# ASP.NET Core and EF Core

## Materials

- Introduction to ASP.NET Core
- ASP.NET Core fundamentals
- Overview of ASP.NET Core MVC
- Create web APIs with ASP.NET Core
- Entity Framework Core
- Angular
- Azure
- Microservices Architecture Guidance

---

**NOTE: Please reference the attached ScreenShots for UI**

## Introduction to ASP.NET Core and Onion/Clean Architecture - Part 1

**Homework Assignment** **Days to Finish - 1**

1. Create the architecture of the project that include ApplicationCore, Infrastructure and MVC
2. Create required Repository and Services Interfaces and their implementation classes.
3. Create MovieService class that will return some test MovieCard model to display on home page.
4. Create MovieCardModel to demonstrate Model use in Razor View
5. Create MovieCard partial view to demonstrate how to re-use partial views
6. Check in the code in to GitHub with all the necessary code that include complete architecture as demonstrated in class
7. Create MoviesController, UserController, AdminController, AccountController and CastController
8. Create IMovieRepository, IUserRepository, ICastRepository, IPurchaseRepository, IReportRepository interfaces in ApplicationCore/Contracts/Repository folder
9. Create IMovieService, IUserService, IAccountService, IGenreService, IAdminService, ICastService, IMovieService interfaces in ApplicationCore/Contracts/Services folder
10. Create MovieRepository, UserRepository, CastRepository, PurchaseRepository, ReportRepository classes in Infrastructure/Repository folder
11. Create MovieService, UserService, AccountService, GenreService, AdminService, CastService, MovieService classes in Infrastructure/Services folder

**Reading Material**

- Overview of ASP.NET Core MVC
- Controllers
- Routing to controller actions
- Views in ASP.NET Core MVC
- Partial Views
- Razor Syntax
- Tag Helpers
- Dependency injection in ASP.NET Core
- Dependency injection into controllers in ASP.NET Core
- Architectural principles
- Common web application architectures
- Model Binding

**FAQ**

1. What are differences between .NET Framework and .NETCore.
2. Difference between asp.net Core 8 and asp.net Core 5 project files, stratup.cs and program.cs
3. What is Dependency Injection and what are different scopes (Transient, Singleton, Scoped) in asp.net core Dependency Injection?
4. What is Routing and how can you configure routing in asp.net core. Difference between convention-based/traditional routing and attribute-based routing ?
5. Explain MVC Pattern and what are advantages of MVC Pattern?
6. How do you pass data from Controller to View and from View to Controller? Explain ViewBag, ViewData and Models/ViewModels
7. What is Razor syntax and how does it help developer?
8. Do you know what is model binding in asp.net core, how is it useful for developers?

---

## Entity Framework Core Code First Approach - Part 2

**Homework Assignment** **Days to Finish - 2**

1. Create MovieShopDbContext and some Entities such as Genre, Movie, Trailer, MovieGenre, User, Cast, MovieCast.
2. Demonstrate use of both Data-annotations and Fluent API
3. Insert data into Genre, Movie, MovieGenre, User, Cast and MovieCast tables
4. Add GetHighestGrossingMovies() method in IMovieRepository and implement it using EF Core

Linq Query.

5. Implement GetMoviebyId(int id) method to demonstrate how to query multiple DbSets with Include method.
6. Create GetMovieDetails(int id) method that will return movie details.
7. Create Details action method in MoviesController that will return Details View as per requirement.
8. Create IRepository<T> interface and implement Generic Repository<T>
9. Finish creating the remaining tables in the Database, Create Role, UserRole, Review, Purchase, Favorite, Crew and MovieCrew tables.
10. Create Details view for displaying Movie Details information. Movie Details page display complete details of a particular movie, such as Movie Facts, Trailers, Casts, Genres etc.
    1. Use Bootstrap row to create two rows with first row containing 3 columns Grid system
        1. First Column should hav movie poster
        2. Second column should have Movie Details such as title, genres and rating using Badges, overview etc.
        3. Third column should have buttons such as buy Movie, Review etc.
        4. When clicked on buy Movie or Review Buttons a bootstrap modal should popup.
        5. For Buy Movie popup a confirmation page along with its Price should display with Purchase Button.
        6. For Review popup a page with a dropdown for rating from 1to 10 and a text area for user to write review of the movie along with Submit button should be there.
    2. Second row should have 2 columns with first column having two parts
        1. First Column should have Movie Facts using Bootstrap List group, below it should have list of trailers for the movie using List group
        2. Second column should display Casts belonging to that Movie using List group, when clicked on cast, should navigate to cast details page.
11. Override GetById(int id) method in CastRepository class that will return Movies belonging to the cast including cast details.
12. Create GetCastDetails method in ICastService that should be implemented in CastService that will call CastRepository using DI.
13. Create Details method in CastController that will return Details View.
14. Create Details View for Cast that will display cast information from Cast Table and also display Movies belonging to that particular cast

**Reading Material**

1. EF Core Properties
2. EF Core Relationships

3. EF Core Keys
4. EF Core Migrations
5. Ef Core Managing Migrations
6. EF Core Applying Migrations
7. EF Core Querying Data
8. EF Core Eager Loading
9. EF Core Lazy Loading
10. EF Core Change Tracking
11. EF Core Saving Data
12. EF Core Saving Related Data
13. EF Core Saving Disconnected entities
14. EF Core - How Queries Work

**FAQ**

1. What is an ORM? What are the advantages of Entity Framework over ado .net ?
2. Disadvantages of EF and how would you improve the performances of EF?
3. What are the different approaches you can use in EF and which approach did you use & why?
4. Do you have experience with any other ORMs such as Dapper?
5. Explain what are main differences between Dapper and Entity Framework and which one would you prefer in what scenarios?
6. Explain the steps of code first approach with migrations.
7. What are DbSet and DbContext classes in Entity Framework?
8. What is Fluent API in and how is it different from Data annotations ?
9. What difference does .AsNoTracking() make?
10. When would you use Skip() and Take() methods in Entity Framework ?
11. What is the difference between lazy loading and eager loading? What is N + 1 problem. Which one did you use in your projects?
12. How would you see the SQL queries generated by Entity Framework? What tools or coding would you implement?
13. How do you disable lazy loading in Entity Framework and what is the use of virtual keyword?

---

## Entity Framework Core & async/await - Part 3

**Homework Assignment** **Days to Finish - 1**

1. Refactor all Repository, Services and Controllers method to use async/await pattern with Task.
2. Create GenresViewComponent to demonstrate View Components in ASP.NET MVC and display

genres as dropdown in Layout page.
3. Create GetMoviesByGenre with Pagination in MovieRepository class that returns Movies belonging a particular Genre.
4. Change all synchronous methods that have been created to use async/await pattern with Task, that include Repositories, Services and Controller action methods.
5. Create MoviesByGenre(int id, int pageSize = 30, int pageNumber = 1) method in the MoviesController that will return PaginatedResultSet<`Movie`> to the View.
6. Create the View for displaying Movies By Genre with Pagination that has Previous and Next Buttons. All Movies should display MovieCard that uses MovieCard partial View.
7. Create a console app to practice the asynchronous tasks
    1. Process asynchronous tasks as they complete
    2. Asynchronous file access

**Reading Material**

1. View components
2. Pagination
3. async/await
4. Real-world Asynchronous programming
5. Async return types

**FAQ**

1. Explain what async/await does how asynchronous programming is different from Multithreaded programming.
2. Explain the scenarios where would you use Task.WhenAny() and Task.WhenAll() ?
3. What are various async return types in C#
4. How would you do Server-side Pagination using Entity Framework?

---

## Azure Introduction, DevOps and Azure Pipelines - Part 4

**Homework Assignment** **Days to Finish - 1**

1. Create Azure Pipeline for MovieShop MVC Application.
2. Setup CI/CD for MovieShop App
3. Setup Azure SQL for MovieShop Database.
4. Test CI/CD with changes.
5. Complete Azure CI/CD Setup for MovieShop MVC App including using Azure SQL for database.

6. Finish the following Azure exercises.
    1. Introduction to Azure fundamentals
    2. Azure Fundamentals part 1: Describe core Azure concepts
    3. Explore core data concepts
    4. Explore relational data in Azure
    5. Get started with Azure DevOps
    6. Create a build pipeline with Azure Pipelines
    7. Create a release pipeline in Azure Pipelines
    8. Migrate an ASP.NET web application to Azure with Visual Studio

**Reading Material**

1. Azure Intro
2. Azure Developer Guide
3. Azure App Service
4. Continuous deployment to Azure App Service
5. What is Azure Pipelines ?
6. DevOps for ASP.NET Core Developers
7. Create your first pipeline

**FAQ**

1. What is your experience with Azure, what Azure services have you used?
2. What is your experience with Azure DeOps CI/CD ?
3. Can you tell me some Azure acronyms, IaaS, SaaS, PaaS and serverless?

---