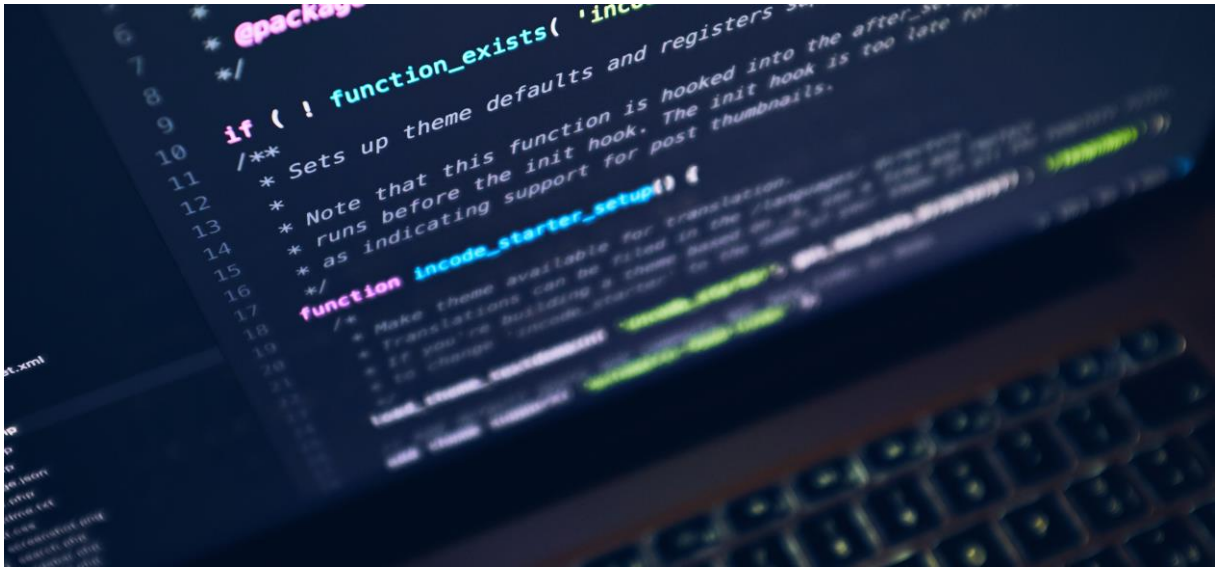

Rapport de synthèse

Hugo Boué

En résumé

Lors de ce projet de développement en .Net, j'ai contribué à la partie récupération des données venant du Front-end par l'intermédiaire d'un formulaire et à l'inscription de ces données dans notre base de données grâce aux dépôts. La fonctionnalité la plus complète que j'ai implémenté est l'écriture d'un nouveau Blu-ray car elle appelle plusieurs méthodes des dépôts et c'est la seule qui m'a posé un peu de problème.

Fonctionnalité

L'utilisateur, à travers l'onglet 'Ajout d'un Blu-ray' peut remplir un formulaire qui comprend tout un ensemble d'information permettant l'ajout du Blu-ray et notamment la partie la plus importante qu'est l'ajout des personnes qu'elles soient acteurs ou réalisateurs et qui est renvoyé par la View grâce à un ViewModel définit par Briac. Ces informations doivent être traité pour pouvoir réaliser les requêtes d'écriture en base de données. Je devais donc en plus d'écrire l'insertion du Blu-ray dans la table liée, ajouter les personnes dans les bonnes tables de jonction grâce à leur clé primaire.

Difficulté rencontrée

La plus grande difficulté rencontrée lors de l'implémentation de cette fonctionnalité a été lors de l'ajout des personnes. Effectivement, ne travaillant pas sur les données qui m'était envoyées par le formulaire car Briac s'en occupait, j'ai dû lui demander de m'envoyer les données dont j'avais besoin mais il ne pouvait pas toujours choisir le type car le Front ne lui permettait pas toujours de le faire. Etant donné qu'un film possède plusieurs acteurs, le formulaire devait me renvoyer un tableau d'identifiant correspondant aux acteurs mais ce n'était pas le cas.

Solution

Mes méthodes du dépôt de personne qui me permettent d'écrire dans les tables de jonction des acteurs, réalisateurs ou scénaristes prennent en paramètre la clé primaire du Blu-ray et la clé primaire des personnes.

Il m'a donc fallu pour commencer créer une méthode pour récupérer l'identifiant du dernier Blu-ray ajouté en base mais il n'y avait ici aucune difficulté. Ensuite, avec cet identifiant, je devais ajouter les acteurs.

Cependant, la liste d'identifiants des acteurs que je souhaitais recevoir devait avoir la forme suivante : [id1, id2, id3] mais grâce au debug (conseillé par vous car l'on ni pensait plus du tout), je me suis rendu compte que j'avais un objet de ce type : [[id1, id2, id3], []] qui est une matrice plutôt qu'une liste.

Une fois que je me suis rendu compte de ce problème (qui n'ajoutait l'acteur quand la liste était composée d'un seul acteur mais dès lors que l'on dépassait un acteur, le site renvoyait une erreur d'écriture en base de données), j'ai donc pris le premier élément de la matrice, qui correspond à mon tableau, que j'ai stocké dans un tableau de string grâce à la méthode split en utilisant comme séparateur la virgule.

Enfin, mon tableau étant bien formé, j'ai pu réaliser une boucle for each sur les éléments de ma liste.

```
int id = brRepository.GetLastBluRay().Id;
foreach (var acteurs:string in formModel.ActeursToAdd)
{
    string[] acteurId = acteurs.Split(separator:",");
    foreach (var acteur:string in acteurId)
    {
        pRepository.AddActeursByFilm(id, acteur);
    }
}
```

Voici la solution finale pour le traitement des id des acteurs à ajouter.

Le mieux aurait été de pouvoir écrire une fonction qui prenait en paramètre ce tableau pour écrire directement toutes les lignes dans la table, mais je n'ai pas réussi à le faire (peut-être que cela n'est pas possible non plus). En y repensant, le problème rencontré est d'une difficulté enfantine (mais qui nous a quand même pris du temps à résoudre) mais sans l'utilisation du debug, je pense que je n'aurais jamais réussi à ajouter les personnes dans les bonnes tables.