Rapport DM – Guillon Bony Rémi et Poupet Vincent

1 – La database

Nous avons créé notre base de données en nous basant sur le modèle E/A que nous avons réalisé (joint avec ce rapport). A partir de ce MCD, nous avons réalisé le MLD suivant :

- Client (Identifiant, Mot_de_passe, Nom_Client, Numero_de_tel, Credit_cook, CdR)
- Commande (Ref_Commande, Date, Prix, #Identifiant)
- Recette (Nom_Recette, Type, Descriptif, Prix_Vente, Rémunération, Compteur, #Identifiant)
- Composition_Commande (#Nom_Recette, #Ref_Commande, Quantite_Recette)
- Fournisseur (Ref_Fournisseur, Nom_Fournisseur, Numero_tel_Fournisseur)
- Produit (Nom_Produit, Categorie, Unite, Stock, Stock_min, Stock_max, #Ref_Fournisseur)
- Composition_Recette (#Nom_Recette, #Nom_Produit, Quantite_Produit)

Afin de permettre l'utilisation de notre database, nous avons créé des scripts de génération et de remplissage que vous pourrez retrouver dans le zip déposé. Ces scripts sont à exécuter dans MySQLWorkbench avant de lancer notre application. Nous avons aussi créé des scripts similaires en ''.txt'' que nous utilisons lorsque nous cliquons sur le bouton ''Reset database''. Ils nous permettent à tout moment d'effacer la database actuelle et d'en générer immédiatement une nouvelle, remplie de données de test.

Afin de pouvoir interagir avec la base de données depuis C#, nous avons créé deux fonctions distinctes, l'une pour les requêtes de type "Select" et la seconde pour les autres types de requête (Insert, Update, Delete, ...).

2 – Options de codage

Pour pouvoir nous coordonner et être capable de coder individuellement sur des parties distinctes du code, nous avons fait le choix d'utiliser GitHub. De ce fait, chacun avait la dernière version du code dans son intégralité, à tout moment.

Concernant l'affichage, nous avons décidé de développer une application WPF. Nous n'étions pas présent le semestre précédent et n'avions donc pas de connaissance préalable. Cependant, nous avons jugé qu'utiliser WPF nous permettrait d'obtenir un résultat plus professionnel.

Pour ce qui est du WPF, nous avons utilisé une Navigation Window, nous permettant ainsi de naviguer facilement entre les différentes pages que nous avons créées.

Pour faciliter l'affichage, plutôt qu'utiliser un Canvas ou un DocPanel, nous avons choisi d'utiliser une Grid car c'est ce qui nous semblait être le plus versatile et facile d'emploi. En effet, cela nous permettait de découper notre Window en un quadrillage et de placer les éléments où nous le souhaitions sans risquer de déformation de la page, si sa taille venait à être changer par l'utilisateur.

Nous avons utilisé plusieurs types de champs en fonction de nos besoins comme des boutons, des labels, des TextBox, des ListView. Pour ce qui est des mots de passe client et administrateur nous avons utilisé des PasswordBox qui, même si elles ne garantissent pas vraiment une confidentialité du mot de passe, nous permettent d'obtenir une interface crédible. Concernant la gestion des valeurs dans nos ListView, nous avons choisi d'utiliser un système de DataBinding pour faciliter l'accès et la modification de chaque item.

Permettre à l'utilisateur de rentrer n'importe quelles données dans les TextBox s'est révélé dangereux pour la stabilité de notre application. Si jamais l'utilisateur venait à entrer des guillemets ou des caractères avec accent, MySQL ne parvenait pas à interpréter les requêtes et l'application plantait. Nous avons donc dû développer des sécurités afin d'empêcher cette situation de survenir. Cela est contenu dans la fonction ''Caractere_interdit'' contenu dans chaque fichier.

Nous avons rendu notre application « Fool-proof » en restreignant les inputs acceptés en fonction de la TextBox et de son utilité dans notre code. Par exemple, essayer d'ajouter une recette dans votre panier sans indiquer de quantité, vous retournera une erreur.

Afin de tester l'ensemble de nos fonctions d'interaction avec MySQL, nous avons réalisé des tests unitaires.

3 – Interprétation du cahier des charges

Nous avons considéré qu'il y avait trois états pour un client : simple client, CdR, client restreint.

- Un simple client peut, depuis la page ''Client'', passer des commandes et devenir CdR en cliquant sur le bouton correspondant.
- Un CdR peut passer des commandes et créer des recettes depuis la page ''CdR''.
- Un client restreint peut passer des commandes mais le bouton lui permettant de devenir CdR reste non-cliquable. Cet état s'obtient en étant passé manuellement par l'administrateur de CdR à client restreint.

Nous avons choisi de donner la possibilité à l'administrateur de créer de nouveaux produits. Durant ce processus, nous le laissons choisir lui-même le stock minimal et maximal qu'il souhaite avoir pour ce produit, en veillant à ce que le stock maximal soit bien supérieur au stock minimal auquel cas une erreur sera retournée.

Chaque fois qu'une recette est créée, le stock minimal et maximal de chaque produit la composant évolue de la façon suivante :

- Le nouveau stock maximal = stock maximal actuel + 2 fois la quantité du produit utilisée.
- Le nouveau stock minimal = stock minimal actuel + 1 fois la quantité du produit utilisée.

Nous avons considéré que l'augmentation du prix de la recette survenait avant ou après une commande mais pas au milieu : commander deux fois une recette ayant un compteur de 9 sur la même commande nous fera payer le même prix pour chacune des recettes. L'augmentation du prix et de la rémunération du CdR survient après le paiement. Nous avons fait ce choix car les clients ne sont pas censés être au courant des détails du système d'augmentation de prix. Dans ce contexte, il paraitrait étonnant d'afficher un certain prix durant le passage de la commande et de leur faire payer un montant différent une fois arrivé à l'étape de validation de paiement.

A l'heure actuelle si le solde d'un client n'est pas suffisant pour payer l'intégralité de la commande, le client est invité à payer le reste par CB. Comme indiqué dans le cahier des charges, cette page n'avait pas à être développée puisque nous n'avions pas à gérer l'aspect financier. Notre seule alternative est donc que tous les clients payent en crédits cooks. Afin de permettre le test de nos fonctions, nous avons donc ajouté un bouton permettant à un client de se donner des crédits cooks et ainsi pouvoir valider ses commandes. Après avoir triché la valeur du bouton valider est ré-évaluée. En effet, si le client possède assez de crédits cooks après avoir tricher, il ne sera plus invité à rejoindre la page pour payer en CB mais sera redirigé vers la page d'accueil et sa commande sera validée. Cette fonction serait

évidemment retirée dans le cadre d'une réelle mise en ligne de l'application et la page pour payer en CB serait développée.

Pour ce qui est du Top, nous avons fait le choix de recalculer les éléments le constituant à chaque fois que la page TOP est ouverte. Les tops sont calculés sur la semaine actuelle, c'est-à-dire à partir du début de la semaine actuelle.