Projet 8 - Créez une plateforme pour amateurs de Nutella

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

Présentation

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

La particularité sur ce huitième projet, est 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 visible dans une catégorie "Mes Aliments"
- Utiliser un template imposé ainsi qu'une charte graphique

Liens du projets

- le code source du projet est disponible sur la plateforme 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 accesible ici : *https://trello.com/b/TWtodZpE/purbeurre*

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 données issues de l'API Open Food Facts grâce à une commande personnalisée
- recherche d'un substitut en fonction du produit rentré par l'utilisateur · rice
- ajout de la fonctionnalité de favoris

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 entre le produit recherché et les substituts.

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

- le 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 permet de s'assurer que les premiers produits proposés soit le plus proche du produit recherché.

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 (acceuil et mentions légales)
2. `openfoodfacts` qui permet la communication avec l'Api d'Open Food Facts et assure le remplissage de la base de données
3. `product` qui gère les produits de la base de données
4. `search` qui effectue les différentes recherches dans la base de données
5. `users` pour la gestion des utilisateurs
6. `config` pour la configuration du projet

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

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 crées pour les besoin du projet, j'ai exclue les répertoires de configuration et de migrations de l'analyse.

Les tests unitaires couvrent principalement deux aspects :

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

Bilan