

Système de Recommandation Alimentaire Basé sur le Deep Learning et l'Encodage Sémantique des Recettes

Siham Kalach

Encadré par **Mohamed Charradi**

École Nationale des Sciences Appliquées d'Al Hoceima (ENSAH)

sihamkalach@etu.uae.ac.ma

1 ABSTRACT

Les systèmes de recommandation jouent un rôle crucial dans l'accompagnement des utilisateurs dans leurs choix alimentaires, surtout dans un contexte où les préférences, les contraintes nutritionnelles, et les restrictions culturelles deviennent de plus en plus spécifiques. Les approches traditionnelles, souvent basées sur des filtres collaboratifs ou des modèles simples de similarité textuelle, peinent à capturer la richesse sémantique des descriptions d'ingrédients et l'intention réelle de l'utilisateur. Ce projet propose une approche fondée sur l'apprentissage profond pour améliorer la pertinence des recommandations alimentaires.

Nous avons construit un système intelligent capable de suggérer des produits alimentaires à partir d'une description de recette librement saisie par l'utilisateur, enrichie par des contraintes personnelles (comme le pays de vente, les niveaux nutritionnels, ou la présence d'alcool). Le cœur du système repose sur un encodeur de phrases pré-entraîné (MiniLM) pour extraire des représentations vectorielles profondes, couplé à un moteur de recherche vectoriel pour identifier les produits les plus similaires. Le tout est encapsulé dans une architecture web intégrée, combinant un backend Django, une base de données MongoDB et une interface utilisateur React.

Ce travail démontre l'intérêt des méthodes d'encodage sémantique et d'indexation vectorielle dans les systèmes de recommandation, tout en soulignant les limites des approches classiques et la valeur ajoutée du deep learning dans l'interprétation contextuelle des besoins utilisateur.

2 INTRODUCTION

Dans un monde où les choix alimentaires sont de plus en plus personnalisés, les systèmes de recommandation jouent un rôle essentiel pour guider les consommateurs en fonction de leurs préférences, de leurs besoins nutritionnels ou de leurs contraintes culturelles et sanitaires. Qu'il s'agisse de suivre un régime sans allergènes, de respecter des principes religieux ou d'adopter une alimentation plus équilibrée, les utilisateurs attendent aujourd'hui des recommandations précises, contextuelles et pertinentes.

Les approches classiques de recommandation, qu'elles soient fondées sur des règles prédéfinies ou sur le filtrage collaboratif, montrent rapidement leurs limites face à la complexité sémantique des données textuelles issues des descriptions de produits ou de recettes. Ces méthodes peinent à comprendre le sens réel des requêtes des utilisateurs, surtout lorsque celles-ci sont formulées en langage naturel, avec des formulations libres, ambiguës ou subjectives.

Face à ces défis, l'intelligence artificielle, et en particulier l'apprentissage profond, offre de nouvelles perspectives. L'utilisation de modèles de représentation textuelle modernes permet aujourd'hui de capturer le sens profond des phrases, d'établir des relations sémantiques fines entre les ingrédients d'une recette et les produits alimentaires disponibles, et de proposer des suggestions plus cohérentes et personnalisées.

Dans ce projet, nous proposons la conception d'un système de recommandation alimentaire intelligent, fondé sur l'encodage sémantique des recettes et des produits alimentaires à l'aide d'un modèle de type Transformer. À partir d'une description libre fournie par l'utilisateur, enrichie de contraintes nutritionnelles ou géographiques, notre système est capable d'identifier et de suggérer les produits les plus adaptés à ses besoins. Ce projet illustre ainsi la synergie entre traitement automatique du langage naturel, apprentissage profond et développement web pour répondre à une problématique concrète et actuelle : aider les consommateurs à mieux choisir, en fonction de leurs profils, dans une offre alimentaire toujours plus vaste.

3 ÉTAT DE L'ART

Les systèmes de recommandation occupent aujourd'hui une place centrale dans les applications numériques, avec des usages variés allant de la recommandation de films et de produits à l'accompagnement nutritionnel personnalisé. Historiquement, les premières approches reposaient sur le filtrage collaboratif, une méthode consistant à analyser les préférences croisées entre utilisateurs pour prédire leurs choix futurs. Si ces techniques ont démontré leur efficacité dans des contextes où les données utilisateur sont abondantes, elles présentent de sérieuses limites dans le domaine alimentaire, notamment face aux problèmes de démarrage à froid (cold start) et à l'incapacité de prendre en compte des contraintes personnalisées comme les allergies, les régimes ou les spécificités culturelles.

En parallèle, les approches basées sur le contenu, qui exploitent les caractéristiques des produits (ingrédients, valeurs nutritionnelles, catégories) pour proposer des éléments similaires à ceux déjà appréciés, offrent une personnalisation plus fine. Cependant, elles peinent à comprendre le sens réel des descriptions formulées en langage naturel, souvent imprécises, subjectives ou complexes. Dans ce contexte, l'émergence de modèles d'apprentissage profond dédiés au traitement du langage naturel a marqué une avancée majeure. Des modèles pré-entraînés comme BERT, SBERT ou MiniLM permettent aujourd'hui de transformer les textes en représentations vectorielles denses, capables de capturer la sémantique profonde des phrases. Ces embeddings rendent possible la comparaison contextuelle de recettes et de produits, même lorsque les formulations diffèrent sur le plan lexical.

Des outils comme FAISS (Facebook AI Similarity Search) permettent d'exploiter efficacement ces vecteurs pour effectuer des recherches par similarité à grande échelle. Plusieurs travaux récents commencent à explorer cette approche dans le domaine alimentaire, mais la majorité des systèmes restent limités à des moteurs de recherche simples, reposant sur des filtres classiques et non sur une compréhension du sens. Ainsi, notre projet s'inscrit dans une nouvelle génération de systèmes de recommandation, capables de traiter des descriptions d'utilisateurs exprimées librement, d'en extraire le sens à l'aide d'un modèle de type Transformer, et de proposer des produits pertinents tout en respectant les contraintes nutritionnelles, culturelles ou diététiques. Cette approche permet de dépasser les limites des systèmes existants en offrant une interaction plus intelligente, fluide et personnalisée.

4 MÉTHODOLOGIE

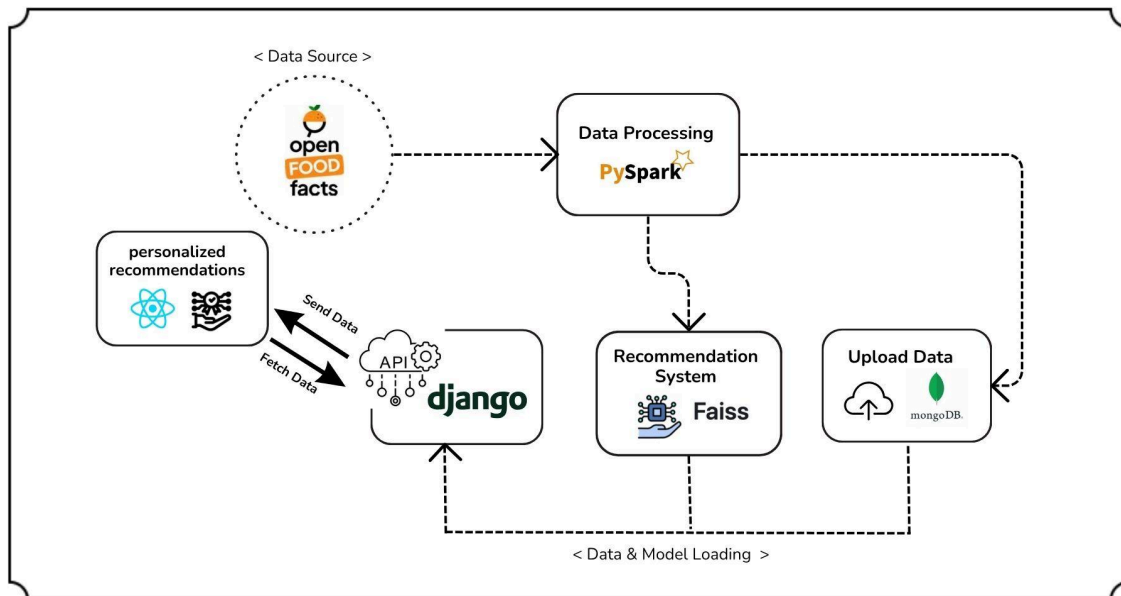


Figure 1 – Pipeline global de traitement et de recommandation

Pour réaliser ce projet, nous avons suivi une méthodologie en plusieurs étapes. Nous avons d’abord nettoyé les données pour corriger les erreurs et les incohérences. Ensuite, nous avons exploré et analysé les données afin de mieux les comprendre. Après cela, nous avons transformé les données textuelles en vecteurs numériques à l’aide de techniques d’embedding. Ces vecteurs nous ont permis de mesurer la similarité entre les recettes et les produits. Une fois les recommandations générées, nous avons appliqué des filtres selon les préférences de l’utilisateur (comme les allergènes). Enfin, nous avons préparé une interface web simple pour permettre à l’utilisateur d’interagir facilement avec le système.

Data et vise à offrir une mise en pratique concrète des concepts théoriques abordés, en combinant des techniques de traitement de données massives, d’analyse exploratoire, d’ingénierie des fonctionnalités, ainsi que des approches d’intelligence artificielle. Pour réaliser ce projet, nous avons adopté une méthodologie structurée en plusieurs étapes, couvrant l’ensemble du cycle de développement d’une application intelligente fondée sur les données.

Nous avons commencé par nettoyer le jeu de données Open Food Facts afin de corriger les erreurs, supprimer les doublons et résoudre les incohérences, notamment dans les colonnes nutritionnelles. Une analyse exploratoire a ensuite été menée pour mieux comprendre la distribution des données et identifier les variables clés.

Nous avons ensuite appliqué des techniques d’embedding sémantique pour transformer les descriptions textuelles (recettes et produits) en vecteurs numériques, capables de capturer la signification profonde des textes. Cela nous a permis de mesurer la similarité sémantique entre une recette saisie par l’utilisateur et les produits disponibles.

Les données enrichies et nettoyées ont été stockées dans une base de données MongoDB, conçue pour accueillir efficacement les structures flexibles et hétérogènes du jeu de données. MongoDB nous permet également d'exécuter des requêtes rapides sur les filtres nutritionnels et géographiques (comme le pays, la teneur en sucre, ou la présence d'alcool). Sur cette base, nous avons développé un back-end modulaire avec Django, capable de :

- télécharger et charger dynamiquement le modèle de recommandation,
- vectoriser les requêtes utilisateurs,
- interroger l'index vectoriel FAISS pour identifier les produits les plus similaires,
- et appliquer des filtres personnalisés selon les préférences de l'utilisateur (ex. : allergies, niveaux nutritionnels).

Enfin, nous avons construit un front-end web interactif avec React, qui permet à l'utilisateur :

- de saisir sa recette ou ses ingrédients,
- de sélectionner des filtres (pays, niveau de sucre, sel, alcool, etc.),
- et de recevoir dynamiquement les recommandations générées via une API REST.

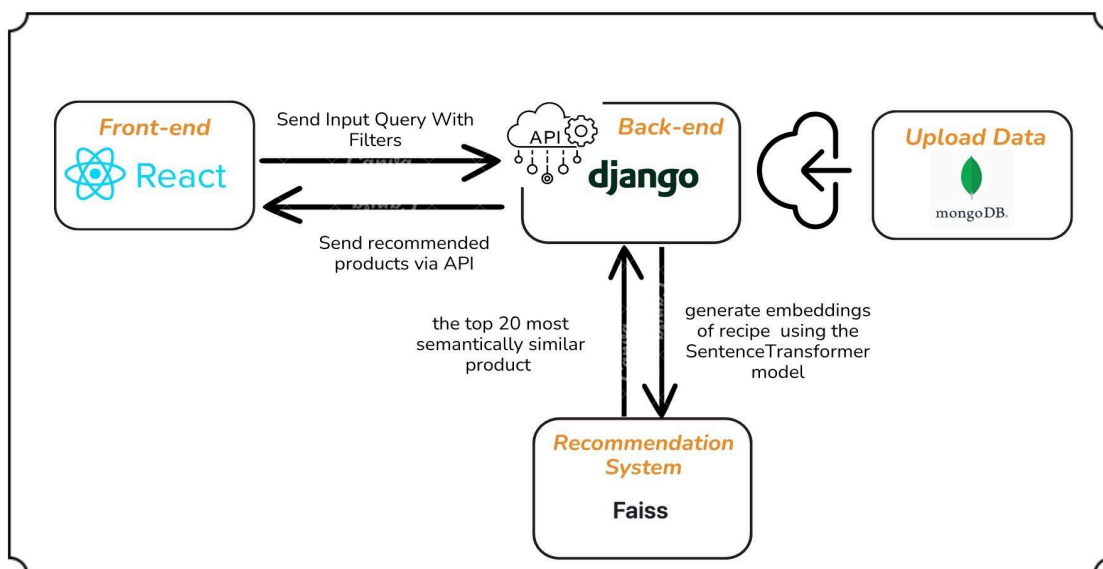


Figure 2 – Architecture d'intégration web du système.

4.1 PRÉPARATION DES DONNÉES

4.1.1 Description du jeu de données

Le corpus utilisé dans ce projet est issu de la base Open Food Facts, une base de données ouverte et collaborative qui regroupe des informations détaillées sur plus de 2 millions de produits alimentaires provenant de plus de 200 pays. Ces données sont accessibles librement via le site officiel : <https://world.openfoodfacts.org/data>. Les données sont disponibles sous différents formats, notamment CSV, JSON et MongoDB dump, ce qui permet une grande flexibilité pour leur traitement. Le fichier principal (*en.openfoodfacts.org.products.csv*) contient généralement plus de 3 millions de

lignes et environ 180 colonnes, bien que certaines colonnes soient partiellement remplies selon les produits.

Afin de mieux appréhender la répartition géographique des produits référencés, nous avons analysé la colonne *countries*, qui peut contenir plusieurs pays séparés par des virgules. La Figure 1 ci-dessous illustre les 10 pays les plus représentés dans les 100 000 premières lignes à l'aide d'un diagramme circulaire, mettant en lumière une forte représentation de certains pays européens.

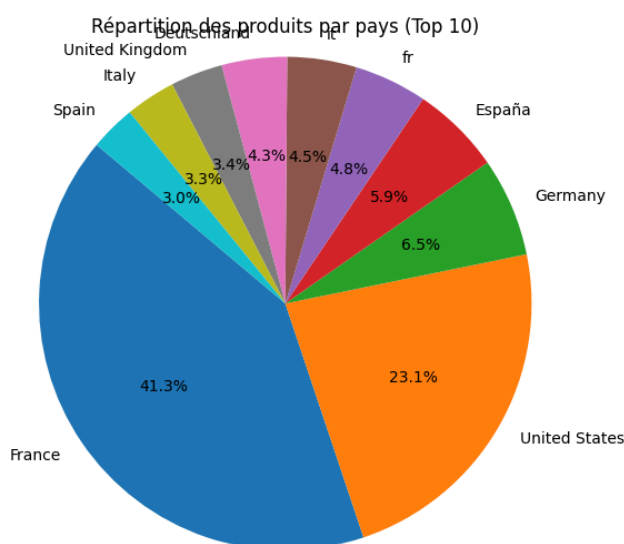


Figure 3 – Répartition des produits alimentaires par pays d'origine (Top 10 pays les plus représentés)

4.1.2 Analyse, Nettoyage et Feature Engineering des Données Nutritionnelles

Le dataset nutritionnel brut issu d'Open Food Facts a d'abord été soumis à une analyse approfondie afin de garantir la qualité des données. Une première étape a permis d'identifier les colonnes contenant des valeurs manquantes, essentielles pour anticiper les éventuelles stratégies de traitement. Ensuite, des analyses statistiques descriptives ont été réalisées sur les variables nutritionnelles clés telles que l'énergie, les matières grasses, les sucres et le sel. Cela a permis de détecter des distributions anormales et de mieux comprendre la structure globale des données. Un nettoyage rigoureux a ensuite été appliqué, visant à supprimer les lignes présentant des incohérences logiques, notamment celles où les graisses saturées ou trans dépassaient la quantité totale de matières grasses. De plus, des valeurs aberrantes, telles que des teneurs en matières grasses supérieures à 100g pour 100g de produit, ont été filtrées afin d'assurer la cohérence du dataset. Après un nettoyage complet des données, nous avons obtenu un total de 820 228 lignes exploitables. Cependant, en raison des contraintes liées à l'intégration et au traitement d'un volume aussi important dans la base de données, nous avons fait le choix de travailler, dans un premier temps, sur un échantillon de 100 000 premières lignes. Cet échantillon est représentatif et permet de mener des recommandations fiables tout en assurant une meilleure performance du modèle.

Une étape clé de ce travail a été le feature engineering, qui visait à enrichir le dataset avec de nouvelles colonnes dérivées, conçues pour aider l'utilisateur final à filtrer les produits selon son mode de vie alimentaire. Plusieurs nouvelles variables qualitatives ont ainsi été créées :

- sugar_level, salt_level, fiber_level, fat_level: pour classifier les produits selon leur teneur respective en sucre, sel et matières grasses .
- cholesterol_level et energy_level : fournissent une indication sur l'apport énergétique et la teneur en cholestérol .
- alcohol_presence : permettant d'identifier les produits contenant de l'alcool, ce qui est particulièrement pertinent pour les utilisateurs suivant un régime alimentaire halal.

Ces variables ont été visualisées sous forme de diagrammes à barres afin de comprendre leur distribution sur les 100 000 premières lignes nettoyées. Les Figures 2 à 7 ci-dessous illustrent la répartition des produits selon ces nouveaux attributs :

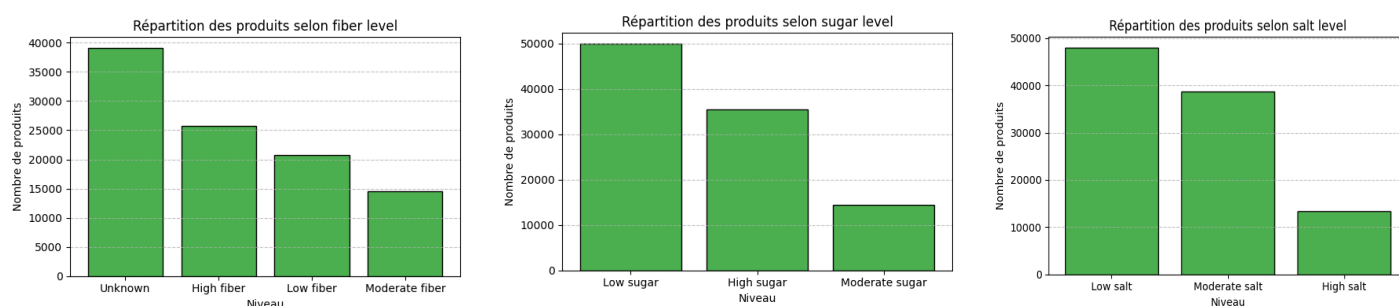


Figure 4 – Répartition des produits selon leur niveau en fibre en sucre en sel

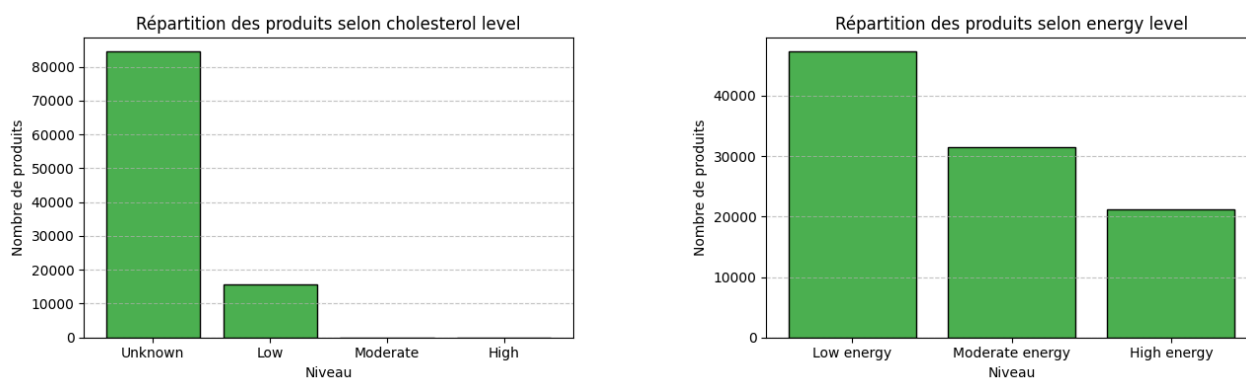


Figure 5 – Répartition des produits selon leur niveau en cholestérol et en énergie

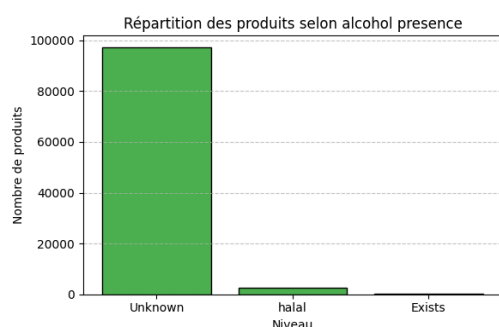


Figure 6 – Répartition des produits selon la présence ou l'absence d'alcool

De plus, des catégories spécifiques liées aux allergies et aux traces alimentaires ont été regroupées sous l'étiquette *allergens and traces*, facilitant la sélection de produits pour les personnes ayant des restrictions ou des sensibilités particulières. Grâce à ces transformations, l'utilisateur peut désormais explorer et filtrer les produits alimentaires en fonction de critères personnalisés de santé et de nutrition, rendant le dataset à la fois plus fonctionnel, accessible et orienté utilisateur.

4.2 ENCODAGE SÉMANTIQUE DES DONNÉES

4.2.1 Objectif de la vectorisation sémantique

Afin de permettre à notre système de recommandations d'identifier les produits alimentaires les plus pertinents à partir d'une description de recette, nous avons recours à un encodage sémantique. L'objectif est de transformer les textes (recettes ou listes d'ingrédients) ainsi que les descriptions de produits en vecteurs numériques. Ces vecteurs capturent le sens profond du contenu textuel, ce qui permet de mesurer la similarité sémantique entre une recette saisie par l'utilisateur et les produits de la base de données, indépendamment de la formulation exacte des textes.

4.2.2 Modèle utilisé : all-MiniLM-L6-v2

Pour cette tâche, nous avons utilisé le modèle all-MiniLM-L6-v2, un encodeur de phrases léger et performant développé par Microsoft. Ce modèle transforme chaque texte (ex. une recette ou une description produit) en un vecteur dense de 384 dimensions, optimisé pour les comparaisons sémantiques. Les raisons de ce choix sont les suivantes :

- **Performance** : Le modèle obtient une excellente performance sur les benchmarks de similarité sémantique, ce qui le rend adapté pour reconnaître des ingrédients ou formulations proches, même s'ils diffèrent lexicalement (ex. "fromage râpé" vs "emmental en poudre").
- **Efficacité** : Grâce à son architecture compacte à 6 couches, le modèle est rapide et peut traiter plusieurs centaines de recettes ou produits par seconde sur un CPU standard.
- **Compatibilité** : all-MiniLM-L6-v2 est compatible avec les bibliothèques comme *SentenceTransformers* et s'intègre facilement dans les environnements Spark ou Flask, ce qui facilite son déploiement dans notre pipeline de recommandation.

En vectorisant à la fois les recettes saisies par l'utilisateur et les produits du dataset Open Food Facts, nous pouvons calculer leur similarité sémantique (par exemple avec la similarité cosinus) et proposer des produits pertinents en fonction des ingrédients recherchés.

4.3 INDEXATION AVEC FACEBOOK AI SIMILARITY RESEARCH FAISS

4.1.5.1 Parlant de la bibliothèque FAISS

Une fois les paires Question et Réponse encodées en vecteurs denses à l'aide du modèle all-MiniLM-L6-v2, il est nécessaire de stocker efficacement ces représentations pour permettre une recherche rapide par similarité sémantique. Pour cela, nous utilisons FAISS (Facebook AI Similarity

Search), une bibliothèque développée par Facebook AI Research pour effectuer des recherches de similarité sur des vecteurs à grande échelle.

FAISS (Johnson et al., 2019) utilise des algorithmes optimisés de k-NN (k plus proches voisins) pour réduire la complexité de recherche de $O(n)$ à $O(\log n)$ dans des espaces de haute dimension, tout en préservant la précision grâce à la similarité cosinus, idéale pour les embeddings sémantiques.

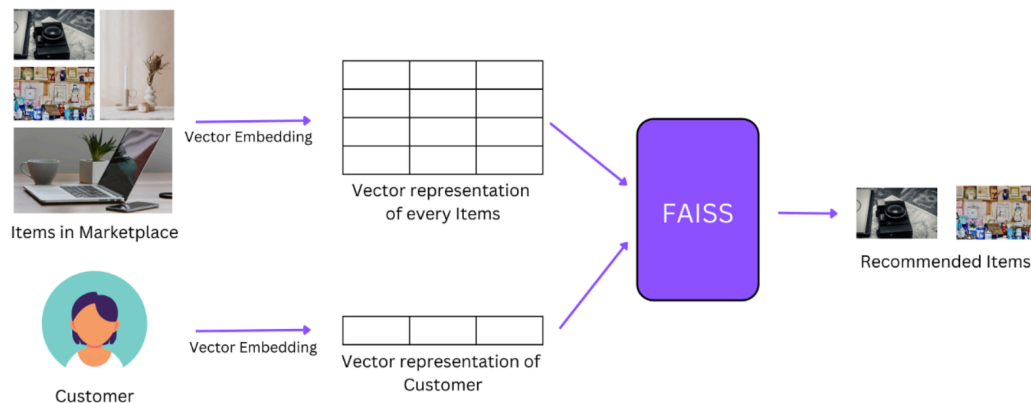


Figure 7. Visualisation du processus d'indexation vectorielle dans FAISS (vue d'ensemble).

4.1.5.2 Organisation et recherche des embeddings dans l'espace vectoriel

Après la génération des vecteurs denses (embeddings) à partir des textes, ces vecteurs sont organisés dans une matrice dense, qui constitue la base de données vectorielle. C'est à ce stade que commence l'indexation. Le type d'index est sélectionné en fonction de la normalisation des vecteurs:

- Le produit scalaire entre deux vecteurs u et v est donné par :

$$IP(u, v) = \sum_{i=1}^d u_i \cdot v_i$$

$$\cos(\theta) = \frac{A \cdot B}{\|A\| \cdot \|B\|} \Rightarrow \text{si } \|A\| = \|B\| = 1, \cos(\theta) = A \cdot B$$

- Si les vecteurs sont normalisés à la norme L2 (c'est-à-dire: $\|A\| = \|B\| = 1$), On utilise IndexFlatIP, basé sur le produit scalaire, ce qui revient à mesurer la similarité cosinus :

Dans ce cas, plus le produit scalaire est élevé, plus les vecteurs sont considérés comme similaires.

4.4 PIPELINE DE RECOMMANDATION

Avant de traiter les recettes saisies par l'utilisateur, un prétraitement a également été appliqué aux données issues de la base OpenFoodFacts. Pour cela, plusieurs colonnes textuelles pertinentes ont été concaténées : product_name, brands, main_category, categories et ingredients_text, afin de former une description textuelle représentative de chaque produit. Ce texte combiné a ensuite été vectorisé à l'aide du modèle de plongement sémantique all-MiniLM-L6-v2, et les vecteurs obtenus ont été stockés dans l'index FAISS pour un calcul rapide de similarité.

Lorsqu'un utilisateur saisit une recette en anglais, elle est d'abord nettoyée à l'aide de la fonction `clean_english_recipe(text)`. Cette fonction convertit le texte en minuscules, supprime les chiffres, les unités de mesure (comme "grams", "cups", "ml", etc.), la ponctuation, puis applique une tokenisation. Les mots vides (stopwords) sont retirés, et un lemmatiseur est utilisé pour ramener chaque mot à sa racine. Seuls les mots significatifs de plus de deux lettres sont conservés et recombinaés en un texte propre.

Exemple :

La recette: *1 cup of low-fat milk, 2 tablespoons cocoa powder, a pinch of salt and vanilla*

Après Nettoyage: *lowfat milk cocoa powder pinch salt vanilla*

Ce texte est à son tour vectorisé avec le modèle `all-MiniLM-L6-v2`, puis normalisé. Le vecteur résultant est comparé à ceux présents dans l'index FAISS à l'aide de la méthode `IndexFlatIP` (Inner Product), permettant de retrouver les 100 produits les plus similaires. Ces résultats sont ensuite filtrés selon les préférences et contraintes de l'utilisateur (régimes alimentaires, allergies, etc.), afin de proposer des recommandations pertinentes et personnalisées.

4.5 INTÉGRATION ET DÉVELOPPEMENT WEB

L'intégration technique du projet repose sur l'interconnexion fluide entre trois composants principaux : une base de données MongoDB Atlas, un back-end Django structuré autour d'une API REST, et une interface utilisateur moderne développée avec React.js. L'ensemble constitue une plateforme web intelligente, capable de gérer la recherche, le filtrage et la recommandation de produits alimentaires enrichis.

4.5.1. Base de Données : MongoDB Atlas

Nous avons choisi MongoDB comme solution de stockage des données pour sa flexibilité et sa capacité à gérer de gros volumes d'informations semi-structurées. Le dataset Open Food Facts, par nature hétérogène et contenant des champs partiellement remplis, s'adapte parfaitement au format document JSON proposé par MongoDB.

Chaque produit est enregistré sous forme de document dans une collection `product`, contenant :

- Des informations générales : `code`, `product_name`, `brands`, `main_category`, `categories`, etc ,
- Des données nutritionnelles : `energy_100g`, `fat_100g`, `sugar_100g`, `salt_100g`, etc.
- Des attributs dérivés (feature engineering) : `sugar_level`, `fiber_level`, `alcohol_presence`, `cholesterol_level`, etc.
- Des métadonnées : `countries`, `ingredients_text`, `allergènes`, `traces`, etc.

4.5.1. a configuration de la base de données MongoDB Atlas

Pour mettre en place cette base de données, nous avons suivi les étapes suivantes :

- Création d'un compte MongoDB Atlas sur <https://www.mongodb.com/cloud/atlas>.
- Création d'un cluster partagé (gratuit) et choix de la région la plus proche.

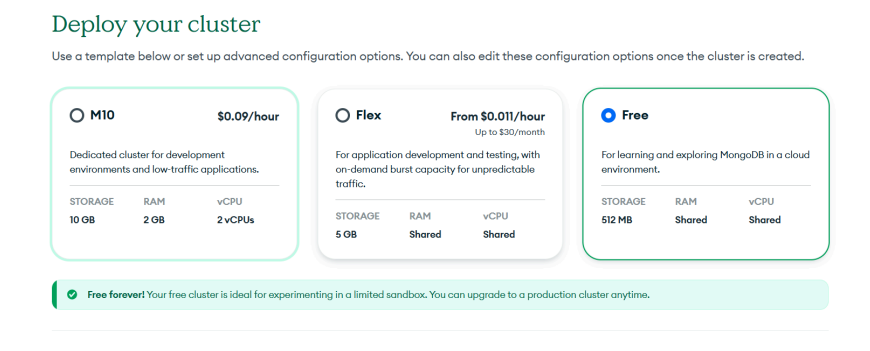


Figure 8. Création du OpenFood cluster.

- Ajout d'une base de données **openfood** avec une collection **product**.

Create Database

Database name ?

Open_Food

Collection name ?

product

Additional Preferences

Select

Cancel Create

Figure 9. Création de base de données openfood avec de la collection product.

- Ajout d'un utilisateur avec les droits readWrite.

Add New Database User

Create a database user to grant an application or user access to databases and collections in your clusters in this Atlas project. Granular access control can be configured with default privileges or custom roles. You can grant access to an Atlas project or organization using the corresponding [Access Manager](#).

Authentication Method

Password Certificate AWS IAM Federated Auth (MongoDB 7.0 and up)

MongoDB uses [SCRAM](#) as its default authentication method.

Password Authentication

siham_kalach

***** SHOW

Autogenerate Secure Password Copy

Database User Privileges

Configure role based access control by assigning database user a mix of one built-in role, multiple custom roles, and multiple specific privileges. A user will gain access to all actions within the roles assigned to them, not just the actions those roles share in common. You must choose at least one role or privilege. [Learn more about roles.](#)

Built-in Role

Select one built-in role for this user.

Read and write to any database

Custom Roles

Select your pre-defined custom role(s). Create a custom role in the [Custom Roles](#) tab.

Specific Privileges

Select multiple privileges and what database and collection they are associated with. Leaving collection blank will grant this role for all collections in the database.

Restrict Access to Specific Clusters/Federated Database Instances/Stream Processing Instances

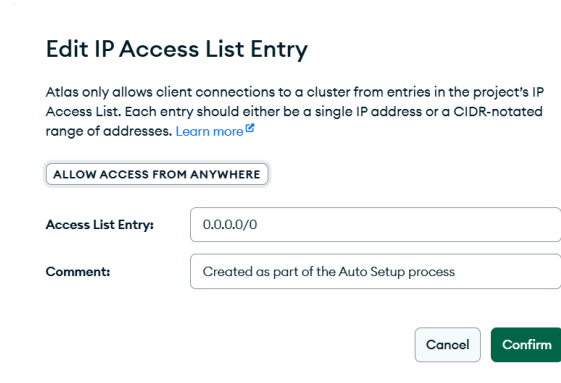
Enable to specify the resources this user can access. By default, all resources in this project are accessible.

Temporary User

This user is temporary and will be deleted after your specified duration of 6 hours, 1 day, or 1 week.

Figure 10. Création de l'utilisateur avec les droits readWrite

- Autorisation des connexions externes (IP : 0.0.0.0/0) pour permettre à Django d'accéder au cluster.



Edit IP Access List Entry

Atlas only allows client connections to a cluster from entries in the project's IP Access List. Each entry should either be a single IP address or a CIDR-notated range of addresses. [Learn more](#)

ALLOW ACCESS FROM ANYWHERE

Access List Entry: 0.0.0.0/0

Comment: Created as part of the Auto Setup process

Cancel Confirm

Figure 10. Autorisation des connexions externes .

- Récupération de l'URL de connexion au format : 'mongodb+srv://<user_name>:<password>@openfood.1pzkiaj.mongodb.net/?retryWrites=true&w=majority&appName=OpenFood'
- Connexion dans Django via mongoengine dans **settings.py**

```
#Connect to mongodb
from mongoengine import connect
import os
from dotenv import load_dotenv
load_dotenv()
db = os.getenv('DATABASE_NAME')
host = os.getenv("HOST")
connect(
    db=db,
    host=host
)
```

Figure 11. Connexion a base de données openfood

4.5. 1.b. Intégration avec Django : Endpoint de test et d'import

Pour valider la connexion et charger les données depuis un script Python, nous avons mis en place deux endpoints spécifiques :

- **GET /api/ping-mongo** : permet de tester que la connexion à MongoDB est bien établie.

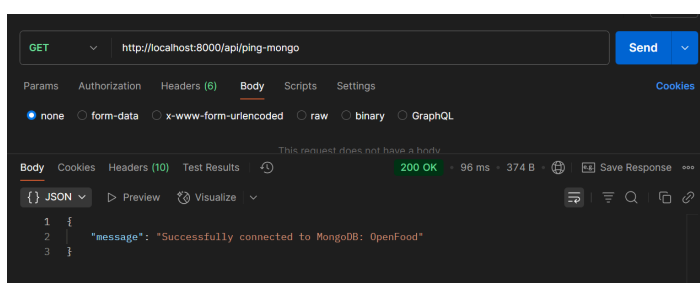


Figure 12. Résultat de /api/ping-mongo dans Postman

- **POST /import-products/** : permet de charger les données du dataset nettoyé dans la base MongoDB (utilisé une seule fois après le prétraitement).

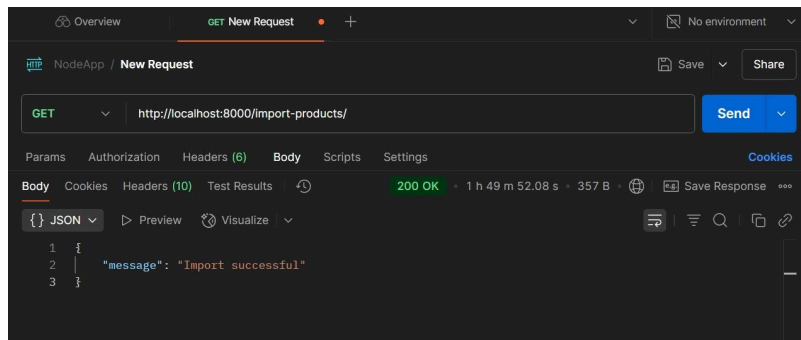


Figure 13. Résultat de Chargement des données dans MongoDB via /import-products/ dans Postman

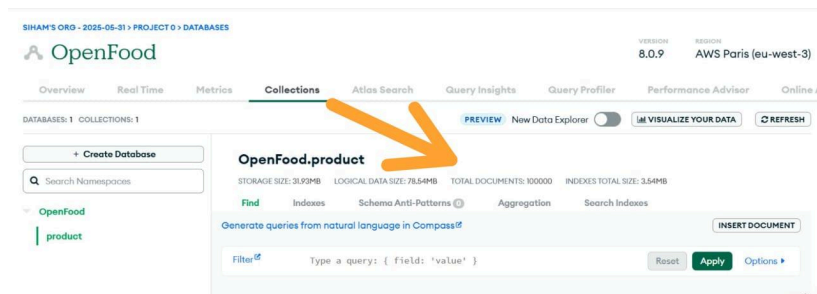


Figure 14. Résultat de Chargement des données dans MongoDB

4.5.2. Back-end : Django + API REST

Le back-end de notre système a été développé avec le framework **Django**, en combinaison avec **Django REST Framework**. Il constitue le cœur logique de l'application, permettant la communication entre la base de données MongoDB, le moteur de recommandation vectorielle FAISS, et l'interface utilisateur React.

4.5.2.1. Fonctionnalités principales

Le back-end assure les opérations suivantes :

- Connexion et interaction avec MongoDB Atlas via la bibliothèque mongoengine.
- Importation des produits issus du jeu de données Open Food Facts nettoyé.
- Moteur de recommandation basé sur la similarité sémantique, utilisant un modèle pré-entraîné (all-MiniLM-L6-v2) et un index vectoriel FAISS.
- API REST pour accéder aux données, filtrer les produits, obtenir les détails d'un produit ou générer des recommandations personnalisées.

4.5.2.2. Endpoints API définis

Voici la structure des routes définies dans `urls.py` :

```
urlpatterns = [
    path('api/ping-mongo', ping_mongo),
    path('import-products/', import_products_view, name='import_products'),
    path('api/products/', list_products),
    path('api/products/<str:product_id>', product_detail),
    path('api/filters/', filters_view, name='filters'),
    path('api/recommendation/', recipe_products_recommend, name='recipe_products_recommend'),
]
```

Figure 15. Liste des routes

Explication des routes :

- **GET /api/ping-mongo** : Vérifie la connexion à la base de données MongoDB.
- **POST /import-products/** : Importe les produits nettoyés dans MongoDB.
- **GET /api/products/** : Retourne la liste des produits avec pagination et filtres.
- **GET /api/products/<product_id>** : Détail d'un produit donné.
- **GET /api/filters/** : Retourne les valeurs uniques des filtres (marques, pays, niveaux, etc.).
- **POST /api/recommendation/** : Reçoit une recette + contraintes et retourne des produits similaires.

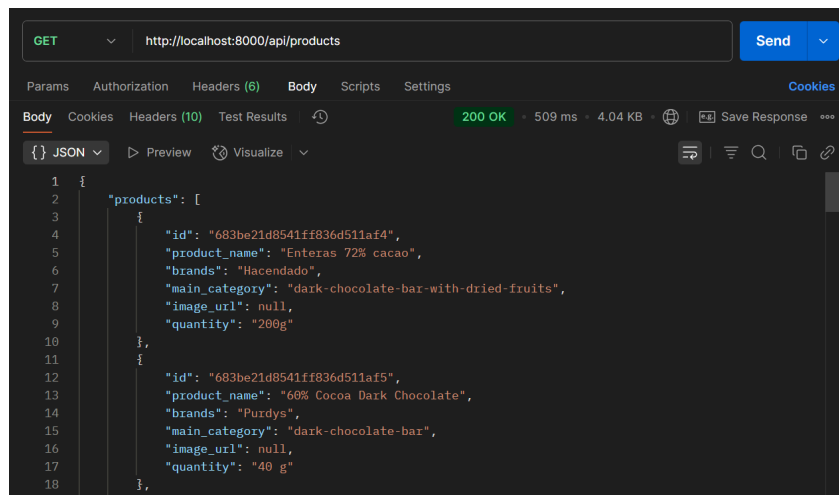


Figure 16. Résultat de endpoint /api/products/ dans Postman

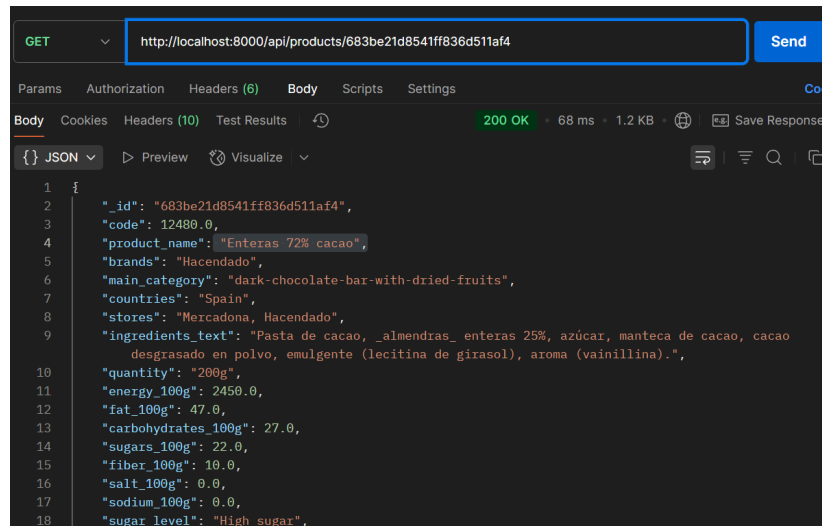


Figure 17. Résultat de endpoint /api/products/product_id dans Postman

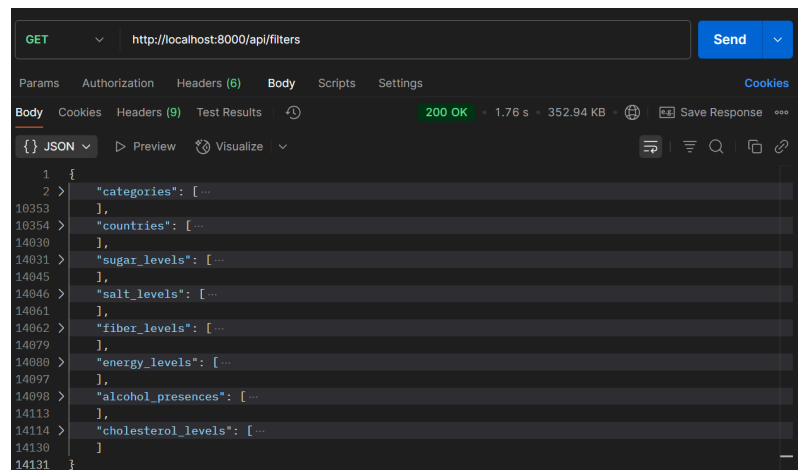


Figure 18. Résultat de endpoint /api/filters dans Postman

4.5.3. Front-end : Application React

L'interface utilisateur a été conçue avec React.js, afin d'offrir une expérience fluide, réactive et responsive. L'application front-end permet à l'utilisateur de :

- parcourir tous les produits avec filtres dynamiques,
- consulter les détails de chaque produit,
- obtenir des recommandations personnalisées à partir d'une recette,
- en apprendre davantage sur le projet (About),
- et contacter les auteurs (Contact).

4.5.3.1. Structure des pages React

1. Home

Page d'accueil simple, présentant un message de bienvenue et les objectifs de la plateforme.

- Présentation générale du projet
- Lien rapide vers les sections principales

2. Products

Page centrale permettant de parcourir la base de données des produits alimentaires. Elle inclut :

- Une grille de cartes produits (nom, image, marque, énergie, etc.)
- Une barre latérale de filtres dynamiques :
 - Catégories
 - Marques
 - Niveaux de sucre, sel, énergie, etc.
 - Pays de vente

Chaque carte comporte un bouton "Voir Détails" qui redirige vers la page produit.

3. Product Details

Page dédiée à un produit spécifique, affichant :

- Image du produit
- Marque, catégorie, description
- Valeurs nutritionnelles
- Indicateurs dérivés (sugar level, alcohol presence, etc.)
- Liste des allergènes ou traces

4. Recipe

Page permettant à l'utilisateur de recevoir des recommandations personnalisées. Elle contient :

- Une zone de saisie libre : description de la recette (ex. : *1 cup of milk, 2 spoons of cocoa powder...*)
- Des filtres personnalisés :
 - Pays
 - Niveau de sucre, sel, énergie, cholestérol
 - Présence ou absence d'alcool
- Un bouton "Find Similar Products"
- Une liste dynamique des produits suggérés par le moteur sémantique

5 Conclusion

Ce projet a permis de concevoir et de mettre en œuvre un système de recommandation alimentaire intelligent, s'appuyant sur les avancées récentes du deep learning appliqué au traitement du langage

naturel. Contrairement aux approches traditionnelles, souvent limitées par leur capacité à interpréter des requêtes textuelles complexes, notre solution repose sur une compréhension fine et contextuelle des descriptions de recettes fournies par les utilisateurs.

Grâce à l'utilisation d'un modèle d'encodage sémantique pré-entraîné, les textes sont transformés en représentations vectorielles riches, permettant de mesurer efficacement la similarité entre les besoins exprimés et les produits disponibles dans la base de données. L'intégration de ce système dans une architecture web moderne, combinant une API REST, une base de données flexible et une interface utilisateur intuitive, a démontré sa faisabilité et son potentiel d'utilisation dans des cas réels.

Les résultats obtenus ont mis en évidence la pertinence des recommandations générées, notamment dans des contextes sensibles comme les régimes spécifiques (pauvres en sucre, sans alcool, sans allergènes) ou les préférences culturelles. Ce travail ouvre la voie à des systèmes encore plus personnalisés, notamment par l'ajout futur d'un apprentissage supervisé sur des retours utilisateurs, ou l'adaptation du modèle à des contextes linguistiques et alimentaires variés.

En somme, ce projet illustre concrètement l'apport du deep learning dans le domaine des systèmes de recommandation, et montre comment des techniques d'intelligence artificielle peuvent répondre à des enjeux de santé, de culture et de confort d'usage au quotidien.