Olena ANDROSOVYCH

Projet 5 « Utilisez les données publiques de l'OpenFoodFact »

Introduction et présentation du plan

Lors de Projet 5, on développe le programme Pur Beurre pour chercher des substituts plus sains aux produits consommés régulièrement et les enregistrer dans une liste.

* Dans une première partie de ce document j’expliquerai la séquence de travail sur le projet.
* Dans une deuxième partie je citerai les difficultés rencontrées lors du projet et les solutions trouvées.
* À la fin du document vous trouverez le lien vers le code du programme (sur Github)

## I. Le choix de l'algorithme et la séquence de travail

1. Lors du projet 5, tout d’abord, j’ai créé un plan du travail et découpé le projet par les tâches et sous-tâches. Pour l’organisation ces modules j’ai utilisé Trello.com.

2. Dans la deuxième étape, j’ai construit la base de données (le diagramme et le code).

3. Par la suite, j’ai créé les classes dans un fichier classes.py :

* Data\_OFF pour récupérer les informations d’Open Food Facts en utilisant le library request,
* Data\_PB pour créer notre base de donnés avec MySQL,
* Display\_data pour réaliser l’interaction de l’utilisateur avec la BdD,
* Saved\_data pour afficher les résultats de requête.

Chaque classe comporte des méthodes afin d’assurer la fonctionnalité des objets de classes (fichier classes.py).

A l’intérieur d’une classe **Data\_OFF** on a développé des méthodes suivantes :

* *def \_\_init\_\_* pour créer l’instance de la classe (une liste des catégories),
* *def get\_data* pour obtenir les informations nécessaires par certaines catégories.

Une classe **Data\_PB** comporte des méthodes suivantes :

* *def \_\_init\_\_* pour créer un instance de la classe (données récupérées d’OFF),
* *def connect*, *def commit\_connection*, *def close\_connection* pour se connecter avec la BdD, commiter les changements et fermer la connexion respectivement,
* *def create\_db*, *def drop\_db* pour créat la BdD et pour la suprimer,
* *def create\_category, def create\_product, def create\_substitute* pour la création des tables,
* *def find\_from\_table* for récupérer certaines données de la table
* *def insert\_category*, *def insert\_product*, *def insert\_substitute* pour insérer les donnés dans les tableaux category, product et substitute respectivement
* *def sql* pour exécuter le code mysql

Une classe **Display\_data** contient en soi les méthodes suivantes:

* *def \_\_init\_\_* pour créer des instances de la classe,
* *def nutricompare* pour comparer les nutri-scores de deux produits,
* *def super\_input* pour vérifier le format de l'input,
* *def choose\_option* pour choisir le produit que l'utilisateur souhaite remplacer,
* *def find\_substutute* pour trouver le substitut et affiche sa description,
* *def add\_data\_choice* pour sauvegarder un substitut dans la liste des produits sains

Et dans une classe **Saved\_data,** les méthodes :

* *def display\_result* pour afficher la liste des produits sains enregistrée par l’utilisateur.

3.2. Les constantes ont été placées dans un fichier constants.py. L’algorithme se place dans un fichier pb\_app.py.

Description d’un algorithme :

1. Tout d’abord, l’utilisateur doit choisir une de deux options : commencer ou continuer le programme. En cas de « commencer », le programme récupère de donnés de l’OFF, remplit de tableaux *category* et *product* par ces données, efface la liste de produit sain s’il existe déjà.

2. Boucle. Sur l’étape suivant le programme propose trois options :

1 - Quel aliment souhaitez-vous remplacer ? (L'utilisateur doit appuyer le chiffre « 1 »)

2 - Retrouver mes aliments substitués (L'utilisateur doit appuyer le chiffre « 2 »)

Q - Quitter - (L'utilisateur doit appuyer un lettre « Q »)

Option 1 (l’utilisateur tape « 1 ») :

1.1 L’utilisateur sélectionne la catégorie du produit de la liste en entrant un chiffre qui corresponde à cette catégorie + « entrée ».

1.2 L’utilisateur sélectionne le produit de la liste des produits de cette catégorie en entrant le chiffre qui corresponde à ce produit + « entrée ».

1.3 Le programme propose des substituts avec ses nutri-scores. L'utilisateur choisit un de ces aliments en entrant le chiffre qui corresponde à ce produit + « entrée ».

1.4 Le programme affiche le nom du produit choisi, sa description, un lien vers la page d'Open Food Facts concernant cet aliment, son nutri-score, sa catégorie, et des magasins où on peut l'acheter.

1.5 Le programme propose à enregistrer le résultat de la recherche.

L’utilisateur appuie « Y » s’il veut enregistrer le résultat, sinon il doit appuyer « N ».

1.6 Le programme relance l'étape 2 en proposant à l'utilisateur à choisir une de trois options.

Option 2 (l’utilisateur tape « 2 »)

2.1 Le programme affiche la liste des produits enregistrés

2.2 Le programme relance l'étape 2 en proposant à l'utilisateur à choisir une de trois options.

Option 3 (l’utilisateur tape « 3 »)

Le programme se termine.

## II. Les difficultés rencontrées et les solutions trouvées

Lors de mon travail sur le projet, j’ai des difficultés avec l’installation mysql, la requête de donnés de l’OFF et intégration de la requête de la BdD dans le code Python.

La décision : j’ai cherché les réponses dans la documentation d’API, sur le forums spécialisés et demandé à mon mentor pour les sujets qui n’étaient pas clairs pour moi.

En plus, au début, il était difficile pour moi de comprendre la logique globale de la fonctionnalité du programme.

La décision : j’ai découpé le travail par les tâches et sous-tâches. Alors, petit à petit, j’ai réalisé toutes les fonctionnalités du programme.

## III. Le code du programme est disponible sur Github :

<https://github.com/OlenaChe/Project5>