**Semaine 09** (Partie 2 sur 2)



**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 :**

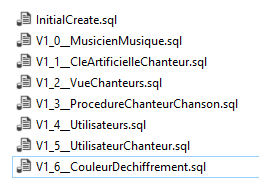
* Il y a des **chanteurs** et des **chansons**. (Les chansons appartiennent à un chanteur) On peut les afficher sur le site. (C’était la partie 1)
* Il y aura des utilisateurs, on pourra s’**inscrire**, se **connecter** et se **déconnecter**. On pourra voir nos propres informations dans notre **profil**.
* Les utilisateurs pourront **mettre des chanteurs dans leur liste de favoris**.

**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 S09\_Labo (avec la partie 1 complétée).

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

****

**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 S09\_Labo. 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.
* Appliquez les migrations jusqu’à 1.3. Vous ne devriez pas avoir besoin d’ajuster les modèles puisque vous les avez déjà arrangés auparavant.
  + 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**.

La gestion des utilisateurs est commencée dans la migration 1.4. MAIS Il y a deux problèmes colossaux :

Les **mots de passe** sont **stockés en clair!!!** et 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 !

* Pour la migration **1.4**, qui est déjà bien entamée, nous allons faire quelques modifications.
  + Pour le hachage du mot de passe, il faudra modifier :
    - Le type de MotDePasseHache.
    - La procédure d’inscription (création d’un utilisateur).
    - La procédure d’authentification d’un utilisateur.
  + 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**.

**Dans le controller UtilisateursController:**

* + Décommentez les actions POST **Inscription** et **Connexion** dans **UtilisateursController**. Ils sont identiques à celles dans les notes de cours.
  + Décommentez les configurations requises dans **Program.cs**. Tout est déjà là.
  + 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.**
  + 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 @. Et quelques autres petits changements.
  + 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. Aussi █ à remplacer par @.
  + 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 voir 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 !