

## Interface

```
using Day12.DTO;
using Day12.Models;

namespace Day12.Interfaces
{
    public interface IMovieService
    {
        string AddSales(SalesDTO salesDto);
        IEnumerable<NewMovie> GetAllNewMovies();
        IEnumerable<NewMovie> GetAllNewMoviesByDirector(string director);
        string AddDirector(Director director);
        string AddNewMovie(MovieDirectorEntry movieDirectorEntry);
        string AddMovie(Movie movie);
        IEnumerable<Movie> GetMovies();
        Movie? GetById(int movieId);
        string DeleteById(int movieId);
        IEnumerable<Movie> GetMoviesOrderedByCollection(bool isAscending);
        IEnumerable<Movie> GetTop5Collection();
        object GetCollectionDetails();
        IEnumerable<object> GetByJoin();
    }
}
```

## Controller

```
using Day12.DTO;
using Day12.Interfaces;
using Day12.Models;
using Day12.Services;
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Mvc;

namespace Day12.Controllers
{
    [ApiController]
    [Route("[controller]")]
    public class MovieController : ControllerBase
    {
        private readonly IMovieService _movieService;

        public MovieController(IMovieService movieService)
        {

```

```
        _movieService = movieService;
    }

    [HttpPost("AddNewMovieSales")]
    public IActionResult AddSales(SalesDTO sales_dto)
    {
        var result = _movieService.AddSales(sales_dto);
        if (result == "Movie Not found") return NotFound(result);
        return Ok(result);
    }

    [HttpGet("GetAllNewMovies")]
    public IActionResult GetAllNewMovies() => Ok(_movieService.GetAllNewMovies());

    [HttpGet("FetchAllNewMoviesByAuthor")]
    public IActionResult GetAllNewMoviesByAuthor(string director) =>
    Ok(_movieService.GetAllNewMoviesByDirector(director));

    [HttpPost("AddDirector")]
    public IActionResult AddDirector(Director director) =>
    Ok(_movieService.AddDirector(director));

    [HttpPost("AddNewMovie")]
    public IActionResult AddNewMovie(MovieDirectorEntry entry)
    {
        var result = _movieService.AddNewMovie(entry);
        if (result == "Invalid Director") return NotFound(result);
        return Ok(result);
    }

    [HttpPost("AddMovie")]
    public IActionResult AddMovie(Movie movie) => Ok(_movieService.AddMovie(movie));

    [HttpGet("GetMovies")]
    public IActionResult GetMovies() => Ok(_movieService.GetMovies());

    [HttpGet("GetbyID")]
    public IActionResult GetByld(int movieId)
    {
        var movie = _movieService.GetByld(movieId);
        if (movie == null) return NotFound("Movie not found");
        return Ok(movie);
    }
}
```

```

[HttpDelete("DeleteById")]
public IActionResult DeleteMovie(int movieId)
{
    var result = _movieService.DeleteById(movieId);
    if (result == "Movie not found") return NotFound(result);
    return Ok(result);
}

[HttpPost("GetBooksOrderedByCollection")]
public IActionResult GetMoviesOrderedByCollection([FromBody] bool isAscending) =>
    Ok(_movieService.GetMoviesOrderedByCollection(isAscending));

[HttpGet("GetTop5Collection")]
public IActionResult GetTop5Collection() => Ok(_movieService.GetTop5Collection());

[HttpGet("GetCollectionDetails")]
public IActionResult GetCollectionDetails() => Ok(_movieService.GetCollectionDetails());

[HttpGet("GetByJoin")]
public IActionResult GetByJoin() => Ok(_movieService.GetByJoin());
}
}

```

## Service

```

using Day12.Context;
using Day12.DTO;
using Day12.Interfaces;
using Day12.Models;
using Microsoft.EntityFrameworkCore;

namespace Day12.Services
{
    public class MovieService : IMovieService
    {
        private readonly MyAppDbContext _appDbContext;

        public MovieService(MyAppDbContext appDbContext)
        {
            _appDbContext = appDbContext;
        }

        public string AddSales(SalesDTO sales_dto)

```

```
{
    var movie = _appDbContext.NewMovies.FirstOrDefault(x => x.NewMovieId ==
sales_dto.NewMovieId);
    if (movie == null) return "Movie Not found";

    Sales sale = new Sales() { NewMovieId = movie.NewMovieId, Num_of_Days =
sales_dto.Num_Of_Days, year = sales_dto.Year };
    _appDbContext.MovieSaless.Add(sale);
    _appDbContext.SaveChanges();
    return "Sales Details Added Successfully";
}

public IEnumerable<NewMovie> GetAllNewMovies()
{
    return _appDbContext.NewMovies.Include(x => x.director).ToList();
}

public IEnumerable<NewMovie> GetAllNewMoviesByDirector(string director)
{
    return _appDbContext.NewMovies.Include(x => x.director)
        .Where(y => y.director.DirectorName == director)
        .ToList();
}

public string AddDirector(Director director)
{
    _appDbContext.Directors.Add(director);
    _appDbContext.SaveChanges();
    return "Director Added";
}

public string AddNewMovie(MovieDirectorEntry movieDirectorEntry)
{
    var dir = _appDbContext.Directors.FirstOrDefault(x => x.DirectorName ==
movieDirectorEntry.DirectorName);
    if (dir == null) return "Invalid Director";

    NewMovie newMovie = new NewMovie()
    {
        MovieName = movieDirectorEntry.MovieName,
        Collection = movieDirectorEntry.Collection,
        DirectorId = dir.DirectorId
    };
    _appDbContext.NewMovies.Add(newMovie);
}
```

```
        _appDbContext.SaveChanges();
        return "Movie Added";
    }

    public string AddMovie(Movie movie)
    {
        _appDbContext.Movies.Add(movie);
        _appDbContext.SaveChanges();
        return "Movie Added";
    }

    public IEnumerable<Movie> GetMovies()
    {
        return _appDbContext.Movies.ToList();
    }

    public Movie? GetById(int movieId)
    {
        return _appDbContext.Movies.FirstOrDefault(x => x.Id == movieId);
    }

    public string DeleteById(int movieId)
    {
        var movie = _appDbContext.Movies.FirstOrDefault(x => x.Id == movieId);
        if (movie == null) return "Movie not found";

        _appDbContext.Movies.Remove(movie);
        _appDbContext.SaveChanges();
        return "Movie Deleted";
    }

    public IEnumerable<Movie> GetMoviesOrderedByCollection(bool isAscending)
    {
        return isAscending
            ? _appDbContext.Movies.OrderBy(m => m.Collection).ToList()
            : _appDbContext.Movies.OrderByDescending(m => m.Collection).ToList();
    }

    public IEnumerable<Movie> GetTop5Collection()
    {
        return _appDbContext.Movies.OrderByDescending(m => m.Collection).Take(5).ToList();
    }

    public object GetCollectionDetails()
```

```
{
    var totalCollection = _appDbContext.Movies.Sum(m => m.Collection);
    var avgCollection = _appDbContext.Movies.Any() ?
Math.Round(_appDbContext.Movies.Average(m => m.Collection), 2) : 0;

    return new
    {
        TotalCollection = totalCollection + " Cr",
        AverageCollection = avgCollection
    };
}

public IEnumerable<object> GetByJoin()
{
    var joined = _appDbContext.NewMovies.Join(_appDbContext.Directors,
        m => m.DirectorId,
        d => d.DirectorId,
        (m, d) => new { m.MovieName, d.DirectorName });

    return joined.ToList();
}
}
```

## Output

Request URL

`https://localhost:7196/Movie/GetMovies`

Server response

Code	Details
200	<p>Response body</p> <pre>[   {     "id": 1,     "movieName": "Leo",     "director": "Lokesh",     "rating": 5,     "collection": 650   },   {     "id": 2,     "movieName": "Mersal",     "director": "Atlee",     "rating": 5,     "collection": 400   } ]</pre> <p> Download</p>

Request URL

https://localhost:7196/Movie/GetAllNewMovies

Server response

Code

Details

200

Response body

```
[
  {
    "newMovieId": 1,
    "movieName": "Leo",
    "collection": 650,
    "directorId": 1,
    "director": {
      "directorId": 1,
      "directorName": "Lokesh"
    }
  },
  {
    "newMovieId": 2,
    "movieName": "Beast",
    "collection": 450,
    "directorId": 2,
    "director": {
      "directorId": 2,
      "directorName": "Nelson"
    }
  }
]
```

 Download