Ok();
    }
    else
    {
        return Results.NotFound();
```

```
    }
})
.WithName("Update movie")
.WithOpenApi();

// Delete Movie
// Gibt bei erfolgreicher Löschung Statuscode 200 OK zurück.
// Bei ungültiger id wird Statuscode 404 not found zurückgegeben.
app.MapDelete("/api/movies/{id}", (IMovieService movieService, string id) =>
{
    if (movieService.Get(id) != null)
    {
        movieService.Delete(id);
        return Results.Ok();
    }
    else
    {
        return Results.NotFound();
    }
})
.WithName("Delete movie")
.WithOpenApi();


app.Run();
```