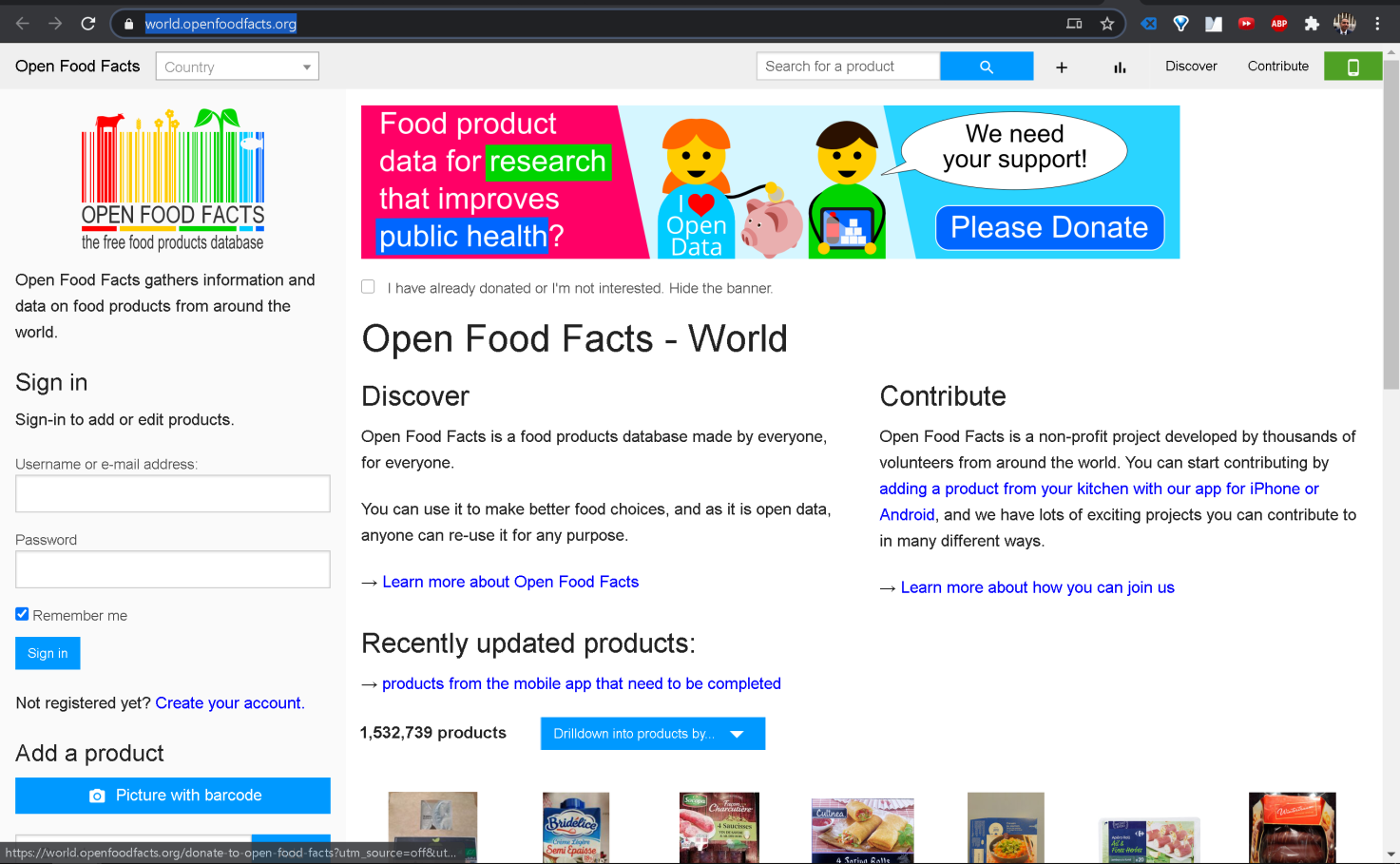
# **Mise en situation :**

La startup **Pur Beurre** connait bien les habitudes alimentaires françaises. Leur restaurant, **Ratatouille**, remporte un succès croissant et attire toujours plus de visiteurs sur la butte de Montmartre.

L'équipe a remarqué que la clientèle voulaient changer leur alimentation mais ne savaient pas par quoi :

* Remplacer le *Nutella* par une *pâte aux noisettes*
* Oui, mais *laquelle* ? *Et dans quel magasin l'acheter* ?
* …

L’idée est de créer une webApp [Django](https://www.djangoproject.com/) avec un front React JS et une base de données MongoDB) qui interagit avec l’[API](https://openfoodfacts.github.io/openfoodfacts-server/api/ref-v2/) d’**Open Food Facts** pour i) *récupérer les aliments*, ii) *les comparer* et iii) *proposer un substitut* plus sain / non allergène, etc. ([exemple de call](https://world.openfoodfacts.org/api/v0/product/737628064502.json) de l’API, [documentation](https://openfoodfacts.github.io/api-documentation/#1GeneralInformation) de l’API).



# **Cahier des charges :**

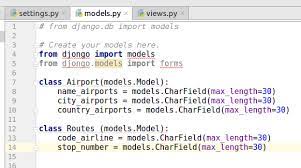
# **‘’Idée’’ de Parcours utilisateur :**

L'user crée un compte, s’y connecte et on lui affiche les choix suivants :

1. *Quel aliment souhaitez-vous remplacer* ?
2. *Entrez le code barre de l’aliment à remplacer*?
3. *Retrouvez mes aliments substitués*!

Si l’user sélectionne(**1**), les questions suivantes sont alors posées :

* *Choisissez la catégorie d’aliment* [*Plusieurs propositions associées]*? *L'utilisateur sélectionne la catégorie souhaitée et appuie sur entrée*
* *Choisissez l'aliment. L'utilisateur sélectionne l’aliment à remplacer et appuie sur entrée*
* Le programme propose :
  + - 1. *un substitut*,
      2. *sa description*,
      3. *un magasin ou l'acheter* (le cas échéant),
      4. *un lien vers la page* d'[Open Food Facts](https://world.openfoodfacts.org/) (*concernant cet aliment*).
* Le client a la possibilité d'enregistrer le résultat (dans la base de données : *mes aliments substitués*).

# **Construire et interagir avec la base de données**

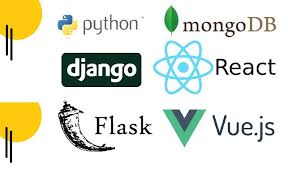
* Créez la base de données sur [Atlas](https://www.mongodb.com/cloud/atlas/register) en ajoutant vos collègues de groupe et en gérant les privilèges (un owner, un read only, …)

**Pour vous aider :**

* Commencez par vous questionner sur les informations dont vous avez besoin :
  + - *Quelles informations allez-vous enregistrer* (dans MongoDB)
    - *Quelles données allez-vous manipuler* ?

Remarque : Les utilisateurs de la startup Pur Beurre-IPSSI sont français. Il n'est pas nécessaire d'importer l'intégralité des données de l’API, d'autant plus qu'elles sont si grandes que cela ralentirait considérablement votre application (et ferait fuir vos utilisateurs).

* Ecrivez le code Python qui insèrera les données récoltées de l'API dans la base
* Vous avez la base de données et vous avez les classes, Bravo ! À présent, permettez à votre utilisateur d'interagir avec la DB.
* Commencez par travailler sur le système de question réponse (*input*, *validation des champs*). Puis concentrez-vous sur la recherche : quelles requêtes ? Dans quelle(s) collections(s) ?
* Enfin, cherchez comment enregistrer les données générées par le programme pour que l'utilisateur les retrouve.

# **Construire l’application Django**

Afin de garantir une bonne séparation des responsabilités entre les différentes couches du projet, vous aurez avantage à séparer les **classes** responsables de :

* Télécharger les données à partir de l’API de OpenFoodFacts ;
* Nettoyer la data obtenues ;
* Enregistrer les données nettoyées et faire des recherches en base de données ;
* Présenter les données dans le client.

Ces différentes classes pourront être développées de manière indépendantes avant de les faire travailler ensemble au sein du projet

# **Compétences à valider**

* Travailler en respectant une méthodologie de projet Agile
* Respecter les bonnes pratiques de développement
* Développer une application proposant les fonctionnalités attendues par le client
* Gérer une base de données NoSQL avec un ORM
* Organiser son code pour garantir sa lisibilité et sa maintenabilité

# **Livrables attendus (15 pts)**

* **Un repo** [**Github**](https://github.com/) où on trouve :
  + Un fichier README où vous expliquez : **(3 pts)**
    - *Comment lancer les scripts* (de la WebApp, ...)
    - *Mentionnez le lien de votre* [trello](https://trello.com/fr) *avec la répartition des tâches à travers les membres de votre groupe* (à faire/en cours/fait)
    - …
    - *Lien vers le support de présentation* (par exemple [google slides](https://www.google.fr/intl/fr/slides/about/)) où on trouve *les principaux résultats* (pour votre manager)
  + Le code générant la WebApp et permettant de la déployer (par ex sur Heroku) **(7 pts)**

# **Modalités de présentation du travail (5 pts)**

Votre présentation pourra prendre cette forme (à titre indicatif) :

|  |  |
| --- | --- |
| 5 min. | Rappel de la problématique, présentation du jeu de données (API), de la DB, de la WebApp |
| 15 min | Explication de l’approche pour mettre en place votre solution et démo |
| 10 min | Séance de questions-réponses |