

Projet 8 - Créez une plate-forme pour amateurs de Nutella

Étudiant : Guillaume OJARDIAS Mentor : Erwan KERIBIN
Mentor évaluateur : Babacar SYLLA

Sommaire

I. Présentation	2
I.1. Liens du projets	2
II. Démarche de création	2
II.1 Algorithme de recherche d'un substitut	2
II.2. Découpage du projet	2
II.3. Plan de tests	3
III. Bilan	3
III.1. Découpage du projet	3
III.2. Plan de tests	3
III.3. Utilisation de Django	3

I. Présentation

Ce projet reprend les fondements du projet 5 : utiliser l'API proposé par *Open Food Facts* pour proposer à l'utilisateur · rice un produit de substitut au produit qu'il · elle recherche dans la base de données.

La particularité sur ce huitième projet : l'utilisation du *framework Django* pour la réalisation d'une interface Web.

Les fonctionnalités minimales demandées sont les suivantes :

- permettre à l'utilisateur · rice d'avoir un compte ;
- voir les substituts proposé sur une page ;
- permettre de sauvegarder un favoris, consultable sur la page “*Mes Aliments*” de l'utilisateur · rice ;
- utiliser un template et une charte graphique.

I.1. Liens du projets

- Le code source du projet est disponible sur la plate-forme GitHub à cette adresse : <https://github.com/GuillaumeOj/Pur-Beurre>.
- Le site est visible en ligne à cette adresse : <http://projet-8.ojardias.io/>.
- Le Trello du projet est accessible ici : <https://trello.com/b/TWtodZpE/purbeurre>.

II. Démarche de création

Pour réaliser le projet, j'ai suivi les étapes suivantes :

- initialisation du projet Django ;
- mise en place du système de gestion utilisateur ;
- création des modèles de données (**Product** et **Category**) ;
- insertion des produits de la base de données Open Food Facts grâce à une commande personnalisée (**init_db**) ;
- recherche d'un substitut en fonction du produit recherché par l'utilisateur · rice ;
- ajout de la fonctionnalité de favoris et du modèle associé (**Favorite**).

II.1 Algorithme de recherche d'un substitut

Pour être le plus précis possible et afin de proposer à l'utilisateur · rice un produit meilleur que le produit initial, l'algorithme de recherche se base sur deux critères :

- le nombre de catégories en commun ;
- le nutriscore.

Le but est d'avoir le plus grand nombre de **catégories en commun** et un **nutriscore** inférieur pour le substitut par rapport au produit recherché.

Afin d'avoir de meilleurs résultats, les substituts proposés sont classés comme suit :

- le plus grand nombre de catégories en commun ;
- puis par le nutriscore le plus faible.

Ce choix ne permet pas forcément d'obtenir les produits avec un meilleur nutriscore en premier, mais assure que les premiers produits proposés sont le plus proche du produit recherché.

II.2. Découpage du projet

Le découpage du projet s'appuie sur le système d'application de Django. Ainsi, nous avons 6 applications :

1. **homepage** regroupe les pages standards du site (accueil et mentions légales) ;
2. **openfoodfacts** permet la communication avec l'API d'Open Food Facts et assure le remplissage de la base de données ;
3. **product** gère les produits de la base de données ;
4. **search** effectue les différentes recherches dans la base de données ;
5. **users** gère les utilisateurs ;
6. **config** configure le projet (application générée avec la commande **startproject**).

De plus il y a trois répertoires généraux qui sont :

1. **static** pour le stockage des fichiers statiques (CSS, JS, images, etc.) ;
2. **tests** pour regrouper l'ensemble des tests du projet (reproduit la hiérarchie des application du projet) ;
3. **templates** pour le *template* de base et les *templates* d'erreur.

II.3. Plan de tests

Pour faciliter la mise en place des tests sur le projet, je me suis appuyé sur l'utilisation du *package* **coverage**. Ce package m'a permis de mettre en relief les parties du code non couvertes par les tests.

Afin de couvrir uniquement les applications créées pour les besoins du projet, j'ai exclu les répertoires **config** et **migrations** de l'analyse.

Les tests unitaires couvrent principalement deux aspects :

1. tests des méthodes de modèle ;
2. tests des différentes vues.

III. Bilan

III.1. Découpage du projet

Le découpage utilisé sur le projet pourrait être revu sur deux points en particulier.

Dans un premier temps, il pourrait être intéressant de trouver un nom plus approprié pour l'application **homepage**. Ce nom était logique au démarrage du projet, car elle devait contenir uniquement la page d'accueil. Seulement, elle est plus vouée à contenir l'ensemble des pages "générales" du site, ainsi **general** serait plus approprié.

Ensuite les *templates* auraient pu être regroupés dans un répertoire commun pour faciliter le travail. Étant donné le nombre d'applications, il devenait difficile sur la fin du projet de s'y retrouver avec les différents *templates* disséminés dans chaque application.

III.2. Plan de tests

Je n'ai pas réalisé mon projet en suivant une méthodologie de travail *Test Driven Development* pour une raison particulière.

Dans ma logique d'apprentissage, j'ai besoin de maîtriser un minimum un sujet pour réussir d'une part, à prendre du recul et d'autre part, à mener une réflexion de construction de projet.

Or, lors du démarrage de ce projet, je ne connaissais ni **Django**, ni l'outil de test **unittest**. Sur la fin, en revanche, j'ai su surmonter ce blocage, grâce notamment à **coverage**, à une meilleure maîtrise de **Django** et à une meilleure compréhension d'**unittest**.

La pertinence et la fiabilité de mes tests pourraient être améliorées de manière significative, cependant, je reste persuadé que ce genre d'exercice deviendra plus naturel au fur et à mesure de mon expérience de développeur.

III.3. Utilisation de Django

L'une des grandes difficultés pour ma part sur cet exercice était l'apprentissage de **Django**.

J'ai ainsi découvert, un *framework* très complet permettant la réalisation de projets vastes "facilement". Cependant, la courbe d'apprentissage a été plus longue que ce que j'avais pu imaginer en débutant ce projet de site.

J'ai aussi découvert un concept qui m'a beaucoup parlé et qui explique une des raisons pour lesquelles j'ai mis beaucoup de temps avant d'avancer significativement dans la réalisation de ce projet.

Django a, pour un débutant, une part importante de magie. C'est-à-dire qu'il intègre de nombreux outils ne demandant qu'à être utilisés par le développeur **out of the box**.

Or, je n'arrive pas à utiliser un outil si je ne comprends pas un minimum son fonctionnement. Mais grâce au coup de pouce d'un étudiant de mon parcours et de mon mentor, j'ai pu débloquer la situation et aller de l'avant.