TP3 – Projet de session

(15%)

Cette deuxième partie du **TP3 (Projet)** doit être remise avant le **mardi 5 mai à 23h59**.

Notez que pour que le TP3 fonctionne, vous devez avoir complété le TP2 !

Étape 3.2 - Recherche (LinQ) et routage par attribut (R15 à R18)

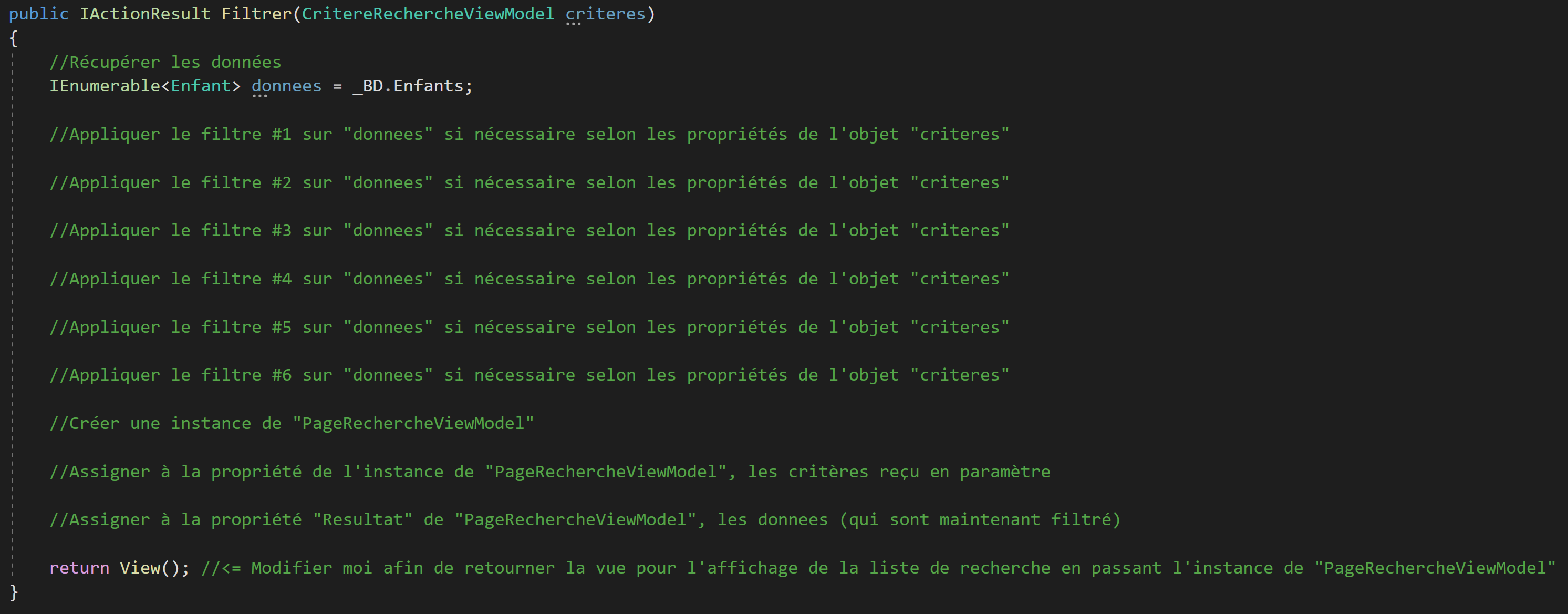
1. **RENDRE LA RECHERCHE FONCTIONNELLE (2e partie 😉)**

**RÉALISER LA RECHERCHE D’ENFANTS**

Bravo! À ce stade-ci l’affichage de la vue fonctionne bien! Cependant, il reste encore du travail pour rendre le bouton « Rechercher » fonctionnel! Nous allons ajouter une nouvelle action dans le contrôleur EnfantController afin de pouvoir lancer une recherche basée les propriétés d’un objet « CritereRechercheViewModel » qui sera généré à partir des informations transmises par le formulaire de recherche. Nommez cette action « Filtrer » au niveau du contrôleur. Cette action « /Enfant/Filtrer » devra être appelée à la suite d’un clic dans le formulaire de recherche sur le bouton « Rechercher ». L’action devra être appelée à l’aide d’un GET. Modifier les 2 propriétés de l’élément <form> en conséquence!

**SUGGESTION NUMÉRO #1**

Voici la structure de départ suggéré pour l’action avec quelques commentaires.



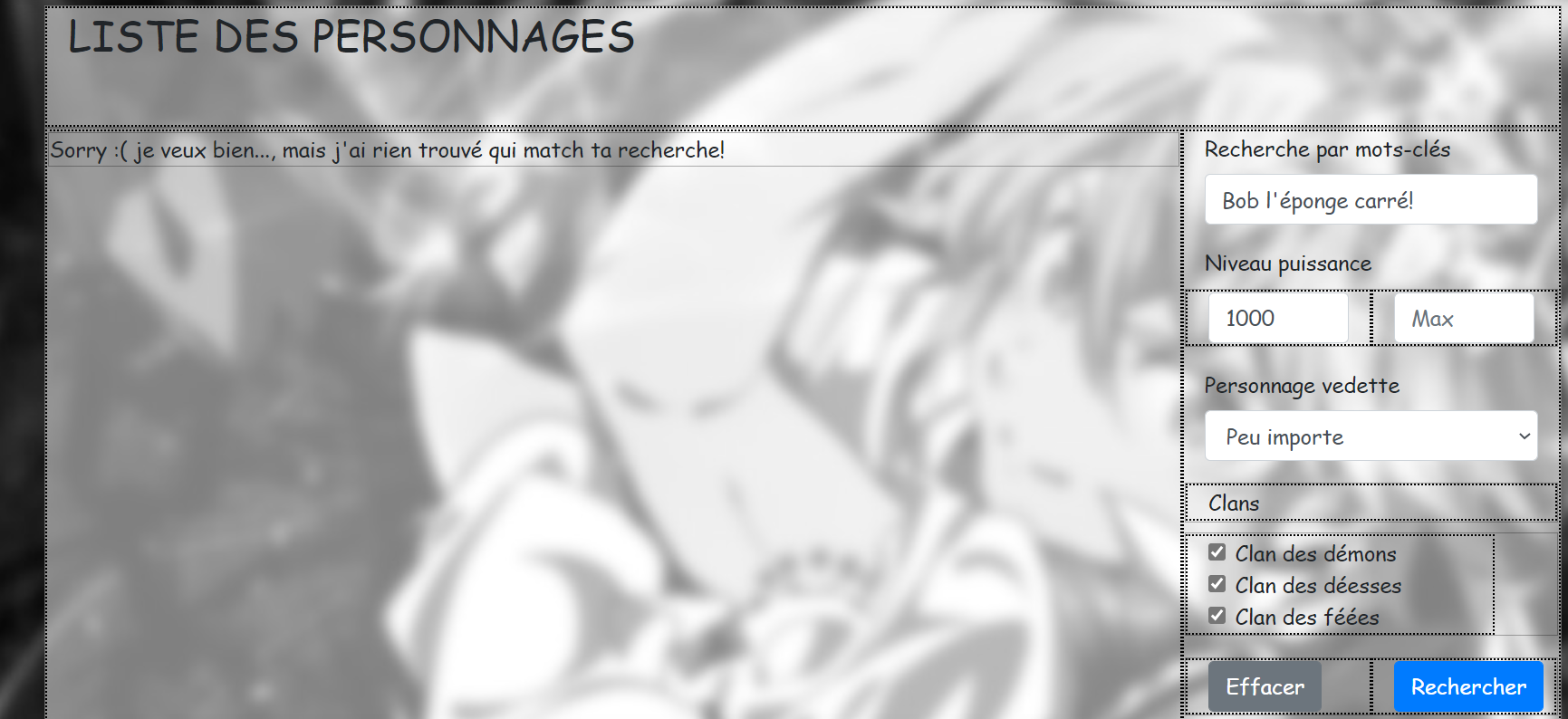
N’hésitez pas à utilisez LINQ et des IF!

**SUGGESTION NUMÉRO #2**

NE TENTEZ PAS DE LES RÉALISER TOUT D’UN COUP! Et n’attendez surtout pas à la fin pour tester! Testez au fur et à mesure le bon fonctionnement des critères. N’hésitez pas à utiliser des points d’arrêt et déboguer pas-à-pas si nécessaire pour observer le contenu de vos variables!

**RECHERCHE : AFFICHER UN MESSAGE SI AUCUN ENFANT TROUVÉ**

Si la séquence d’enfants dans la vue « Recherche.cshtml » ne contient pas d’éléments (c’est une liste vide), la vue doit afficher un message sympathique à l’utilisateur de votre choix. Les champs de recherche doivent rester visibles.

**Exemple :** 

Étape 4 – Gestion des entités enfants (R21 à R22)

**ÉTAPE 4 DE 5 – GESTION DES ENTITÉS ENFANTS**

**OBJECTIF**

Cet exercice est une suite directe de l’exercice de la semaine #9. En résumé, vous allez faire en sorte d’être en mesure de supprimer un enfant.

# CRÉER UN CONTRÔLEUR POUR LA GESTION DES ENFANTS

Nous allons créer un nouveau contrôleur nommé « GestionEnfantController ». Ce contrôleur va servir à la gestion des enfants. Il va servir à supprimer des entités enfants! Utilisez l’assistant pour créer le contrôleur (prenez le modèle « Contrôleur MVC avec des actions de lectures/écritures »).



L’assistant à créer le contrôleur pour vous! Il l’a créé avec plusieurs actions. Supprimer l’action « Index », l’action « Create », l’action « Details » et les actions « Edit », conservez l’action « Delete ». Nous la compléter éventuellement! Apporter les modifications nécessaires afin que le contrôleur puisse avoir accès à la base de données (comme vous avez déjà fait avec les autres contrôleurs en récupérant le singleton).

**REMARQUES IMPORTANTES POUR LES TAGS HELPERS**

Certaines vues générées par l’assistance utilisent des tags helpers. Vous devez vous assurez d’avoir dans votre fichier « \_ViewImports.cshtml » qui se trouve à la racine du dossier « View » la directive permettant l’utilisation des tags helpers dans les vues.

# SUPPRIMER UN ENFANT

## GÉNÉRER UNE VUE POUR SUPPRIMER UN ENFANT

À l’aide de l’assistant de génération de vue, ajoutez une vue pour supprimer un enfant.  
Cette vue devra correspondre à l’action « delete » de votre contrôleur « GestionEnfantController ».  
  
La vue générée devrait être présente dans le dossier « View/GestionEnfant/Delete.cshtml »  
Vous devriez y avoir accès en tapant l’URL : /GestionEnfant/Delete/{ID}

## MODIFIER LA VUE GÉNÉRÉE PAR L’ASSISTANT

Modifier la vue générée par l’assistant afin qu’elle   
contienne au moins les 3 éléments suivant :

* Le nom de l’enfant;
* L’image de l’enfant (elle doit être affichée);
* L’ID représentant l’enfant;

Assurez-vous qu’aucun texte en anglais ne soit visible. Par exemple, modifiez le bouton « Delete » afin qu’il affiche le texte « Supprimer » et renommez le message « Are you sure you want to delete this? » par autre chose. Retirez également le lien « Back to List ».

## AJOUT D’UN LIEN

Ajouter une icône dans l’affichage des résultats afin que l’utilisateur puisse accéder à la vue pour supprimer le personnage.

## ADAPTER L’ACTION « DELETE » DU CONTRÔLEUR « GESTIONENFANTCONTROLLER »

**La méthode « HttpGet »**

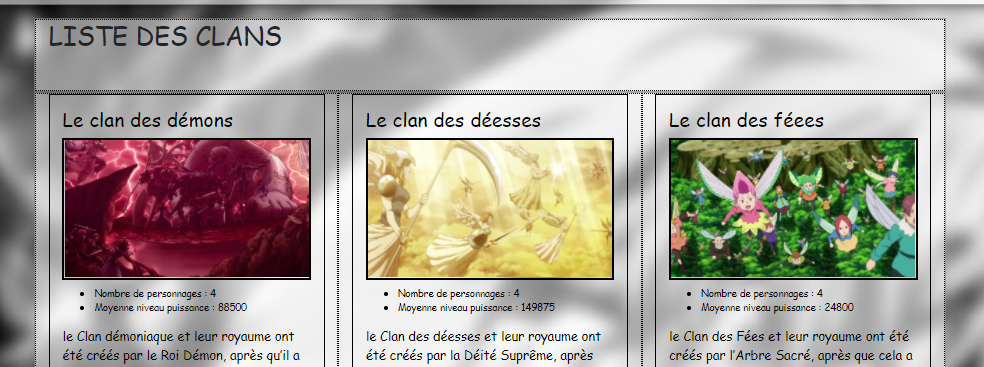
Adapter la méthode « GET » de l’action « Delete » du contrôleur « GestionEnfantController » afin de récupérer l’entité enfant associée à l’identifiant reçu en paramètre et passer cette enfant à la vue. Si aucun enfant ne possède cet identifiant affiché à l’utilisateur la vue « NotFound » **P.s.** Pensez à mettre cette vue dans le dossier « Shared » afin qu’à la fois le contrôleur GestionEnfantController et EnfantController puisse l’utilisez.

**La méthode « HttpPost »**

Adapter la méthode « POST » de l’action « Delete » du contrôleur « GestionEnfantController » afin que celle-ci puisse faire deux choses. Premièrement, retirer l’enfant de la liste des enfants de son parent. Deuxièmement, retirer l’enfant de la liste des enfants de la base de données. Vous devez prendre en considération les points suivants dans votre code :

* + Faites une requête LINQ sur les enfants de la base de données afin de récupérer l’enfant possédant le même « Id » que celui de l’enfant reçu en paramètre.
  + Sur l’enfant obtenu à l’aide de la requête LINQ plus tôt, fait en sorte d’accéder à son parent à l’aide de la propriété correspondant, puis accéder à la liste des enfants de ce parent. Sur cette liste, appeler la méthode « remove » et passer l’enfant obtenu à l’aide de la requête LINQ. Cela aura pour effet de supprimer l’enfant de la liste des enfants du parent. (voici un exemple) 
  + Sur la liste des enfants de la base de données, faites un « remove » de l’enfant obtenu à l’aide de la requête LINQ plus tôt.
  + Mettre l’instruction suivante afin de rediriger l’utilisateur vers l’action « Index » du contrôleur « Home » (l’instruction aura expliqué le prochain cours) :   
    

# VALIDER LA BONNE MISE À JOUR DES STATISTIQUES



L’action de supprimer termine le réaffichage à l’utilisateur de la vue correspondant à l’action « Index » du contrôleur « Home ». Vérifier que les statistiques sont toujours à jour à la suite de ces deux actions. Si vous avez bien fait les étapes jusqu’à présent, il ne devrait pas y avoir d’erreurs et ça devrait se mettre à jour automatiquement. Dans le cas contraire, voici des pistes de recherche afin d’identifier votre erreur :

* Valider dans la vue « Home/Index.cshtml » avoir bien utilisé LINQ afin de générer les statistiques sur les différentes entités enfant;
* Valider d’avoir bien fait les étapes en lien avec l’ajout d’un enfant;
* Valider d’avoir bien fait les étapes en lien avec la suppression d’un enfant.

Étape 5 - Gestion des favoris (R23 à R24)

**ÉTAPE #5 de 5 – GÉRER LA LISTE DES FAVORIS ET VUE PARTIELLE**

**OBJECTIF**

Cette étape est la dernière du projet de session.   
Vous allez devoir manipuler les variables de session afin de gérer votre liste de favoris.   
Vous allez aussi simplifier le code de vos vues en utilisant des vues partielles.



# PRÉALABLE – CONFIGURATION POUR LA SESSION

Vous devez préparer votre projet afin de pouvoir utiliser le mécanisme de session :

* Ajoutez à votre projet le package NuGet pour utiliser le mécanisme de session.
* Ajoutez dans le fichier « Startup.cs » le code nécessaire à la configuration de la session.
* Ajoutez la classe « SessionExtension » permettant la sérialisation d’objets complexes.

Normalement, votre projet devrait compiler sans erreur.

# STRATÉGIE POUR RÉALISER LA GESTION DES FAVORIS

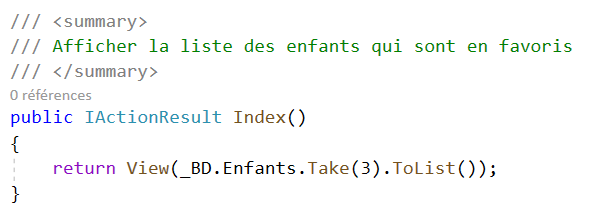
Le principe qui sera utilisé pour la liste des favoris sera de conserver en session une ***List<int>*** représentant les IDs des enfants qu’on souhaite conserver dans nos favoris. Le contrôleur FavorisController aura au final 3 actions :

* **Index()** : afficher les enfants favoris (on va te donner le code)
* **AjouterUnEnfant(int id)** : ajouter un ID (int) à la liste stockée en session
* **SupprimerUnEnfant(int id**) : supprimer un ID (int) à la liste stockée en session

On va commencer par faire fonctionner la vue rattachée à l’action « Index ».   
Par la suite, vous allez devoir réaliser les actions « AjouterUnEnfant » et « SupprimerUnEnfant ».

# MODIFIER LA VUE « INDEX.CSHTML » DE FAVORIS

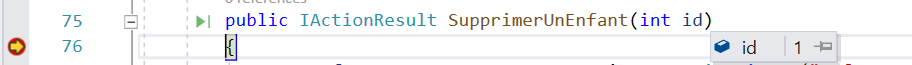
Actuellement les données de la vue « Index.cshtml » sont hardcodées, vous devez modifier la vue afin qu’elle reçoive plutôt une ***List<TP2.Models.Enfant>*** comme modèle. Afin de tester le bon fonctionnement de votre vue, mettez temporairement un code ressemblant à ceci pour l’action « Index » afin d'être en mesure d'afficher les 3 premiers enfants de la base de données. Ça devrait déjà être le cas, mais bon… n’oubliez pas que votre contrôleur doit recevoir la base de données par singleton à travers son constructeur.



Concernant votre vue :

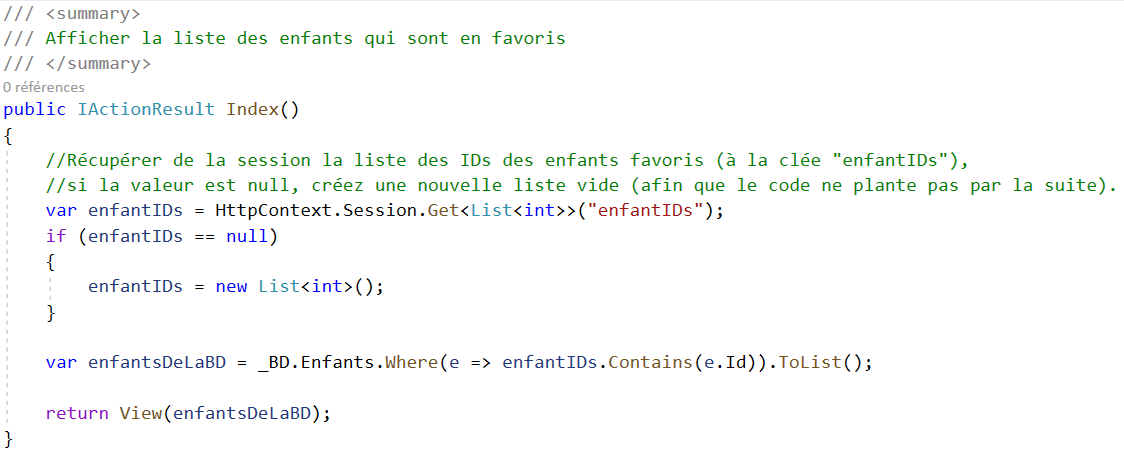
* Chaque élément enfant de la liste doit apparaître dans un <tr>
* Le bouton « Supprimer » sur chaque <tr> doit soumettre un formulaire dont :
  + L’action est « SupprimerUnEnfant » (faites-le avec un tag helper)
  + Un champ hidden avec le name « id » et comme value l’id de l’enfant
* Il est important que le nombre d’éléments de la liste soit affiché   
  

Tester que les boutons « Supprimer » envoient bien les données au serveur (le ID de l’enfant) en créant au niveau du contrôleur une nouvelle action nommée « SupprimerUnEnfant » vide. Placez un point d’arrêt et assurez-vous que lorsque vous cliquez sur un bouton supprimer l’exécution s’arrête et que la valeur pour le paramètre « id » correspond bien à l’ID de l’enfant que vous avez cliqué!



# CODER L’ACTION « INDEX » DU CONTRÔLEUR

On avait mis du code temporaire pour l’action « Index ». Voici maintenant à quoi ressemble le vrai code à utiliser pour l’action. Vous devrez retaper ce-code SANS réfléchir!!!

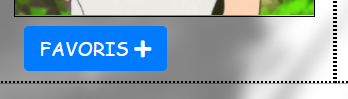


NONNNNN! C’est une blague! Il faut réfléchir, c’est important de comprendre ce que ce code fait et de ne pas juste le recopier comme un… un… recopieur niveau 70. Voici quelques explications : on tente de récupérer la liste ***List<int>*** représentant les IDs des enfants qu’on souhaite conserver dans nos favoris. Ensuite, on tente de faire une recherche dans la BD pour obtenir une ***List<Enfant>*** dont les enfants possèdent l’un des ID de la liste. On fournit ensuite cette liste à la vue. Il se peut que la ***List<int>*** soit null (quand on démarre l’application, on n’a pas nécessairement d’éléments dans notre liste de favoris encore), c’est pour cela qu’on dit que si la liste dans la session est vide, créer une nouvelle liste, pour éviter de faire planter la requête LINQ qui va suivre.

# MODIFIER LA VUE « DETAIL.CSHTML » DE ENFANT

Dans la vue servant à l’affichage du détail d’un enfant, vous devriez avoir un bouton permettant de soumettre un formulaire afin d’ajouter l’enfant à votre liste de favoris. Concernant ce formulaire, assurez-vous que :

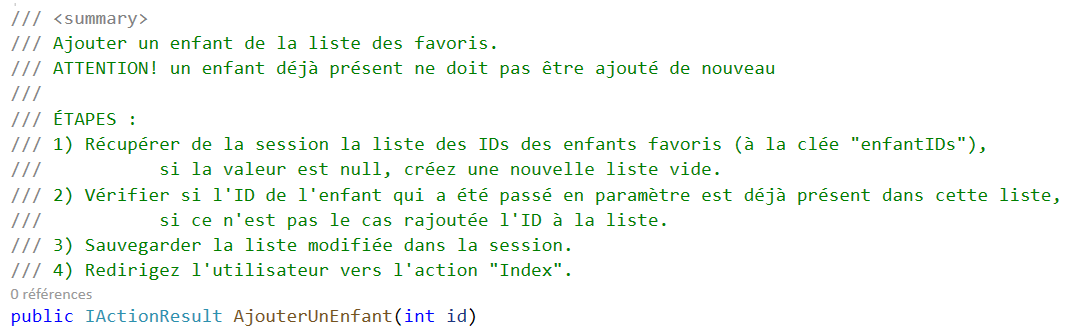
* + L’action est « AjouterUnEnfant » du contrôleur « Favoris » (tag helper)
  + Un champ hidden avec le name « id » et comme value l’id de l’enfant



Tester que le bouton envoie bien les données au serveur en créant au niveau du contrôleur Favoris une nouvelle action nommée « AjouterUnEnfant » vide. Placez un point d’arrêt et assurez-vous que lorsque vous cliquez sur le bouton l’exécution s’arrête et que la valeur pour le paramètre « id » correspond bien à l’ID de l’enfant que vous avez cliqué!

# CODER L’ACTION « AJOUTERUNENFANT »

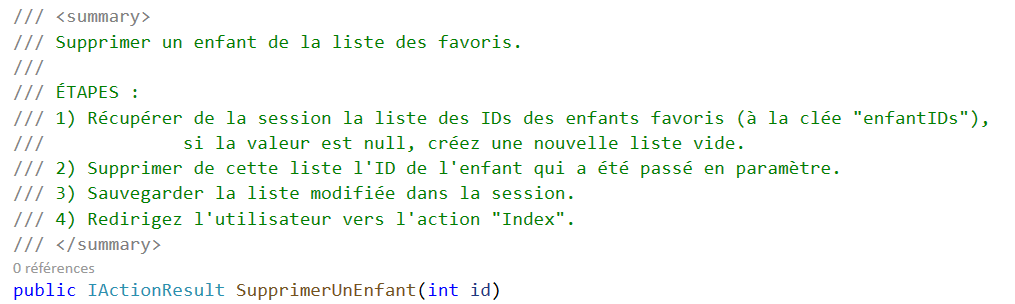
Vous devez maintenant coder l’action! On ne vous donne pas le code (car dans la vie le travail d’un programmeur c’est de trouver le code :P). On te donne malgré tout quelques étapes à suivre qui vont te permettre de réaliser le code.



N’hésitez pas à utiliser les points d’arrêt, explorer les variables et réaliser des tests. Normalement, après avoir codé ceci, l’ajout d’un enfant dans la liste des favoris devrait fonctionner.

# CODER L’ACTION « SUPPRIMERUNENFANT »

Réaliser maintenant le code pour la dernière action.   
Comme précédemment, voici les grandes étapes.

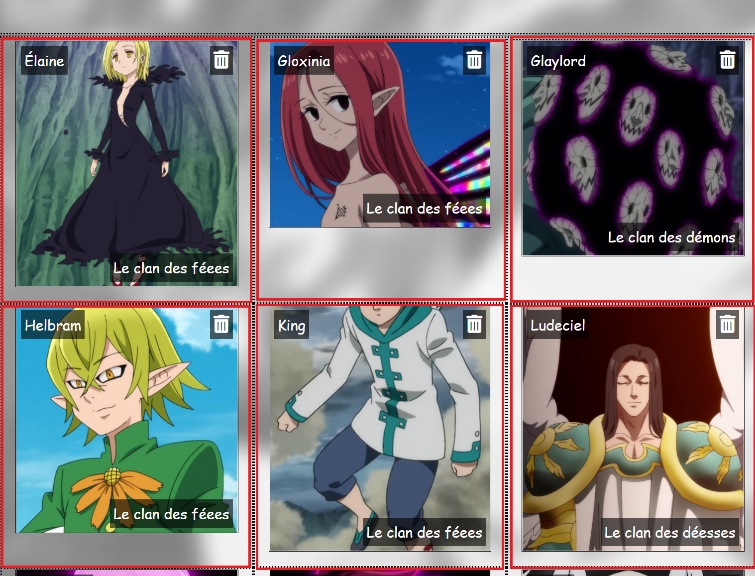


N’hésitez pas à utiliser les points d’arrêt, explorer les variables et réaliser des tests. Normalement, après avoir codé ceci, l’ajout d’un enfant dans la liste des favoris devrait fonctionner.

BRAVO! À ce stade-ci ta liste des favoris devrait fonctionner normalement!

# SEGMENTER LE VISUEL À L’AIDE DE VUE PARTIELLE

Dans la page d’affichage des résultats de recherche, les informations de chaque enfant sont affichées une colonne par enfant. Le code pour le visuel peut-être rapidement lourd et gros (on va dire)! On souhaiterait que tu rendes ton code plus simple à lire. Commence par créer une vue partielle permettant l’affichage des informations d’un enfant. La vue va devoir recevoir comme modèle un ***Enfant***.



Par la suite, dans la boucle servant à l’affichage des résultats de recherche, appeler cette vue partielle avec les différents éléments de la boucle. (P.s. Au final, rien n’a changé, mais tu as créé un petit morceau de visuel réutilisable!)

**BRAVO! Tu as fini ^^**

 Spécifications (non exhaustives) en lien avec les éléments de contenu évalués

|  |  |
| --- | --- |
| **ÉTAPE #3 - RECHERCHE D’ENTITÉS ENFANTS ET ROUTAGE PAR ATTRIBUT** | **Éléments de contenu évalués** |
| * Modification de l’action Enfant/Recherche pour filtrer la liste d’entité enfant et passer le ViewModel. | Contrôleurs |
| * Modification de la vue Enfant/Recherche pour transmettre les critères de recherche au contrôleur. | Vues |
| * Modification de l’affichage de la vue Enfant/Recherche pour les recherches infructueuses. | Vues |
| **ÉTAPE #4 - GESTION DES ENTITÉS ENFANTS (SUPPRESSION)** |  |
| * Création du contrôleur GestionEnfantController à l’aide d’un assistant. | Contrôleurs |
| * Création et modification de la vue GestionEnfant/Delete. (Ainsi que l’ajout d’un lien vers cette nouvelle page) | Vues |
| * Modification des deux actions GestionEnfant/Delete (Get et Post) pour la suppression d’une entité enfant. | Contrôleurs |
| * Validation du bon fonctionnement des statistiques suite à l'ajout et la suppression d'une entité enfant. | Vues |
| **ÉTAPE #5 - GESTION DES FAVORIS ET VUE PARTIELLE** |  |
| * Préparer la configuration nécessaire au bon fonctionnement des variables de session. | Modèles |
| * Modifications initiales de la vue et de l’action Favoris/Index. | Vues et Contrôleurs |
| * Modification du formulaire de la vue Enfant/Detail. | Vues |
| * Création d’une action Favoris/AjouterUnEnfant pour gérer l’ajout aux favoris. | Session et Contrôleurs |
| * Création d’une action Favoris/SupprimerUnEnfant pour gérer le retrait des favoris. | Session et Contrôleurs |
| * Création d’une vue partielle pour l’affichage d’une entité enfant dans la vue Enfant/Recherche. | Vues |

Grille d’évaluation du TP3 (65 points). Le TP3 complet comptera pour 15% de la note finale du cours.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | Niveau de performance | | | | |
|  |  | Insuffisant | Faible | Suffisant | Satisfaisant | Supérieur |
| Éléments de contenu évalués | **Vues** (25pts) :  -Typer correctement les vues (choisir le bon modèle)  -Créer une vue partielle et la typer correctement  -Utiliser une vue partielle à l'intérieur d'une vue  -Afficher les informations d'un ViewModel correctement à l'utilisateur  -Afficher les informations d'un Model correctement à l'utilisateur  -Utiliser des expressions implicites Razor  -Utiliser des expressions explicites Razor  -Concevoir des requêtes LINQ simples pour afficher des informations sur le modèle  -Créer une vue à l'aide de l'assistant  -Modifier des vues existantes ou créer à l'aide de l'assistant  -Utiliser les directives Razor de manière appropriées lorsque nécessaire  -Utiliser des "tags helpers" | 0 à 11 pts  L'utilisation de vues est **absente** ou plutôt **superficielle** et **ne démontre pas la maitrise** par l'étudiant de ces possibilités.   **Peu d'éléments** en lien avec les spécifications attendues ont été **réalisés** correctement. | 12 pts L'utilisation de vues **démontre peu la maitrise** par l'étudiant de ces possibilités.   **La plupart des éléments principaux**  en lien avec les spécifications attendues ont été **réalisés** correctement. | 16 pts  L'utilisation de vues **démontre une maitrise minimale** par l'étudiant de ces possibilités.   **Les principaux éléments** en lien avec les spécifications attendues ont été **réalisés** correctement. | 21 pts  L'utilisation de vues **démontre une grande maitrise** par l'étudiant de ces possibilités.   **Les principaux éléments** en lien avec les spécifications attendues ont été **réalisés** correctement. | 25 pts  L'utilisation de vues **démontre une parfaite** **maitrise** par l'étudiant de ces possibilités.    **L'ensemble** des éléments en lien avec les spécifications attendues ont été **réalisés** correctement. |
| **Contrôleurs** (28pts)**:**  -Configurer un "pattern" à l'aide du routage par attribut afin de donner l'accès à une action d'un contrôleur  -Récupérer les données variables dans le "pattern" du routage par attribut vers les paramètres de l'action d'un contrôleur  -Appliquer correctement des contraintes au niveau du routage par attribut lorsque nécessaire  -Déterminer et retourner la bonne vue selon le contexte d'exécution de l'action  -Préparer une instance d'un ViewModel et la transmettre à la vue choisie devant être retournée  -Concevoir des requêtes LINQ simples au niveau des actions  -Concevoir des requêtes LINQ composées au niveau des actions (avec 1 ou \* where selon le contexte)  -Créer un contrôleur à l'aide de l'assistant  -Ajuster le constructeur d'un contrôleur afin de récupérer l'injection de dépendance  -Récupérer les données transmises vers le serveur provenant de la vue  -Configurer, lorsque nécessaire, l'utilisation des bons verbes HTTP pour les actions du contrôleur (HttpGet/HttpPost)  -Réaliser une redirection à l'intérieur d'une action d'un contrôleur lorsque nécessaire | 0 à 13 pts  L'utilisation de contrôleurest **absente** ou plutôt **superficielle** et **ne démontre pas la maitrise** par l'étudiant de ces possibilités.   **Peu d'éléments** en lien avec les spécifications attendues ont été **réalisés** correctement. | 14 pts  L'utilisation de contrôleur **démontre peu la maitrise** par l'étudiant de ces possibilités.   **La plupart des éléments principaux**  en lien avec les spécifications attendues ont été **réalisés** correctement. | 17 pts  L'utilisation de contrôleur **démontre une maitrise minimale** par l'étudiant de ces possibilités.   **Les principaux éléments** en lien avec les spécifications attendues ont été **réalisés** correctement. | 22 pts  L'utilisation de contrôleur **démontre une grande maitrise** par l'étudiant de ces possibilités.   **Les principaux éléments** en lien avec les spécifications attendues ont été **réalisés** correctement. | 28 pts  L'utilisation de contrôleur **démontre une parfaite** **maitrise** par l'étudiant de ces possibilités.    **L'ensemble** des éléments en lien avec les spécifications attendues ont été **réalisés** correctement. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | Niveau de performance | | | | |
|  |  | Insuffisant | Faible | Suffisant | Satisfaisant | Supérieur |
|  | **Modèles** (4 pts) :  -Garder dans un état cohérent les relations entre les différents objets de la base de données  -Définir les règles de validations sur le modèle à l'aide d'annotations  -Vérifier l'état du modèle reçu par un contrôleur à l'aide du ModelState | 0 pt  L'utilisation de modèles est **absente** ou plutôt **superficielle** et **ne démontre pas la maitrise** par l'étudiant de ces possibilités.   **Peu d'éléments** en lien avec les spécifications attendues ont été **réalisés** correctement.. | 1 pts  L'utilisation de modèles **démontre peu la maitrise** par l'étudiant de ces possibilités.   **La plupart des éléments principaux**  en lien avec les spécifications attendues ont été **réalisés** correctement. | 2 pts  L'utilisation de modèles **démontre une maitrise minimale** par l'étudiant de ces possibilités.   **Les principaux éléments** en lien avec les spécifications attendues ont été **réalisés** correctement. | 3 pts  L'utilisation de modèles **démontre une grande maitrise** par l'étudiant de ces possibilités.   **Les principaux éléments** en lien avec les spécifications attendues ont été **réalisés** correctement. | 4 pts  L'utilisation de modèles **démontre une parfaite** **maitrise** par l'étudiant de ces possibilités.    **L'ensemble** des éléments en lien avec les spécifications attendues ont été **réalisés** correctement. |
| **Session** (8 pts) :  -Configurer l'utilisation de sessions au niveau applicatif  -Stocker dans la session un objet complexe  -Lire de la session un objet complexe  -Mettre à jour dans la session un objet complexe | 0 à 3 pts  L'utilisation de modèles est **absente** ou plutôt **superficielle** et **ne démontre pas la maitrise** par l'étudiant de ces possibilités.   **Peu d'éléments** en lien avec les spécifications attendues ont été **réalisés** correctement.. | 4 pts  L'utilisation de modèles **démontre peu la maitrise** par l'étudiant de ces possibilités.   **La plupart des éléments principaux**  en lien avec les spécifications attendues ont été **réalisés** correctement. | 5 pts  L'utilisation de modèles **démontre une maitrise minimale** par l'étudiant de ces possibilités.   **Les principaux éléments** en lien avec les spécifications attendues ont été **réalisés** correctement. | 6 pts  L'utilisation de modèles **démontre une grande maitrise** par l'étudiant de ces possibilités.   **Les principaux éléments** en lien avec les spécifications attendues ont été **réalisés** correctement. | 8 pts  L'utilisation de modèles **démontre une parfaite** **maitrise** par l'étudiant de ces possibilités.    **L'ensemble** des éléments en lien avec les spécifications attendues ont été **réalisés** correctement. |