Projet 5 – Utilisez les données publiques de l'OpenFoodFacts

Pour réaliser ce projet j'ai suivi, en partie, les différentes étapes proposées dans l'énoncé. Vous trouverez le code sous ce lien : <https://github.com/Baptiste68/P5>

# 1 – Organiser son travail

Pour commencer, je devais visualiser l'ensemble du programme afin de comprendre les fonctionnalités. Ainsi, j'ai dans un premier temps dessiné des schémas pour avoir une vue d'ensemble.

Une fois ces schémas réalisés, j'ai pu en faire ressortir les différentes "user stories". Ceci m'a permis de créer mon tableau agile.

Le tableau agile étant créé, je me suis mis à écrire la documentation du programme en y décrivant les fonctions. Cela était intéressant comme façon de procéder et m'a donné la possibilité de développer mon programme de façon clair, en aillant une ligne directrice et des objectifs.

# 2 – Construire la base de données

En me basant sur le Readme et le tableau agile, il a été possible de définir la base de données nécessaire.

Ainsi, une petite base de données a été modélisée via un MPD. Cette base contient 2 entités (food et category) et deux relations (saved et foodcate).

Grâce au MPD, il a été possible de créer le script SQL permettant la création de la base de données.

Pour le remplissage de la base, je me suis d'abord créé un petit échantillon test afin de très un nombre de données plus limité. Nous verrons donc le remplissage de la base de données via l'API en dernier.

# 3 – Construire le programme

Pour la structure du programme, j'ai opté pour un modèle composé de seulement 2 classes :

* Databaselink, qui sert à interagir avec la base de données (cf. point 4)
* Main, qui contient les fonctions d'affichage des menus et l'interaction avec l'utilisateur.

Le corps du programme se trouve donc dans le "main".

Je me suis donc basé dans un premier temps, sur les fonctions qui ne nécessite pas, ou peu, d'interaction avec la base de données :

* Afficher les menus (menu principal, menu catégories & aliment, menu des substituts sauvés et le menu qui permet de quitter le programme). Cela consiste à écrire dans le terminal les choix de l'utilisateur et d'attendre sa réponse.
* Gérer les réponses de l'utilisateur. Il a fallu ici écrire une fonction qui permet de vérifier l'entrée clavier suivant les différentes possibilités possible.
* Afficher les détails d'un aliment. Il faut ici prendre les éléments d'un aliment et les afficher.

# 4 – Interagir avec la base de données

La classe databaselink nous permet donc de se connecter a la base de données, d'y envoyer des requêtes et de fermer la connexion.

Cette classe nous a permit de compléter notre "main" afin de finaliser les fonctions.

Pour les menus catégories et aliments, nous avons besoin d'une requête qui sélectionne l'id et le nom de la catégorie ou de l'aliment pour ensuite pouvoir afficher la liste des noms. J'ai dû faire attention ici a bien retourner l'id de l'élément choisi et non le nombre entré par l'utilisateur.

Pour le menu des substituts sauvé, l'idée est la même, il faut juste sélectionner les éléments que nous souhaitons afficher.

La difficulté se trouve plus dans la recherche et la sauvegarde de substitut. En effet, il a fallu traiter les plusieurs cas possible :

* Un substitut avec une meilleure note est trouvé -> on continu le processus
* Aucun substitut avec une meilleure note n'est trouvé -> relancer la recherche pour un substitut de même valeur
* Aucun substitut avec une même valeur n'est trouvé -> relancer la recherche pour un substitut de valeur inférieur

C'est pourquoi, j'ai opté pour la création d'une fonction récursive qui permet de changer la condition (valeur du signe : '<', '=' ou '>').

# 5 – Peupler la base de données

Une fois mon programme opérationnel avec ma base de données test, il faillait peupler la base de données via l'API d'OpenFoodFact.

Ma difficulté principale était de trouver les bons champs et les bons produits à sélectionner.

MA seconde difficulté était de conditionner les champs pour éviter les erreurs dans les "insert". Notamment du aux ' ' ' ou ' \'.