Clément VALOT

Rapport devoir maison BDD

Le code C#

J'ai décidé de coder avec des mots clés, nom de dossiers, classes et méthodes en Anglais. L'affichage en WPF est également en Anglais. Cependant, mon modèle relationnel est lui fait en Français. Il est donc normal que dans ce rapport je fasse parfois référence à la classe RecipeCreator lorsque je veux parler de l'entité Cdr.

Je voudrais également ajouter que j'ai passé un temps considérable sur ce projet. Même si je n'étais pas à l'ESILV au semestre 5, j'ai décidé de me lancer sur un affichage WPF que j'ai dû apprendre de A à Z. De plus, mon binôme n'ayant pas du tout travaillé pour ce projet, j'ai dû me résoudre à me séparer de ce dernier à deux semaines du rendu du projet ce qui a grandement impacté ma charge de travail qui a considérablement augmenté. Finalement, après de longs rushs finaux pour boucler le projet, je suis heureux et satisfait de présenter un travail fini, beau et dont je suis fier car il symbolise parfaitement l'implication que j'ai fournie pour ce dernier.

Mon code est segmenté en 4 répertoires différents : Model, Database, Service et Page.

- **Model** : ce répertoire va servir à implémenter les classes de chacune des entités du MCD. Dans mon cas, j'ai donc 6 classes : Recipe.cs, Client.cs, RecipeCreator.cs, Product.cs, Provider.cs et Command.cs. Chaque classe est composée au moins des attributs correspondant à sa table et de certains attributs supplémentaires.
  Lorsque l'on a une clé étrangère, on appelle tout simplement le constructeur de la classe dont la clé étrangère est une clé primaire. Par exemple dans RecipeCreator on a : public int identifiant_cdr et public Client client car normalement identifiant_client est une clé étrangère de la table Cdr, mais dans ce projet, au lieu de simplement affecter la valeur de l'attribut de la clé étrangère, on établit une liaison entre cette table/classe et la classe de la clé étrangère.

- **Database** : ce répertoire sert exclusivement de passerelle entre le code C# et la base de données MySQL. On crée donc une classe pour établir la connexion avec la bonne base de données (Database.cs) et une classe par entité que l'on appelle Dao*NameEntity* (DaoClient.cs, DaoProduct.cs) (DAO=Data Access Object).
  On n'affiche rien avec le DAO ni ne fait de calculs dans ces fichiers, ils ne font que des corrélations entre les classes C# et les entités de la BDD MySql : soit on récupère les informations de la BDD (SELECT), soit on insère, update ou supprime des valeurs (INSERT INTO, UPDATE, DELETE).
  Toutes les méthodes créées ne sont pas utilisées dans le code. Globalement on retrouve les mêmes méthodes dans chaque classes : créer l'instance d'une classe model et l'insérer dans la DB, update les classes dans la DB, supprimer, Récupérer classes depuis la DB. Par exemple dans le DaoProduct, on trouve la méthode Delete (Product produit) même si on ne l'utilise pas.

- **Service** : Ce dossier est composé de fichiers qui vont réaliser du traitement (calculs, affichage console) sur les données récupérées par les fichiers du dossier Database ou sur les données à envoyer à la DB. Par exemple, quand un client réalise une commande, on doit créer la commande, calculer le cout total de celle-ci et calculer le nouveau solde cook de l'utilisateur après la validation ou encore mettre à jour les stocks de produits qui sont diminués après une commande.
  On trouve un fichier Service*NomTable* par classe de Model, un dossier Auth pour tout ce qui est l'identification de l'utilisateur (Sign Up, Log In, Forgot Password) et un dossier Validation qui s'occupe de vérifier si les inputs rentrés par l'utilisateur sont valides (mail correct, format verif mdp valide et forme correcte)

- **Page** : ce répertoire est composé des fichiers qui font le lien entre les informations rentrées par l'utilisateur et le Service. Ce sont ces fichiers qui réalisent l'interface Homme Machine (.xaml et .xaml.cs). Les .xaml sont les affichages WPF purs (textBox, ComboBox, TextBlock, largeur de fenêtre, bouton etc.) et les .xaml.cs réalisent le lien entre les .xaml et les données récupérées du service.
  On trouve retrouve 4 dossiers :
  - Auth : accessibles par tout les utilisateurs (forcément vu qu'il s'agit de l'affichage de la page de connexion et de la création de compte)
  - Admin : pages accessibles seulement par l'admin et le superAdmin
  - Fragment : pages de création recette et création commande et visuel des informations du compte (visibles par tous)
  - Demo visibles par l'admin et le superAdmin

Dans ce répertoire on trouve également le répertoire Session qui contient le fichier AuthUser qui sert à la définition du rôle de l'utilisateur (Client, Cdr et Admin (=SuperAdmin pour mode démo du professeur)) disponible à tout moment dans chaque page du WPF pour justement savoir quelle page rendre disponible à l'utilisateur.

Pour ce qui est du rendu visuel final de mon projet, voici comment il est structuré :

- Page de connexion, création de compte, accès admin et mode démo (pour prof)
- La page suivante a le même affichage pour tout le monde mais l'accès à certaines fonctionnalités est restreint en fonction de l'utilisateur :
  - Le client peut commander et créer une recette mais pas dans le même onglet que le cdr
  - Le cdr peut créer une recette dans un onglet dédié, passer une commande et regarder ses informations (solde cook, recettes créées, commandes passées…)
  - L'admin peut consulter le tableau de bord (Dashboard, supprimer des recettes et des cdr, réapprovisionner les produits, rajouter produits et fournisseurs)
  - Le professeur (mode démo) qui a accès aux informations suivantes :

1. Voir le nombre de clients ;
2. Voir nombre puis noms des CdR et le nombre de leur recette commandée ;
3. Voir nombre total de recettes ;
4. Voir liste des produits ayant un stock inférieur à 2 fois leur quantité minimale ;
5. Voir la liste des recettes comprenant le produit rentré au clavier (avec quantité) ;