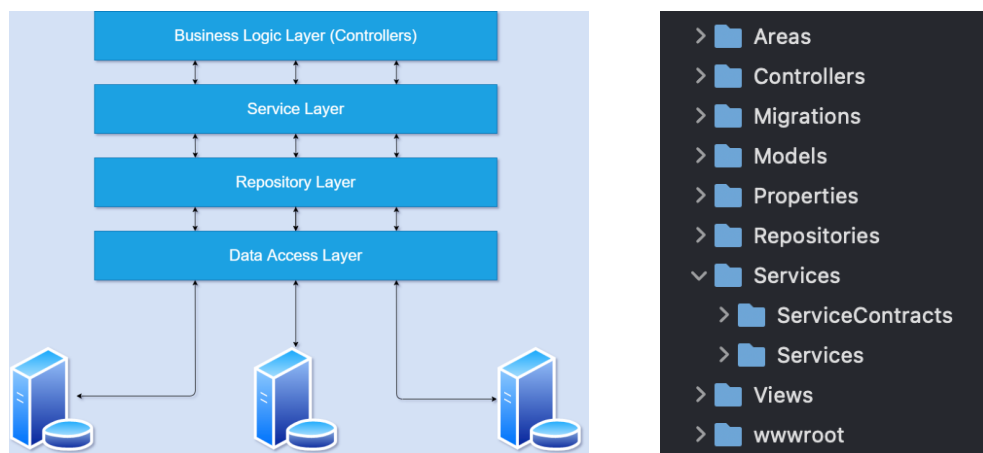


Frameworks de développement

TP4 Changement de l'architecture et personnalisation Layout

1. Architecture à base de Services

- On va à présent garder l'architecture MVC mais en découplant notre solution sur la base d'un ensemble de services. Pour ce faire, veuillez suivre l'architecture et l'arborescence suivantes.



N'oubliez pas d'injecter les services dans program.cs
`builder.Services.AddScoped<IMovieService,MovieService>();`

2. LINQ

LINQ (Language-Integrated Query) est le nom d'un ensemble de technologies basées sur l'intégration de fonctions de requête directement dans le langage C#. Vous pouvez écrire des requêtes LINQ en C# pour des bases de données SQL Server, des documents XML, des jeux de données ADO.NET et toute collection d'objets prenant en charge IEnumerable ou l'interface générique IEnumerable. La prise en charge LINQ est également fournie par des tierces parties pour de nombreux services web et autres implémentations de base de données.

Ce qu'il faut retenir des LINQ Queries :

- ✓ Les expressions de requête peuvent être utilisées pour interroger et transformer des données à partir de n'importe quelle source de données compatible LINQ. Par exemple, une même requête peut récupérer des données d'une base de données SQL et générer un flux XML en sortie.

- ✓ Les expressions de requête sont faciles à maîtriser, car elles utilisent de nombreuses constructions familières du langage C#.

- ✓ Les variables d'une expression de requête sont toutes fortement typées, même si, dans de nombreux cas, il n'est pas nécessaire de fournir le type explicitement, car le compilateur peut le déduire.

✓ Une requête ne s'exécute pas tant que vous n'avez pas itéré la variable de requête, par exemple dans une instruction foreach.

✓ En règle générale, lorsque vous écrivez des requêtes LINQ, nous vous recommandons d'utiliser la syntaxe de requête dans la mesure du possible et la syntaxe de méthode si nécessaire. Il n'y a aucune différence de sémantique ou de performances entre les deux formats. Les expressions de requête sont souvent plus lisibles que les expressions équivalentes écrites avec la syntaxe de méthode.

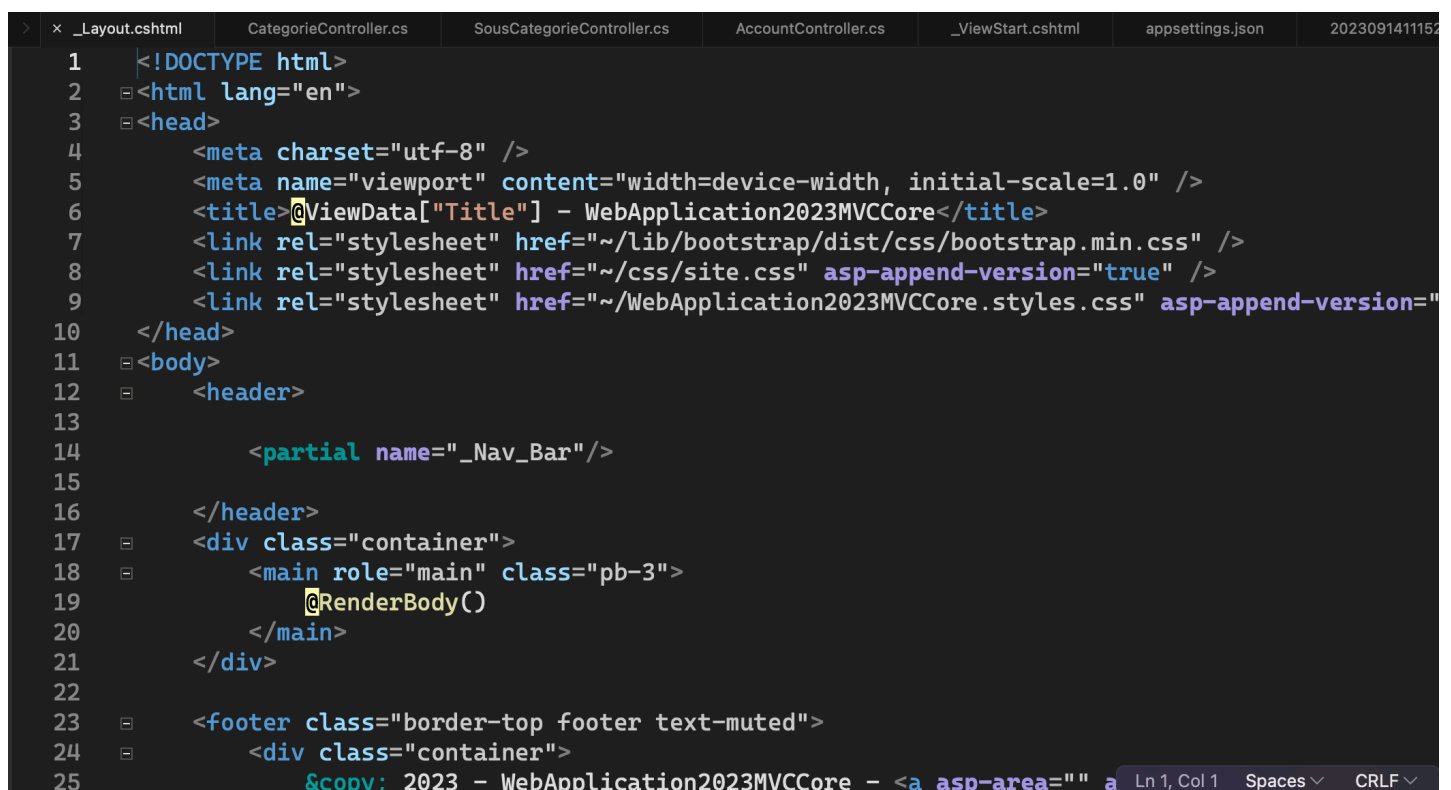
✓ Certaines opérations de requête, comme Count ou Max, n'ont pas d'expression de requête équivalente et doivent par conséquent être exprimées sous la forme d'un appel de méthode. La syntaxe de méthode peut être combinée avec la syntaxe de requête de différentes manières.

- Ecrire un service qui permet de lister tous les films associés à un genre définie par l'utilisateur avec une requête LINQ (dot notation syntax) permettant d'assurer l'extraction des données.
- Ecrire un service qui permet de lister tous les films ordonnés dans l'ordre croissant
- Ecrire un service permettant de récupérer les films par leur genre ID

3. Personnalisation Du Layout

Tout d'abord, on va modifier le lien « Nom de l'Application » en haut de la page. Ce texte est commun à chaque page. Il fait partie du Layout.

• Allez dans le dossier `/Views/Shared` dans L'Explorateur de solutions et ouvrez le fichier `_Layout.cshtml`. Ce fichier est le layout qui est utilisé par toutes les autres pages. Le layout permet de préciser le Html du conteneur dans un seul endroit. Localiser `@RenderBody()`. `RenderBody` permet d'injecter les vues « encapsulées » dans le conteneur.



```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="utf-8" />
5     <meta name="viewport" content="width=device-width, initial-scale=1.0" />
6     <title>@ViewData["Title"] - WebApplication2023MVCCore</title>
7     <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
8     <link rel="stylesheet" href="~/css/site.css" asp-append-version="true" />
9     <link rel="stylesheet" href="~/WebApplication2023MVCCore.styles.css" asp-append-version="
10 </head>
11 <body>
12 <header>
13     <partial name="_Nav_Bar"/>
14 </header>
15 <div class="container">
16     <main role="main" class="pb-3">
17         @RenderBody()
18     </main>
19 </div>
20 <footer class="border-top footer text-muted">
21     <div class="container">
22         &copy; 2023 - WebApplication2023MVCCore - <a asp-area="" a
Ln 1, Col 1 Spaces CRLF
```

- Modifier le contenu de l'élément title. Changer le texte de ViewData(Title) dans le layout du "Nom de l'Application" à "MVC Movie" et le contrôleur de Home à Movies. Les parties suivantes de mise en page sont illustrées ci-dessous :

@ViewData["Title"] - MVC Movie

```
.....
<div class="container-fluid">
    <a class="navbar-brand" asp-area="" asp-controller="Home" asp-action=« Index »>MVC Movie</a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target=".navbar-
collapse" aria-controls="navbarSupportedContent"
        aria-expanded="false" aria-label="Toggle navigation">
        <span class=« navbar-toggler-icon »></span>.....
```

- Exécutez l'application et Notez le nouveau titre au niveau de la barre de navigation.

Lorsque nous avons créé tout d'abord le fichier *Views\Movie\Index.cshtml* , il contenait le code suivant :

```
@{
    Layout = "~/Views/Shared/_Layout.cshtml";
}
```

Le code razor ci-dessus définit le layout. Examinez le fichier *Views_ViewStart.cshtml* , il contient la balise exacte de Razor. Le fichier *Views_ViewStart.cshtml* définit le layout commun qui sera utilisé par toutes les vues, donc vous pouvez commenter ou supprimer ce code dans le fichier *Views\Movie\Index.cshtml*.

Vous pouvez utiliser la propriété de Layout pour définir un layout différent ou aucun layout si elle est nulle.

Maintenant, nous allons changer le titre de la vue Index.

- Ouvrez *AspMovieApp\Views\Movie\Index.cshtml*. Il y a deux endroits pour se faire un changement : tout d'abord, le texte qui apparaît dans le titre du navigateur, puis dans l'en-tête secondaire (l'élément <h2>) Pour modifier le titre de la page qui est définit au niveau du layout on utilisera la propriété Title de l'objet ViewBag. En utilisant l'approche ViewBag, vous pouvez passer facilement des paramètres entre votre vue et layout.

```
@*@{
    Layout = "~/Views/Shared/_Layout.cshtml";
}*@

@{
    ViewBag.Title = "Index";
}

<h2>Index</h2>

<p>Bonjour ceci est une vue!</p>
```

A présent, nous voulons ressortir la barre de navigation dans une vue partielle.

Une vue partielle est une vue non-complète (On ne trouve pas toutes les balises : `html, head, body...`) utilisée pour factoriser une partie du code répétée dans les différentes vues de l'application.

- Créer une vue partielle « _Nav_Bar.cshtml » et l'appeler au niveau du _Layout.
- Ajouter au niveau du _Layout l'appel de la vue partielle
`@Html.Partial("_Nav_Bar")`

```
<header>  
  
    <partial name="_Nav_Bar"/>  
  
</header>
```

- Changer le fichier Bootstrap