# Projet 8 : Plateforme pour amateurs de Nutella - Code source - Deploiement sur Heroku

https://github.com/horlas/OC_Project8 https://pbquality.herokuapp.com/

## Généralités

Ce projet a été pour l'occasion de se mettre en situation professionnel réelle, de fournir au client un projet respectant un cahier des charges donné et d'apprendre l'utilisation du Framework python : Django.

## Contexte

Des restaurateurs soucieux de la qualité de ce que leur client pourrait manger au quotidien, décident de créer une plateforme web qui propose des aliments de substitution de meilleure qualité pour ce qu'ils consomment au quotidien .

## Choix technologiques

Le site statique est développé en Django. Il intègre un template Bootstrap pour le Front end permettant la gestion du responsive. Le back-end est codé bien évidement en Python. Les fonctionnalités font appel à l'API Openfood Facts : pour un aliment saisi il renvoie 6 propositions de produit dans la même catégorie avec un nutriscore meilleur si il en existe. Le site permet à l'utilisateur de créer un compte et de sauvegarder ses recherches. La base de données est en Posgresql. Nous avons utilisé l'Administration de Django pour la gérer. Le déploiement du site est fait sous Heroku. Les tests utilisent le module de test de Django et le module coverage. Le projet est versionné sur Github.

## Parcours Utilisateur

L'utilisateur est invité à saisir le nom d'un produit alimentaire. Six aliments correspondant au mot de sa recherche lui sont proposés. Il en sélectionne un (nous appelons ce produit : produit selectionné / selected product). Six aliments présentant des qualités nutritives meilleures et faisant partie de la même catégorie lui sont alors renvoyés. A ce stade l'utilisateur doit soit créer un compte soit se connecter si il veut sauvegarder sa recherche. Suite à cette connexion il peut sauvegarder la recherche en selectionnant l'aliment de substitution de son choix . Lui est alors renvoyé une page présentant le couple produit sélectionné / produit substitué et la possibilité de faire une nouvelle recherche. L'utilisateur à accès à son compte (nom et mail ) il n'a, pour l'instant , pas la possibilité de changer son mot de pass. Il a accès aussi à la page Aliment (icône « Carotte ») qui lui renvoie toutes les sauvegardes qu'il aurait pu faire . L'utilisateur peut se déconnecté , il est renvoyé alors vers la page d'accueil.

### Gestion des erreurs

Si l'utilisateur fait une saisie vide, un message lui indique et la champ de recherche est à nouveau disponible.Si l'utilisateur fait une saisie incompréhensible , le champ de recherche lui est renvoyé avec la notification que la saisie est erronée (gestion de l'erreur 500). Si l'utilisateur demande une page inexistante (erreur 400) , une notification

## Algorithme de recherche sur l'API OpenfoodFacts (OFF) :

Les quatre fonctions constituant l'algorithme de recherche se trouvent dans le fichier methods.py de l'application. Elles se décomposent en deux pôles : la génération du premier envoi: liste de six produits, qui permet le choix du produit selectionné/ selected_product, et la génération du deuxième envoi : liste de six produits, qui permet le choix du produit substitué/ substutut_product.

query_off(query) : envoie une requête à l'API OFF en utilisant le mot 'query' saisi par l'utilisateur en amont sur le site web. Extrait la liste des produits à renvoyer. Retourne le résultat traité par la fonction data_process() suivante.

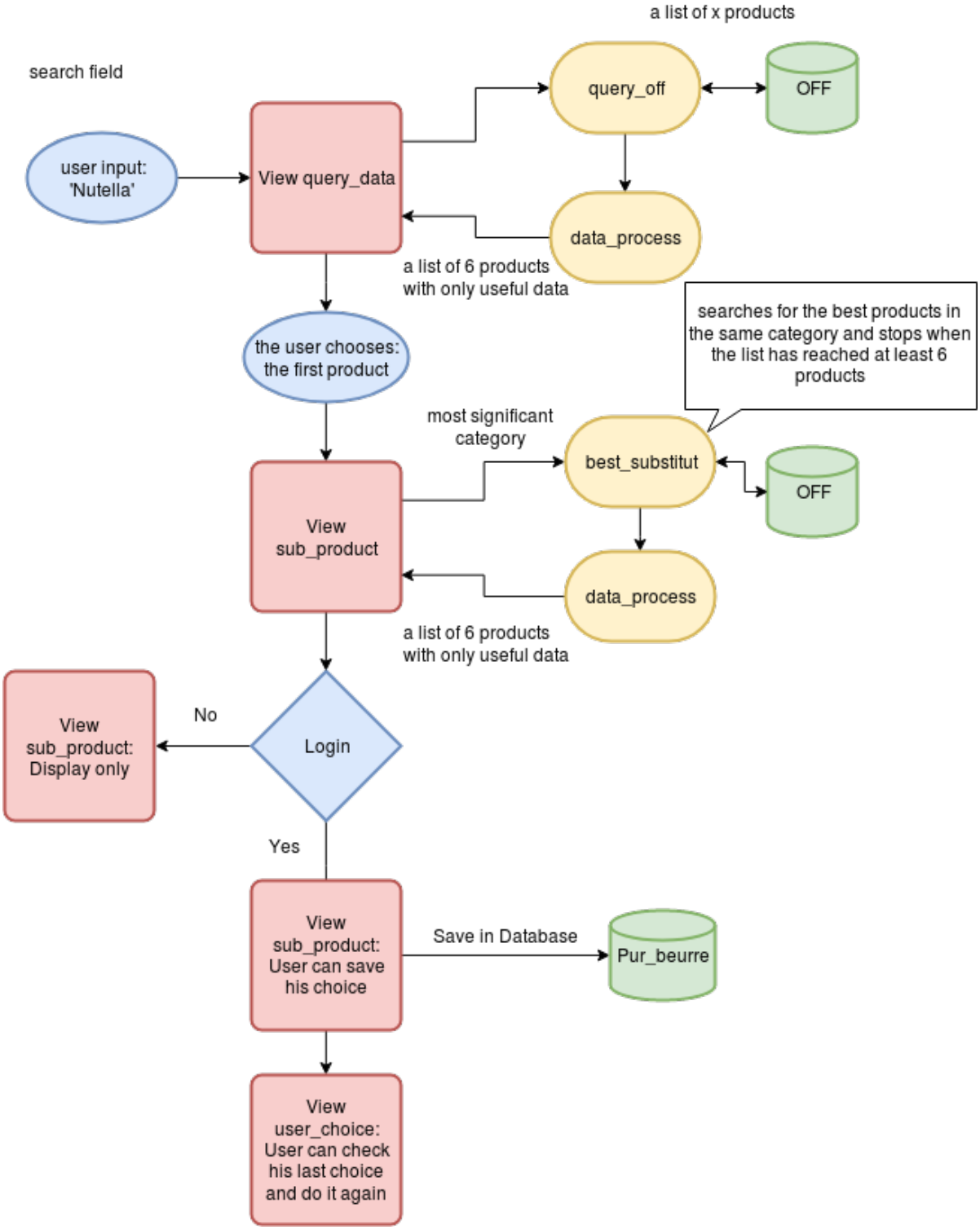
data_process(products) : en charge d'extraire une liste de 6 produits parmi les données renvoyées par la requête à OFF et de ne garder que les sept items utiles au site (nom, catégorie, image, url, nutriscore, ing_nutrition, magasin). La catégorie

conservée est la dernière de la liste de catégories renvoyées par OFF car c'est la plus significative. C'est à partir de cette catégorie que sera faite la deuxième recherche sur OFF.

request_off(cat, ns) : en charge de construire l'url d'appel à l'API OFF , en intégrant un certain nombre de paramètres , dont la catégorie et le nutriscore. Pour permettre la recherche sur la même catégorie et prenant en paramètre le nutriscore générer par la fonction best_substitut..

best_substitut(cat) :

Une itération est réalisée sur une liste de nutriscore de « A » à « E ». Chaque requête à l'API rempli une liste en commençant par le nutriscore « A », dès que cette liste dépasse une limite de 6 (pour permettre le traitement suivant de la fonction data_process traitant 6 produits) , l'itération s'arrête. Nous limitons ainsi les requêtes à l'API et le temps de réponse. Nous garantissons le fait de ne renvoyer que les meilleurs produits.

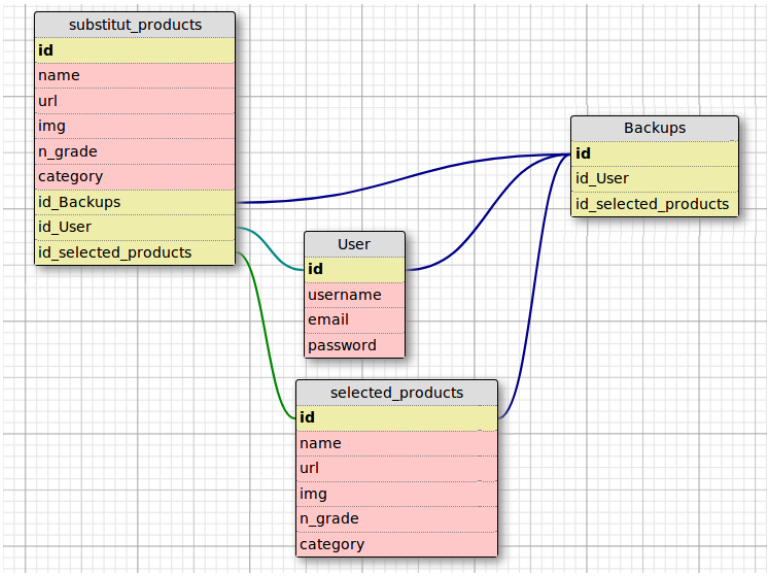


## Structure de la Base de Données

Nous faisons appel à la base de données que lors de l'envoi de l'ordre de sauvegarde par l'utilisateur. Cette base de données ne contient que les sauvegardes utilisateur . Les produits sélectionnés ne sont pas rattachés à un utilisateur et peuvent être multiples . Nous pouvons imaginer comme évolution possible de faire une recherche sur la BDD pour savoir si ce produit existe déjà avant de l'enregistrer,  limitant ainsi la taille de cette table.

La table Backups reprend l'identifiant du produit selectionné en l'associant à l'identifiant de l'utilisateur.

La table  substitut_product enregistre les données du produit substitué le rattachant à la l'id de la sauvegarde, l'id de l'utilisateur et l'id du produit selectionné , garantissant ainsi l'intégrité du renvoi des sauvegardes dans la page Aliment.




## Quelques mot sur la partie FrontEnd

La partie Frontend intègre un template Bootstrap : https://blackrockdigital.github.io/startbootstrap-creative/  Il permet une application entièrement responsive.  Une bibliothèque font correspondant à l'icône demandée de la « carotte » a été crée spécialement. L'authentification de l'utilisateur se fait grâce à des fenêtres modal générées en Ajax (utilisation de Jquery) . Le choix des couleurs , des photos a été transmis dans le cahier des charges  et est respecté autant que possible.

L'affichage des produits est fait grâce à Bootstrap Cards , la pagination est assurée par le module pagination de Django .

Tous les crédits du site peuvent être retrouvés sur la page Mentions légales (lien en bas de chaque page du site)