**Laboratoire #10** (Partie 2 sur 2)

Authentification



**Partie 1** : Bleu et orange : État initial de la BD.

Orange : Devra être supprimé ou modifié.

**Partie 2** : Fushia : Devra être ajouté.

**Résumé des fonctionnalités :**

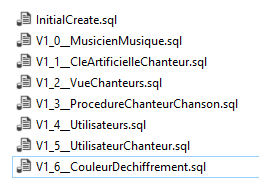
* Il y a des **chanteurs** et des **chansons**. (Les chansons appartiennent à un chanteur) On peut les afficher sur le site. (C’était la partie 1)
* Il y aura des utilisateurs, on pourra s’**inscrire**, se **connecter** et se **déconnecter**. On pourra voir nos propres informations dans notre **profil**.
* Les utilisateurs pourront **mettre des chanteurs dans leur liste de favoris**.
* C’est un peu comme le TP2 de Prog Web services, mais pris par surprise un dimanche matin en peignoir et en pantoufles.

**Attention !**

* On a fait les migrations jusqu’à V1\_3\_ProcedureChanteurChanson.sql dans la partie 1.
* Il faut maintenant faire la gestion des utilisateurs avec les migrations V1\_4\_\_Utilisateurs.sql à V1\_6\_\_CouleurDechiffrement.sql

Pour être certain que vous êtes corrects, nous vous donnons le code Labo 10 Partie 1 SOLUTION.

Le code du projet est modifié de sorte que vous pouvez faire les migrations jusqu’à V1\_3\_\_ProcedureChanteurChanson.sq.

****

**Étape 0 – Mise en place du projet 🏗**

* Avec SSMS ou Visual Studio, exécutez manuellement le script dans le fichier **InitialCreate.sql** pour créer la BD du labo 10. Si vous nommez bien la BD, le string de connexion déjà configuré devrait marcher.
  + Si vous faites des erreurs dans les migrations n’hésitez pas à supprimer la BD et réexécuter toutes les migrations jusqu’à la version critique plus tard.
* Installez Evolve.

dotnet tool install --global Evolve.Tool

(Rappel : c’est à répéter à chaque fois que vous changez d’ordinateur au cégep)

* Appliquez la migration **1.3** à l’aide de la commande Evolve. Vous devez spécifier les schémas Musique ET Utilisateurs pour que la commande soit toujours pareille (à part la version) puisque vous allez l’exécuter plusieurs fois.

evolve migrate sqlserver -c "Server=.\SQLEXPRESS;Initial Catalog=XXXXXX;Integrated Security=True;Persist Security Info=False;Pooling=False;MultipleActiveResultSets=False;Encrypt=False;TrustServerCertificate=False" -s Courses -s Utilisateurs --target-version 1.3

* + À ce stade, le projet Web devrait fonctionner ! Pas besoin de faire un scaffold pour changer le DbContext et les Models.

**Étape 2 - Problèmes identitaires 👤😨**

Dans cette section, nous allons implémenter la gestion des utilisateurs et de leurs chanteurs favoris. Le travail est déjà commencé... mais il y a des maladresses impardonnables dans les migrations. C’est parti pour les migrations **1.4**, **1.5** et **1.6**.

* Pour la migration **1.4**, qui est déjà bien entamée, nous allons faire quelques modifications.
  + Il y a deux problèmes colossaux :
    - Les **mots de passe** sont **stockés en clair**. (Et quand quelqu’un s’authentifie, on compare si le mot de passe fourni est identique au mot de passe en clair)
    - La **couleur préférée** n’est **pas chiffrée** ! 🌈🙄 Bon ok... ce n’est pas grave, mais faisons semblant que la couleur préférée d’une personne est extrêmement secrète et permet de voler facilement son identité. 🕵️‍♂️ Il faut donc absolument **chiffrer** cette donnée !
  + **4-6, 9, 12-17**  Pour le hachage du mot de passe, il faudra modifier :
    - Le type de MotDePasseHache.
    - La procédure de création d’un utilisateur.
    - La procédure d’authentification d’un utilisateur.
  + **4, 7-8, 10-15**  Pour le chiffrement de la couleur préférée, il faudra :
    - Modifier le type de CouleurPrefere.
    - Créez les clés nécessaires pour chiffrer / déchiffrer.
    - Modifier la procédure d’inscription.
  + Une fois que c’est réglé, exécutez la migration **1.4** avec **Evolve**.
  + Faites un **scaffold** pour créer les nouveaux **Models** et remplacer le **DbContext**.
* dotnet ef dbcontext scaffold Name=Lab10 Microsoft.EntityFrameworkCore.SqlServer -o Models --context-dir Data --data-annotations --force
  + **20,22**  Décommentez les actions POST **Inscription** et **Connexion** dans **UtilisateursController**. Ils sont identiques à celles dans les notes de cours.
  + **17**  Décommentez les configurations requises dans **Program.cs**. Tout est déjà là.
  + **25**  Décommentez le bloc de code dans l’action **Index** de **MusiqueController**. Il sert à dire bonjour à l’utilisateur connecté. (S’il y en a un)
  + Testez les boutons Inscription, Connexion et Déconnexion. Tout devrait marcher.
  + **26**  Décommentez l’action Profil et ajoutez-lui un petit quelque chose pour qu’elle fonctionne juste avec les utilisateurs authentifiés.
    - Dans la vue Profil, il y aura des choses à décommenter et des █ à remplacer par des @.
  + Testez le bouton qui affiche le **profil** de l’utilisateur actuellement connecté.
    - C’est normal que la couleur préférée ne soit pas affichée.
* La migration **1.5** est déjà complétée. C’est une table de liaison Many-To-Many entre Utilisateur et Chanteur pour que les utilisateurs puissent mettre des chanteurs dans leurs favoris.
  + Exécutez la migration 1.5 avec **Evolve**.
  + Faites un **scaffold** pour créer les nouveaux **Models** et remplacer le **DbContext**.
  + Décommentez la propriété Utilisateur dans le UtilisateurEtFavorisViewModel.
  + Décommentez l’action **AjouterFavori** dans **MusiqueController**. N’oubliez pas le [Authorize].
  + Dans la vue **Chanteurs**, décommentez le **<th>** et le **<td>** qui permettront d’ajouter des chanteurs à nos favoris.
  + Dans l’action **Profil** de **UtilisateursController**, remplacez le **return View()** par la fin alternative qui permet d’envoyer dans le profil la liste des chanteurs favoris de l’utilisateur authentifié.
  + Dans la vue **Profil**, vous pouvez décommenter le tableau des artistes favoris ainsi que changer le @model dans le haut de la page pour utiliser un ViewModel qui contient l’utilisateur et sa liste d’artistes.
  + Testez (à nouveau) le bouton qui affiche le **profil** APRÈS avoir ajouté des artistes dans votre liste de favoris dans la vue **Chanteurs**.
    - Vous devriez vous la liste des chanteurs que vous avez ajoutés à vos favoris désormais.
* La migration **1.6** est presque terminée ! Elle permettra à un utilisateur de vérifier sa propre couleur préférée après avoir validé son mot de passe dans son profil.
  + **14**  Modifiez la migration, il reste trois choses à ajouter :
    - L’ouverture de la clé symétrique.
    - La conversion et le déchiffrement de la couleur préférée. (Cette colonne sera renommée en « Couleur », c’est normal)
    - La fermeture de la clé symétrique.
  + Exécutez la migration 1.6 avec **Evolve**.
  + Faites un **scaffold** pour créer les nouveaux **Models** et remplacer le **DbContext**.
  + Décommentez l’action **Couleur** dans **UtilisateursController**. Elle reçoit un mot de passe en paramètre, le compare avec celui dans la BD, et retourne la couleur déchiffrée dans la vue **Profil** à l’aide de la procédure stockée dans la migration 1.6.
  + Décommentez le bloc qui accompagne la couleur préférée dans la vue **Profil**. Elle permettra de valider son mot de passe pour accéder à notre couleur préférée quand on est connecté.
  + Testez la nouvelle option du profil qui permet de vérifier notre couleur préférée !
    - Si ça marche, vous avez terminé ~~de décommenter frénétiquement du code~~ ce labo.