

Ce projet est disponible sur Github à l'adresse:

<https://github.com/mehdi-monfort/-P5-OpenFoodFacts>

## **1. Ma démarche:**

### **1. Début du projet :**

J'ai d'abord rédigé le README.md avec l'explication des différents besoins de l'API avec le détails des différents menus et choix. Puis est fait le projet agile détaillant le parcours du projet et la manière dont j'allait m'y prendre jusqu'à ça concrétisation, le modèle physique de données et enfin je me suis m'y à la programmation.

### **2. Le programme :**

1. L'idée était d'apprendre à faire les requêtes avec la bases de données de l'OpenFoodFact, de les utilisées dans un code python pour les afficher sur la console. J'ai commencé par cela.
2. Puis est crée la database en python (qui utilise un fichier SQL disposant des différentes tables), avec l'aide de la [documentation](#). J'ai, par la suite, rempli les tables à l'aide de la commande « .get », testé les différentes requêtes d'affichage nécessaire au programme.
3. J'ai terminé par la « main » où est crée le menu et d'où les différents appels sont fait pour l'affichage console.

## **1.1. Constitution de l'API :**

Le programme est constitué de :

- README.md détaillant le fonctionnement de l'application et son cahier des charges.
- Le requirements.txt avec tous les modules nécessaires à l'execution du programme.
- Un fichier config.py avec la configuration de la base de données (modifiable par l'utilisateur)
- La main.py contenant le menu de l'application et ses différent choix.

Un dossier Database contenant les fichiers :

- create.py pour la suppression et création de la base de données
- request.py permettant de remplir la base de données
- table.py avec le code SQL des différentes tables
- database.py avec les différentes requêtes nécessaire à l'application

Un dossier Class contenant les différents fichiers de classes (une par table).

## **2. Difficultés :**

1. Ma difficultés principale est de mettre le code en orienté objet, il me restera la main à améliorer. Mon mentor m'a aidé à débloquer la situation grâce à ses explications ce qui m'a facilité la tâche par la suite et devrait se ressentir sur la suite de la formation. J'ai donc dû refaire mon code pour l'orienté objet et également pour y insérer les requêtes directement en python sans passer par un fichier SQL comme c'est le cas pour mes tables. J'ai également dû travailler sur mes « cursor.execute » qui ne se trouvaient pas tous dans le même fichier et qui étaient faites, en partie, dans la main.
2. Désormais toutes mes requêtes se font dans le même fichier en python à l'aide de la commande cursor.execute(), le code n'en est que plus compréhensible.
3. La création des menus, je n'avais aucune idée de comment m'y prendre sans boucle while à l'infini.
4. Pour la création de la table plusieurs à plusieurs et principalement la nécessité de l'utiliser dans le programme.
5. Lors de ma première soutenance je n'utilisais pas de classes pour utiliser les produits par exemple, ce qu'il m'a été demandé de faire. Mes lacunes en orienté objet m'ont causé quelques difficultés mais ça m'a été très utile. Je pense ma compréhension sur ce point bien supérieure.
6. Mon code en orienté objet permettant les requêtes avec l'api d'OpenFoodFacts et le remplissage des tables prenait un temps considérable (2,5min/catégorie) car morcelé entre la Main() et les classes Category() et Product(), ce que j'ai pu corriger en faisant tout cela dans la même classe (actuellement : 6s/catégorie).

### **2.1. Améliorations possibles :**

#### **1. Dans le code actuel :**

1. Découper la méthode create\_database() de la classe Request() avec une partie requête, une partie tri et une dernière pour l'ajout en table de la base de données.
2. Créer un fichier avec les différents messages (ex : les prints, erreurs...).
3. Créer une table magasin avec relation plusieurs à plusieurs.
4. Utiliser un mot de passe chiffré plutôt que de l'avoir en clair dans le programme.

#### **2. En supplément :**

1. Demander à l'utilisateur combien et quelles catégories il souhaite.
2. Utilisation de Tkinter pour créer une interface graphique ou le module console\_menu pour garder l'affichage console.
3. Utilisation de Flask pour en faire une application web, cela permettrait de se rapprocher d'une demande client.

## **3. PEP8 et vérification :**

Après utilisation de flake8 à la racine, la majorité des erreurs rapportées sont dû à la longueur des lignes (toutes des prints). Supérieur à 79 caractères.

Ainsi que 2 erreurs de syntaxe qui semblent être une erreur de flake8, cette erreur a été rapportée à de nombreuses reprises comme étant un faux-positif.