**Laboratoire R19**

Images



**Résumé des fonctionnalités 🍎🍊🍋🍐🍇**

* Il y a des **étudiants** et des **fruits**. On note les trois fruits préférés de chaque étudiants grâce à la table **EtudiantFruit**.
* Essentiellement, nous allons ajouter la **gestion des images** pour les fruits. Un formulaire nous permettra de vérifier les trois fruits préférés d’un étudiant au choix et nous afficherons les images des fruits en question.

**Migrations de départ**

* Il y a déjà quelques migrations (à ne pas modifier) qui créent les trois tables et les meublent. Ne les exécutez pas tout de suite !

**Une image contenant texte

Description générée automatiquement**

**Étape 0 - Un foyer pour accueillir les images 📷🏠💾**

* **9**  Par défaut, **FILESTREAM** n’est pas activé avec Sql Server. MAIS au cégep, c’est déjà activé.

Si vous travaillez à la maison, suivez la procédure pour l’activer.)

* **10**  Complétez le fichier **CreateDatabase.sql** (Situé dans **Sql\_Scripts**) pour configurer un nouveau FILEGROUP, puis exécutez ce script dans SSMS ou Visual Studio pour créer la BD.
  + **Rappel** : Evolve ne peut ni créer la BD, ni configurer FILESTREAM. C’est pour ça qu’on exécute ce script à la main.
  + Faites la commande pour upgrader l’installation d’Evolve au cégep. (Dans Script Preparation Labo 19.txt)
    - 3) Mettre à jour pour la version prerelease qui fonctionne avec .Net 8.0

**dotnet tool update evolve.tool -g –prerelease**

* + Utilisez C:\EspaceLabo\FG\_Images comme dans les notes de cours. Nommez votre FILEGROUP **FG\_Images**. (À la maison, créez un dossier nommé EspaceLabo dans votre disque C:)

Une image contenant texte, Police, blanc, capture d’écran

Description générée automatiquement

**Étape 1 - Migration imagée** 🕊📷

* **11**  Créez une nouvelle migration **V1\_4\_\_imageFruit.sql**
  + Le but de cette migration est d’ajouter deux colonnes et deux contraintes à la table **Fruit**. Cela se fera en 3 étapes :
    - Ajouterlacolonne **Identifiant** detype **uniqueidentifier NOT NULL** et, parce qu’il y a déjà des données dans la table, on va lui donner tout de suite une contrainte **DEFAULT ,** pourluidonnerlavaleur **newid() :**
    - **Une image contenant texte, capture d’écran, Police, ligne

      Description générée automatiquement**Ajoutezsacontrainte **UNIQUE¸**
    - Ajouter la colonne Photo de type varbinary(max) **FILESTREAM** NULL
  + Exécutez maintenant toutes les migrations (1.0 à 1.4). La commande est dans le fichier **Script Preparation Labo19.txt**
    - Si vous êtes à la maison et que vous ne l’avez pas déjà installé, Installez Evolve :

**dotnet tool install --global Evolve.Tool**

* + - Attention les migrations 1.2 et 1.3 prennent environ une minute à s’exécuter car il y a 700 000 insertions)
* **13**  *Scaffoldez* les Models suite à l’exécution (réussie) des migrations :

**dotnet ef dbcontext scaffold Name=R19\_Labo Microsoft.EntityFrameworkCore.SqlServer -o Models –context --context-dir Data --data-annotations --force**

Observez que vous avez maintenant dans le modèle Fruit.cs les deux champs que l’on vient d’ajouter.

**Étape 2 - Mettre un visage sur un fruit 🍎🧑**

* **14-15**  Complétez l’action **AjouterImage** dans **FruitsController** (et sa vue Razor) pour qu’elle permette d’associer une image à un fruit. Cela impliquera d’utiliser le ViewModel **ImageUploadVM**. (Qui est déjà fait et que vous n’avez pas besoin de modifier pour ajouter Image car on va directement remplir photo de Fruit.)
* Afin de tester l’action AjouterImage, *uploadez* trois images de fruit (situées dans le dossier **images\_test**) dans l’application Web :
  + uploadez les images de fruits conservateurs comme la pomme, l’orange et la banane
  + ou uploadez les images de fruits exotiques comme la bergamote, pain de singe et jacque.

**Étape 3 - Trois images valent trois mille mots ? 📷💬**

* **17-21**  Complétez l’action **PreferenceEtudiant** dans **FruitsController** (et sa vue Razor) pour afficher les trois images de fruit. Les préférences sont déjà affichées, mais sans les images. Cela impliquera d’utiliser **FruitsPreferesVM.cs**.
* Lorsqu’il sera le temps de tester, recherchez l’étudiant suivant dans la page
  + Fruits conservateurs:

Une image contenant texte

Description générée automatiquement

Une image contenant texte, fruit, intérieur, orange

Description générée automatiquement

* + Fruits exotiques :



Une image contenant diagramme

Description générée automatiquement

**Étape 4 - De rapide à un peu plus rapide ⚡**

* **26**  Vous avez peut-être remarqué que le temps de traitement pour récupérer les fruits préférés d’un étudiant est affiché lorsqu’on fait une recherche.
  + Nous allons essayer d’améliorer légèrement ce temps en ajoutant deux index.
  + Avant de créer le moindre index, notez le temps obtenu : (Perso j’ai 1937 millisecondes)
    - **Premier index** : Comme on utilise le prénom et le nom d’un étudiant comme critères de recherche, le premier index à créer est sur la table **ETUDIANT**, pour les colonnes **Prenom** et **Nom**. (On peut créer un **NONCLUSTERED INDEX** et spécifier plusieurs colonnes dans les parenthèses à la fin !)
    - **Deuxième index** : Comme la jointure entre Fruit et EtudiantFruit se sert de la colonne FruitID et que la jointure entre Etudiant et EtudiantFruit se sert de la colonne EtudiantID, le deuxième index à créer est sur la table EtudiantFruit, pour les colonnes **EtudiantID** et **FruitID**. (C’est encore un **NONCLUSTERED INDEX**)
  + Après avoir créé les deux index, testez à nouveau, plusieurs fois, successivement, la même recherche que tout à l’heure. (Il est normal que la première fois ce soit plus long)
    - (Là j’ai eu 141 milisecondes)
    - Il n’y a pas de quoi se lancer dans les murs, mais à plus grande échelle la différence aurait été encore plus intéressante.