

# Rapport de Projet tuteuré : FoodRating

---

**Martin Niombela, Matthieu Reveyron, Thomas Lacomblez, Thomas Martin**

*Mars à Juin 2020*

**Tuteur académique :** Brigitte Groléas

**Etablissement et formation :** Institut Universitaire de Technologie de Vélizy, Licence  
Professionnelle SISW

<b>PRÉSENTATION DU PROJET</b>	<b>3</b>
INTRODUCTION	3
FONCTIONNEMENT	3
<b>TECHNOLOGIES UTILISÉES</b>	<b>4</b>
LANGAGES WEB	4
FRAMEWORK	4
SERVEUR	6
OUTILS ORGANISATIONNELS	6
<b>LISTE DES FONCTIONNALITÉS</b>	<b>6</b>
FONCTIONNALITÉS OBLIGATOIRES	6
FONCTIONNALITÉS OPTIONNELLES	7
<b>ORGANISATION DU PROJET</b>	<b>7</b>
VOLUME HORAIRE	8
DIAGRAMME DE GANTT	8
<b>ANALYSE DU PROJET</b>	<b>9</b>
PRODUIT ET NOTATION	9
FORUM ET COMMENTAIRES	11
BASE DE DONNÉES	12
<b>PROGRAMMATION DU SITE</b>	<b>12</b>
DÉBUT, MISE EN PLACE DE LA PAGE D'ACCUEIL	12
INSCRIPTION ET CONNEXION DE L'UTILISATEUR	13
MODIFICATION DES INFORMATIONS DE L'UTILISATEUR	15
AFFICHAGE DES PRODUITS SUR LE SITE	15
AJOUT DE LA FONCTIONNALITÉ DE RECHERCHE D'UN PRODUIT	17
LA PAGINATION AVEC KNP PAGINATOR	17
SYSTÈME DE NOTATION	18
AFFICHAGE DES PRODUITS DANS LE SITE : RÉVISION	19
AJOUT DES FONCTIONNALITÉS DE LA PARTIE "FORUM"	20
RECHERCHE DE PRODUIT : RÉVISION	21
AJOUT D'UNE LISTE DES CATÉGORIES	21
SYSTÈME DE COMMENTAIRE	21
CARROUSEL DE LA PAGE D'ACCUEIL	22
AJOUT DE LA PHOTO DE PROFIL	22

<b>FONCTIONNALITÉ DES PRODUITS SIMILAIRES</b>	<b>23</b>
<b>TABLEAU DE BORD</b>	<b>23</b>
<b>MESSAGE ENTRE LES UTILISATEURS</b>	<b>24</b>
<b>MAIL DE NOTIFICATION POUR LA CRÉATION D'UN COMPTE</b>	<b>24</b>
<b>FONCTIONNALITÉS D'ADMINISTRATION</b>	<b>25</b>
 <b><u>CONCLUSION DU PROJET : RETARDS ET DERNIÈRES FONCTIONNALITÉS</u></b>	 <b><u>25</u></b>
 <b>PROBLÈME RÉCURRENT : REDIRECTION</b>	 <b>25</b>
<b>PROBLÈME RÉCURRENT : HÉTÉROGÉNÉITÉ DES DONNÉES DES PRODUITS</b>	<b>26</b>
<b>PROBLÈME RÉCURRENT : VÉRIFICATION DE DOUBLON DANS UN TABLEAU MULTI-DIMENSIONNEL</b>	<b>26</b>
<b>FONCTIONNALITÉS INACHEVÉES</b>	<b>26</b>

# Présentation du projet

## Introduction

Le but de notre site web est de donner la possibilité à l'utilisateur de commenter et noter les produits alimentaires trouvables en grande surface. Beaucoup de sites web de notation existent, et ce dans plusieurs domaines : les jeux vidéos, les lieux de vacances, les restaurants, etc... Les utilisateurs peuvent noter en fonction de leur ressenti (une note généralement sur 5 ou sur 10). C'est pour cela que nous avons eu l'idée de développer ce site web. Cela permettra aux utilisateurs d'avoir un site sur la qualité des aliments, qui sera fait pour eux et par eux.

Afin de récupérer les données concernant un nombre de produits relativement important, nous utiliserons [l'API d'Open Food Facts](#), qui est open-source.

## Fonctionnement

### Partie "Utilisateur non connecté"

L'utilisateur non connecté pourra accéder à la page d'accueil, à toutes les fiches de produit (description, notes et commentaires, certaines statistiques), mais il ne pourra pas donner son avis. Il pourra également avoir accès au forum, sans pouvoir poster de message ou créer un nouveau sujet.

Cet utilisateur pourra créer un compte s'il n'en a pas ou se connecter s'il en possède un.

### Partie "Utilisateur connecté"

Il sera d'abord redirigé vers sa page personnelle (tableau de bord). Il pourra consulter et modifier les informations de son compte. Il pourra également contacter les responsables du site web via un formulaire de contact ou s'inscrire à la newsletter. Il pourra noter et laisser un commentaire sur un produit. Cette note et ce commentaire seront modifiables dans le cas où l'utilisateur aurait déjà noté ou commenté un produit. Il pourra ensuite créer des sujets dans la section "Forum" et y poster des réponses. Il pourra également avoir des liens avec d'autres utilisateurs. Enfin, cet utilisateur aura la possibilité de supprimer son compte, s'il le souhaite.

### Partie Administrateur

L'administrateur pourra gérer les utilisateurs qui possèdent un compte (exemple : bannissement, suppression) et gérer les commentaires (suppression, modification). Il pourra aussi gérer toute la partie "Forum" du site (suppression et modification de sujet, modération). L'administrateur pourra éditer les informations des produits répertoriés et les statistiques qui leurs sont associées.

## But du site web

Le but est de donner une vision globale aux utilisateurs sur ce qu'ils achètent comme produits alimentaires en grande surface. Il affichera aussi le Nutri-Score des produits, et toutes les informations sur les capacités nutritionnelles des produits répertoriés.

## Technologies utilisées

### Langages Web

Les langages Web qui seront utilisés pour notre projet sont les suivants :

- HTML 5
- CSS 3
- PHP

Ces langages ont été choisis car chaque membre du groupe possède des compétences dans ces langages, notamment PHP. Certains membres possèdent aussi des compétences avancées dans ces langages, ce qui permet d'éviter les blocages lors de la programmation du site.

### Framework

Les frameworks que nous comptons utiliser sont les suivants :

- Bootstrap (Framework CSS)
- Symfony 5 (Framework PHP)
- JQuery (Framework Javascript)

Le choix du framework Bootstrap repose sur notre volonté de produire un site **responsive**. En ce qui concerne le framework Symfony, l'utiliser est une occasion pour les membres d'apprendre à utiliser ce framework ou de mettre à profit les connaissances déjà acquises. De plus, ce framework permet de faciliter la sécurisation du site, ainsi que le dialogue avec une base de données ou encore la création de pages web. Enfin, l'utilisation de JQuery permettra d'ajouter du dynamisme et de la fluidité à notre site.

### Symfony

#### *Qu'est ce que Symfony ?*

Symfony est un framework PHP qui permet de réaliser des sites complexes rapidement. Ce framework regroupe nombre de composants permettant de simplifier l'architecture d'un site et la lisibilité de son code.

Symfony est pour l'heure un des frameworks les plus utilisés dans le monde, notamment dans les entreprises. Il est d'ailleurs open-source. Étant très utilisé et donc relativement connu des développeurs, c'est un framework bien documenté et maintenu. L'assurance d'un framework populaire, d'une bonne documentation et d'une communauté active sont autant

d'arguments qui nous ont poussé à choisir ce framework, et ce, dans sa dernière version : Symfony 5.

De plus, Symfony embarque une technologie pratique pour la création de page HTML : **Twig**. Twig est un moteur de template pour le langage PHP. Utilisé par défaut dans Symfony, il permet une séparation entre les traitements faits en PHP et l'affichage que l'on veut avoir dans une page HTML. Cela permet donc une abstraction du langage PHP, mais aussi une standardisation moins coûteuse en temps en ce qui concerne l'apparence des pages HTML d'un site.

Symfony embarque aussi l'ORM Doctrine, qui permet une abstraction de la base de données pour le langage PHP.

Symfony permet, avec l'aide de Twig, de respecter un schéma Modèle-Vue-Contrôleur (MVC).

### *Installation et mise en place de Symfony*

Sous Windows, l'installation du framework est simple.

1. Récupération et exécution du `setup.exe` de Symfony (disponible sur [le site de Symfony](#))
2. Récupération et exécution du `ComposerSetup.exe` de Composer (disponible sur [le site de Composer](#))

Une fois ces deux éléments installés et mis à jour, on peut créer un nouveau projet via la console de Windows en utilisant la commande :

```
composer create-project symfony/web-skeleton projet_foodrating
```

Le framework Symfony embarque aussi son propre serveur, pouvant être activé via une ligne de commande : `symfony server:start` et pouvant être désactivé avec la commande : `symfony server:stop`. On retrouve alors la page d'accueil de notre projet à l'adresse `localhost:8000`.

### **Bootstrap**

Parmi ce que nous propose le framework Bootstrap, nous utiliserons notamment les éléments suivants :

- **Modal** : Une boîte de dialogue qui sera utilisée pour notre formulaire de connexion et d'inscription, les messages de confirmation ainsi que la rédaction d'une réponse sur le forum.
- **Carousel** : Un diaporama qui servira à afficher des produits populaires ou un échantillon d'éléments appartenant à la même catégorie d'aliments

- **Pagination** : La pagination servira à afficher un nombre fixe de produits ou de forums lorsque ceux-ci seront recherchés dans la barre de recherche.
- **Form** : Cet élément de Bootstrap sera utilisé pour tout ce qui nécessite un transfert de données entre l'utilisateur et le site (formulaire de connexion, barre de recherche, ...).
- **Toast** : Cet élément servira à informer l'utilisateur de manière ponctuelle.
- **Media Objects** : Cet élément servira pour l'affichage des réponses dans les forums et sera également utilisé pour les messages privés entre les utilisateurs.
- **Card** : Cet élément servira pour l'affichage des notes et des commentaires des produits.
- **NavBar** : Cet élément sera utilisé pour la barre de menu du site

## Serveur

Côté serveur, nous nous baserons sur l'outil XAMPP (ou WAMP) et sur le serveur intégré de Symfony. Cela nous permet d'avoir accès à PhpMyAdmin pour gérer notre base de données.

## Outils organisationnels

Les outils que nous utiliserons pour organiser notre projet sont les suivants :

- Git (Gestionnaire de version)
- Wireframe (Site web de création de maquette)
- GanttProject (pour créer un modèle Gantt)
- Discord (outils de communication et d'échange)

## Liste des fonctionnalités

Dans cette section, les principales fonctionnalités du site, obligatoires et optionnelles, seront répertoriées.

### Fonctionnalités obligatoires

#### Fonctionnalités basiques

- Afficher la page d'un produit
- Catégorisation des produits
- Création / Suppression de compte
- Déconnexion de compte
- Modification des informations du compte
- Notation d'un produit
- Modification d'une note attribuée à un produit
- Voir tous les votes pour un produit
- Création d'un compte administrateur

## Commentaire et forum

- Commenter un produit
- Modification / Suppression de commentaire
- Création d'un sujet dans la section "Forum"
- Création / Modification / Suppression de message dans un sujet
- Citation de message
- Voir tous les commentaires pour un produit

## Autres fonctionnalités

- Formulaire pour contacter l'administrateur
- Statistiques sur les produits et les notes des produits
- Agrandissement de l'image au survol
- Recherche d'un produit
- Tri sur les produits recherchés
- *Responsivité*
- Signaler un utilisateur
- Newsletter
- Affichage du tableau de bord (recueil d'information concernant l'utilisateur)
- Confirmation d'inscription par mail
- fonctionnalités uniquement pour l'administrateur (bannissement, suppression de message etc ...)

## Fonctionnalités optionnelles

Ici sont répertoriés les fonctionnalités qui ne sont pas impératives pour le fonctionnement du site mais qui seraient source de valeur ajoutée pour celui-ci.

- Répertoire d'utilisateurs "ami"
- Envoi de demande d'ami à un autre utilisateur
- Envoi de messages privés à un autre utilisateur
- Ajout de produit dans la base par un utilisateur
- Création d'un "Compte producteur"

*La fonctionnalité "Compte producteur" serait la création d'un compte spécial pour une firme (par exemple Nestlé) qui pourrait afficher toutes les statistiques du site relative à leur produit, dans différentes catégories de produits.*

## Organisation du projet

Dans le cadre de la réalisation de notre projet, il est nécessaire que nous nous organisions. Outre nos outils organisationnels, qui sont plus portés vers l'aspect programmation du projet, nous allons définir un **volume horaire de tâche** par membre composant l'équipe. De plus, nous ferons aussi un diagramme de Gantt pour apporter un support visuel à l'organisation de nos tâches.



## Volume horaire

Le volume horaire pour ce projet est estimé à environ 150 heures de travail pour chaque membre de groupe.

Nous avons estimé à 30 heures par personne le temps qu'il faudrait pour développer les fonctionnalités de la section "Fonctionnalités basiques" (voir la section [ici](#)), 30 heures par personne pour les fonctionnalités de la section "Commentaire et forum" (voir la section [ici](#)), 50 heures par personne pour les fonctionnalités de la section "Autres fonctionnalités" (voir la section [ici](#)), et 20 heures par personne pour les fonctionnalités de la section "Fonctionnalités optionnelles" (voir la section [ici](#)). Ces volumes comprennent aussi les heures dédiées au débbug et au test des fonctionnalités.

## Diagramme de Gantt

Les pages suivantes sont l'impression de notre diagramme de Gantt.

# Tâches

## Nom

---

Création - Maquettes  
Recherche - API pour les produits  
Création - Logo et nom  
Documentation - Présentation de projet  
Documentation - Liste des fonctionnalités  
Documentation - Fiche collaborative  
Planning - Création  
Apprentissage - Symfony  
Création - Repository Github  
Apprentissage - Git / Github  
Apprentissage - Bootstrap  
Création - Page d'accueil  
Documentation - Base de données  
Documentation - Analyse de projet  
Planning - Mise à jour  
Gestion des comptes utilisateurs  
Création de la base de données avec l'API  
Affichage des produits  
Amélioration du formulaire de connexion  
Pagination  
Ajout des tables Notes et Commentaires  
Import CSV  
Notation  
Recherche de produit  
Intégration du wrapper  
Forum  
Recherche V2  
Catégorisation  
Commentaires

## Tâches

3

### Nom

---

Produits similaire

Mail et serveur SMTP

Débug

Page d'accueil

Modification du template "Produit"

Espace admin

Photo de profil

Correction

Tableau de bord

Demande d'ami

Delete Cascade Utilisateur

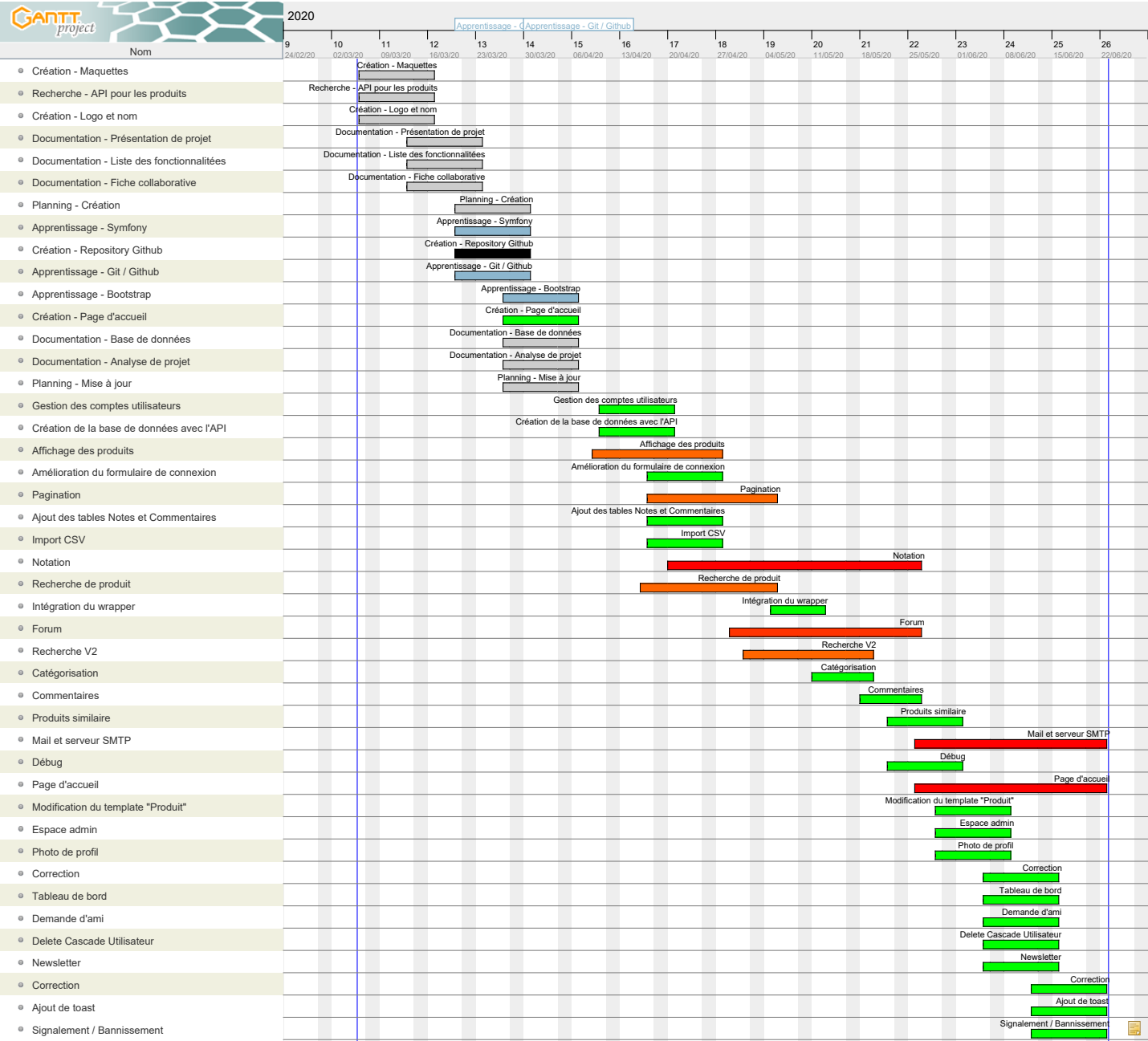
Newsletter

Correction

Ajout de toast

Signalement / Bannissement

Diagramme de Gantt



## Analyse du projet

Dans cette section du dossier, il sera montré comment le site sera structuré. En effet, le site comporte deux grandes parties, qui seront implémentées l'une après l'autre, la première étant la partie "Produit et notation" et la seconde étant la partie "Forum et commentaires".

Cette section abordera aussi l'agencement de la base de données du site.

*Il est à noter que cette analyse étant antérieur à la phase de programmation, le résultat du projet peut légèrement différé de ce qui était prévu.*

## Produit et notation

Cette partie du site est son essence. Elle prend donc une part importante dès la page d'accueil, où il faudra y avoir la possibilité de voir certains produits. La page d'accueil montrera un carrousel contenant des images choisies au hasard parmi nos produits. De plus, il y figurera une liste des cinq produits les mieux notés du site.

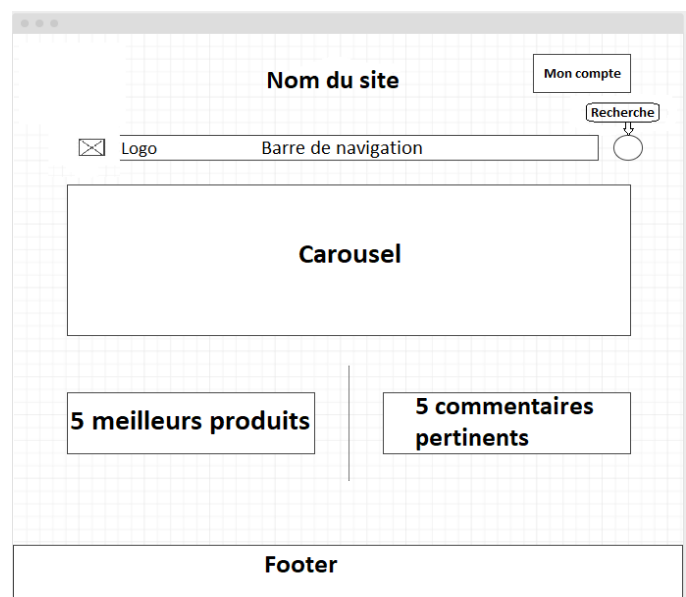


Figure 1 : Maquette de la page d'accueil

Le bouton "Mon Compte" qui figure sur la page fera apparaître un double formulaire, comme nous pouvons le voir dans la figure 2.

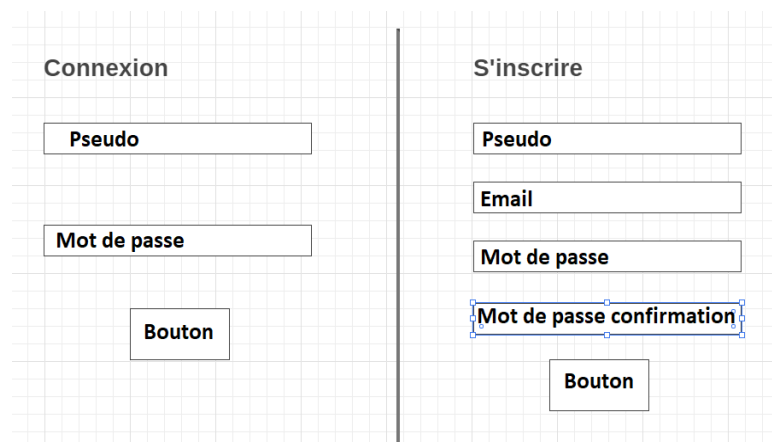
Maquette des formulaires de connexion et d'inscription. Le design est basé sur une grille. À gauche, sous le titre "Connexion", il y a un champ "Pseudo", un champ "Mot de passe" et un bouton "Bouton". À droite, sous le titre "S'inscrire", il y a un champ "Pseudo", un champ "Email", un champ "Mot de passe", un champ "Mot de passe confirmation" (surligné avec une bordure bleue) et un bouton "Bouton".

Figure 2 : Maquette des formulaires

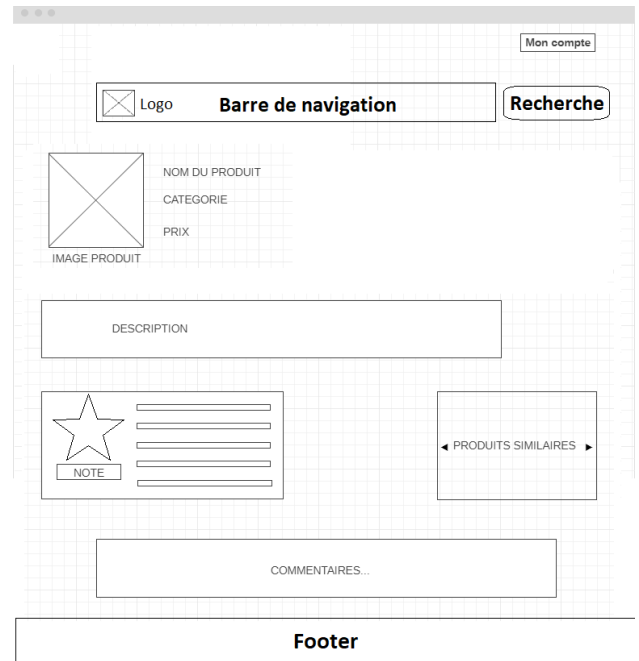
Une fois l'utilisateur connecté, son icône d'utilisateur remplacera le bouton "Mon compte" tel qu'il apparaît dans la figure 3.



Figure 3 : Maquette de l'icône d'utilisateur

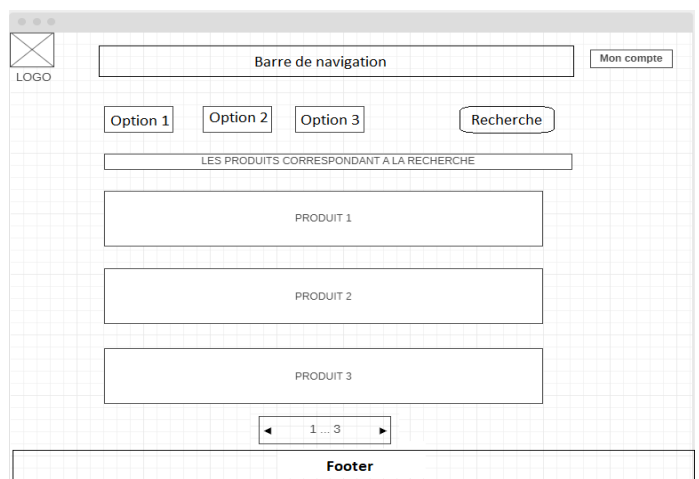
L'utilisateur pourra également voir les produits du site. Il pourra consulter diverses informations concernant le produit, notamment son nom, sa catégorie, son Nutri-Score et éventuellement une description de ce qu'il contient.

Figure 4 : Maquette de la page d'un produit



Lorsque l'utilisateur recherchera un produit, la page affichant le résultat de la recherche comportera une liste paginée des éléments correspondant à la recherche. De plus, différentes options seront disponibles concernant la recherche, comme différentes manières de trier le résultat par exemple.

Figure 5 : Maquette d'un résultat de recherche



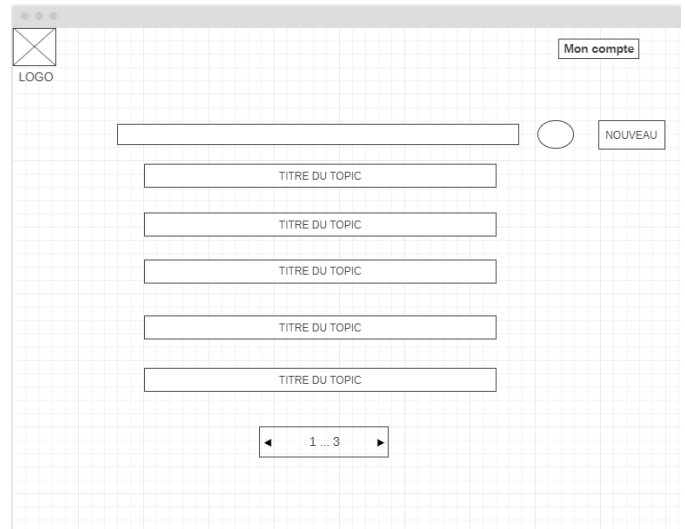
## Forum et commentaires

Cette partie du site a pour objectif l'interaction entre utilisateurs. L'objectif est de donner toutes les fonctionnalités nécessaires à l'utilisateur afin qu'il puisse échanger avec les autres utilisateurs. Cependant, seuls les utilisateurs connectés pourront créer des sujets dans la section "Forum" ou commenter un produit. De plus, un utilisateur ne pourra pas commenter un produit sans lui donner une note.

### Forum

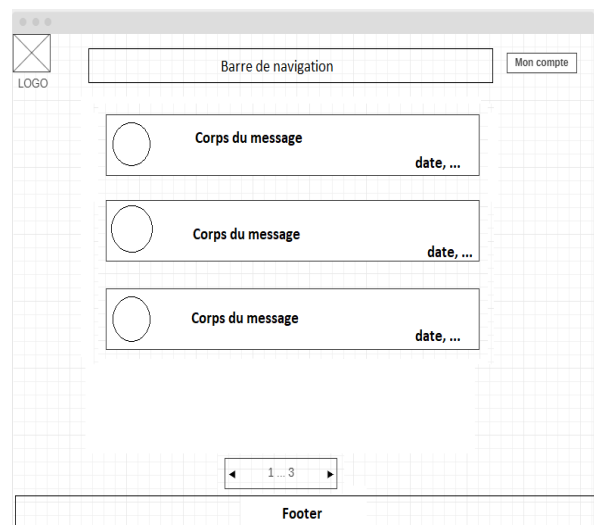
Pour la section "Forum", un aspect très simple a été choisi. L'utilisateur connecté ou non, pourra voir une liste de tous les sujets du forum du site. En cliquant sur le nom du sujet, l'utilisateur pourra avoir accès à la conversation et pourra envoyer une réponse, s'il est connecté (figure 6).

Figure 6 : Maquette de l'accueil du forum



Dans un sujet, les réponses des utilisateurs sont affichées les unes à la suite des autres, et comporte le pseudo de l'utilisateur ainsi que son image, le texte de sa réponse et la date d'envoi (figure 7).

Figure 7 : Maquette d'un sujet de discussion



## Commentaires

Les commentaires seront visibles sous les produits, et accompagnés de la note qu'aura donnée l'utilisateur. Ils comporteront aussi le nom de l'utilisateur et le nombre d'utilisateurs qui ont trouvé ce commentaire pertinent (nécessaire pour afficher les cinq commentaires les plus pertinents dans la page d'accueil). Leur affichage se fera avec les "Media Objects" de Bootstrap.

## Base de données

Pour stocker tous les éléments du site, notre base de données contiendra plusieurs tables :

- **Utilisateurs** : Les données sur les utilisateurs
- **Commentaires** : Les commentaires des utilisateurs sur les produits
- **Notes** : Les notes des utilisateurs sur les produits
- **Catégories** : Les catégories des produits
- **Discussions** : Les sujets du forum
- **Réponses** : Les réponses aux sujets du forum
- **Produit** : Les produits de l'API

Les produits que l'on stockera dans notre base seront les produits existant dans l'API Open Food Facts, mais aussi des produits que les utilisateurs voudront ajouter.

## Programmation du site

Cette section a pour objectif de montrer les différentes évolutions qu'aura connues notre projet au cours de son développement. Il comportera aussi les différents problèmes rencontrés au cours du développement et les solutions apportées.

### Début, mise en place de la page d'accueil

#### Page d'accueil

Comme montré dans notre maquette, notre page doit montrer un carrousel de produits, les produits les mieux notés sur le site, et les commentaires que les utilisateurs avaient le plus aimés.

Avec nos connaissances acquises en auto-formation sur Bootstrap, nous avons commencé à mettre en place la page d'accueil. Nous nous sommes mis d'accord sur un thème de couleur pour notre site (thème disponible sur le site [Bootswatch](#)) et Matthieu a pu mettre en place la page d'accueil conformément à la maquette en utilisant un template Twig.

Pour éviter de multiplier le code inutile, Twig embarque une fonctionnalité pratique : **l'héritage**. Un template peut hériter d'un autre template. C'est de cette manière qu'il faut procéder pour que le thème choisi se répercute sur toutes les pages de notre site. Concrètement, cela veut dire que tous les éléments que l'on veut voir sur toutes les pages du site devront figurer dans un template qui sera la base de tous les autres templates (comme la barre de navigation et le pied de page).



Notre template de base ressemble donc à cela :

```
<html>
  <head>
    <title> {% block title %} FoodRating {% endblock %} </title>
    {% Feuille de style et script JS %}
    {% block stylesheets %}{% endblock %}
  </head>
  <body>
    {% Barre de navigation %}
    <div> {% block body %}{% endblock %} </div>
    {% Pied de page %}
  </body>
</html>
```

Pour le moment, aucun traitement spécifique n'est fait côté PHP, à part l'appel du template de base (base.html.twig).

### Tâches annexes

Dans le même temps, Thomas L. et Martin se sont occupé de mettre un repository sur Github et de faire un logo pour notre projet.

### Inscription et connexion de l'utilisateur

Après avoir mis en place la première page de notre site, les premières fonctionnalités à ajouter sont l'inscription et la connexion des utilisateurs. En effet, plusieurs des fonctionnalités prévues pour notre site repose sur l'existence ou non d'un compte d'utilisateur. Il nous a donc semblé pertinent de commencer par ce premier aspect de la gestion des utilisateurs. C'est Matthieu qui a été chargé de faire cette fonctionnalité.

#### Inscription

La première étape est la création d'un formulaire d'inscription. En accord avec notre architecture, nous avons besoin de trois éléments :

- Un **contrôleur** qui contiendra les fonctions PHP nécessaire à l'inscription
- Un template Twig qui sera la page HTML contenant le formulaire, donc notre **vue**
- Un objet Utilisateur (une *entité* dans Symfony) qui sera notre **modèle**

Avec le bundle **MakerBundle** intégré dans Symfony, on peut créer des contrôleurs et des entités en faisant les commandes suivantes dans un terminal :

```
php bin/console make:controller
php bin/console make:entity
```

Ainsi, quand l'utilisateur rentre ses informations dans le formulaire, un objet Utilisateur est créé par le contrôleur et enregistré dans notre base de données. De plus, Symfony possède des fonctions qui nous ont permis de crypter les mots de passe des utilisateurs.

### Confirmation par email

Lors de l'inscription, un mail est envoyé à l'adresse mail de l'utilisateur pour pouvoir vérifier que l'adresse mail de l'utilisateur soit correcte et fonctionnelle, il permet aussi de pouvoir éviter de créer des utilisateurs en boucle et ainsi limiter le nombre de faux utilisateurs et donc de faux avis et notes. Cela permet aussi de pouvoir être sûr que d'autres fonctionnalités puissent fonctionner correctement.

### Connexion

Le principe pour la connexion est le même que pour l'inscription :

- Les fonctions PHP nécessaires à la connexion seront dans le **contrôleur** contenant les fonctions pour l'inscription
- La connexion se base aussi sur l'entité Utilisateur

Il faut créer un nouveau template pour le formulaire de connexion. Une fois le formulaire créé et les fonctions achevées, il reste encore un élément à modifier : les paramètres de sécurité de Symfony.

En effet, Symfony gère lui-même les sessions des utilisateurs, et il a besoin de connaître certaines informations, notamment :

- L'entité correspondant aux utilisateurs (ici, c'est Utilisateur)
- L'algorithme de cryptage
- Le chemin vers la fonction PHP gérant au formulaire de connexion
- Le chemin vers la fonction PHP gérant la déconnexion

■ Ces informations sont rentrées dans un fichier appelé `security.yaml`.

### Mot de passe oublié

Sur la page de connexion, un lien hypertexte est affiché pour permettre à l'utilisateur de réinitialiser son mot de passe. C'est d'ailleurs l'une des raisons pour lesquelles nous vérifions l'adresse mail de l'utilisateur à son inscription, car s'il donne une fausse adresse, il ne pourra pas réinitialiser son mot de passe.

### Problème rencontré : Cohabitation des deux formulaires

Dans notre maquette pour les formulaires d'inscription et de connexion, il était question de mettre les formulaires côte à côte dans une page HTML. Cependant, le faire causait des conflits entre les formulaires et les fonctions PHP derrière ces formulaires.

Thomas L. et Matthieu ont solutionné le problème en mettant les deux formulaires dans des pages HTML distinctes. Pour conserver notre idée de pouvoir accéder à ces formulaires sans recharger la page, ils ont pensé à faire un script JQuery faisant une requête AJAX. Ainsi, le passage d'un formulaire à un autre se fait sans rechargement de la page.

## Modification des informations de l'utilisateur

Après l'ajout des fonctionnalités d'inscription et de connexion, il a fallu donner la possibilité à l'utilisateur de voir ses informations de connexion mais aussi de pouvoir les modifier.

Il a donc fallu créer un formulaire de modification. Ce formulaire contient, dans des champs pré-remplis, les informations de l'utilisateur. Nous avons aussi décidé de laisser à l'utilisateur la possibilité de supprimer son compte. Quand un utilisateur décide de supprimer son compte, toutes ses informations sont supprimées de notre base de données. Une fois la suppression terminée, l'utilisateur est redirigé vers la page d'accueil avec une nouvelle session anonyme.

### Problème rencontré : Modification du mot de passe

La modification du mot de passe dans Symfony n'était pas aisée. En effet, la mettre dans le formulaire pré-rempli créait des conflits dans le formulaire. Il a donc fallu faire un formulaire à part pour la modification du mot de passe.

## Affichage des produits sur le site

L'affichage d'un produit sur le site est une fonctionnalité importante de notre projet. Ce sont Martin et Thomas M. qui ont implémenté cette fonctionnalité.

Pour cette fonctionnalité, nous avons eu besoin de :

- Créer un template pour la page d'un produit
- Créer des fonctions dans un contrôleur pour gérer la récupération de produits en base
- Créer les entités Produit, Note, Commentaire

### Stockage des produits en base

Pour afficher un produit de l'API, il a été décidé de stocker des produits dans notre base de données. Pour ce faire, nous avons pris un export au format CSV des produits de l'API. Ce CSV est très volumineux (environ 1 million de lignes, soit environ un fichier de 2 Go), rendant l'importation impossible.

Nous avons cherché des solutions, comme augmenter la capacité de PHP et de PhpMyAdmin afin de pouvoir importer des fichiers de grande taille par exemple. Cependant, cela n'a pas fonctionné. Ensuite, Matthieu a créé un programme PHP permettant d'ouvrir le CSV et d'importer ligne par ligne les données. Le bon fonctionnement de cette méthode a nécessité la réduction de la taille du fichier.

Nous avons donc eu recours à un logiciel, afin de créer des échantillons de 100 000 produits pour faire nos tests, et ainsi réaliser un import plutôt fonctionnel, bien que long.

Ces produits ont ensuite été rendus accessibles sous forme de liste pour que nous puissions tester la création de liens vers les pages de ces produits. En effet, l'url d'un produit a la forme `/produit/{categorie}/{nom}`. Ainsi, chaque produit dans la liste possède un lien affichant la page d'un produit.

Les *routes* (l'url menant à une fonction) dans Symfony sont gérées via des annotations dans les contrôleurs, tel que :

```
/**
 * @Route("/ma/route", name="ma_route")
 */
public function maFonction() {}
```

Quand un lien est fait dans une page et qu'il correspond au format d'une route, Symfony fait appel à la fonction correspondant à la route. On peut aussi rediriger un traitement vers une fonction précise en utilisant la fonction `path()` de Twig et le nom d'une route.

Une route comportant des éléments entre accolades est une route possédant des éléments variables, qui peuvent être passés en paramètre de la fonction `path()`.

Avec l'ajout de l'affichage des produits, notre base de données ressemble au schéma suivant:

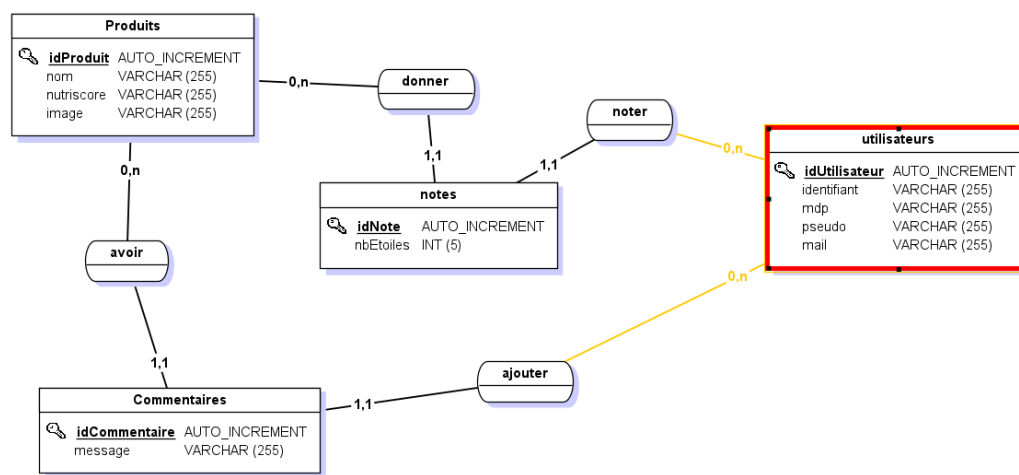


Figure 8 : Base de données V1

### Problème rencontré : Performance

Après avoir effectué nos tests, nous nous sommes rendus compte d'un problème majeur : les performances de notre site. En effet, plus de 100 000 produits dans notre base de données augmentaient de beaucoup les temps de chargement de notre site, que ce soit pour la récupération de la liste des produits ou l'affichage d'un produit, ce qui rendait les tests difficile à effectuer et nous a obligé à réduire drastiquement le nombre de produits en base.

La solution à ce problème n'a pas été trouvée de suite. Il a fallu continuer d'avancer sur le projet en attendant d'avoir cette solution.

## Ajout de la fonctionnalité de recherche d'un produit

Pour faciliter l'accès à un produit, nous avons décidé d'ajouter une barre de recherche sur notre site. Cette barre de recherche permet à l'utilisateur de rechercher un produit dans la base de données en se basant sur son nom.

De plus, nous utilisons l'*autocomplete* de JQuery pour proposer à l'utilisateur une auto-complétion dans la barre de recherche. Lorsque l'utilisateur clique sur le bouton "Rechercher", il est redirigé vers une page contenant les produits correspondant à la recherche.

## La pagination avec KNP Paginator

Au cours du développement du projet, nous nous sommes aperçus que nous aurions besoin d'afficher un grand nombre de produits. Il nous fallait un moyen de **paginer** les produits que nous affichions dans notre liste. C'est Martin qui s'est occupé d'ajouter cette fonctionnalité.

### Ajout et utilisation du bundle

KNP Paginator est un bundle que l'on peut ajouter à un projet Symfony. Pour cela, il suffit de saisir la commande suivante dans un terminal :

```
composer require knplabs/knp-paginator-bundle
```

Une fois le bundle ajouté au projet, on peut utiliser la fonction `paginate()`, qui permet de paginer les éléments d'un tableau PHP, et de donner un nombre maximum d'éléments par page. Pour naviguer à travers les pages, nous utilisons l'objet `Request` dans le contrôleur, qui permet de manipuler les paramètres dans l'url, comme le paramètre `page`, tel que : `/ma/route?page=1`. De plus, dans le template affichant la liste des produits, nous ajoutons une ligne de code qui permet l'affichage du sélecteur de page et qui est la suivante :

```
{{ knp_pagination_render(produits) }}
```

### Problème rencontré : Intégration de Bootstrap

La pagination est simple à mettre en place. Néanmoins, nous avons eu du mal à faire correspondre l'apparence du sélecteur de page avec notre thème Bootstrap. Une première solution proposée par Martin a été de refaire entièrement le sélecteur en calculant les pages et en intégrant les classes Bootstrap. Mais cela n'a pas réglé le problème, au contraire.

<	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53		
54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79		
80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104			
105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125							
126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146							
147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167							
168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188							
189	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209							
210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230							
231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251							
252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272							
273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	291	292	293							
294	295	296	297	298	299	300	301	302	303	304	305	306	307	308	309	310	311	312	313	314							

Figure 9 : Un peu trop de boutons...

Après quelques recherches, il a proposé une solution plus simple. En effet, il s'avère que le bundle possède plusieurs templates qu'il est possible de changer, dont un qui intègre Bootstrap. Il a donc pu solutionner le problème en ajoutant les lignes suivantes dans le contrôleur :

```
// "$p" est le tableau de produit paginé
$p->setTemplate('@KnpPaginator/Pagination/
    twitter_bootstrap_v4_pagination.html.twig');

$p->setCustomParameters(["align" => "center"]);
```

## Système de notation

La notation est une fonctionnalité importante dans notre site. Nous avons implémenté cette fonctionnalité avant d'implémenter les commentaires. Ce sont Matthieu et Thomas L. qui ont ajouté cette fonctionnalité au projet.

### Ajout des étoiles : Icône ☆ et JQuery

Pour noter les produits, nous avons opté pour un système en cinq étoiles. Pour afficher ces étoiles, nous avons utilisé les icônes de la bibliothèque **Font Awesome 5**. Les étoiles sont grises lorsqu'aucune note n'a été donnée. Cependant, certaines d'entre elles (ou toutes) deviennent jaune lorsque l'on clique dessus, à hauteur de la note donnée. Ainsi, cliquer sur la quatrième étoile correspond à une note de 4 sur 5 et fait passer quatre étoiles de la couleur grise à la couleur jaune.

Ce changement de couleur est géré par un script JQuery. De plus, les étoiles se situent dans un formulaire. Par conséquent, le clic sur le bouton "Noter" situé en dessous enregistre dans la base de données le nombre d'étoiles donné et le produit noté.

### Modification et suppression d'une note, moyenne des notes

Quand un utilisateur est connecté et qu'il a donné une note à un produit, l'utilisateur peut revenir sur le produit et sa note sera rechargée. Il pourra alors la modifier à loisir ou la supprimer en cliquant sur le bouton "Supprimer".

Quand l'utilisateur met ou modifie une note, la moyenne des notes du produit est mise à jour.

### Affichage des produits dans le site : Révision

Nous avons dû revenir sur le problème de l'affichage des produits dans notre site. En effet, notre solution de les stocker en base était limitée car elle réduisait beaucoup les performances du site. Il a donc fallu trouver un autre moyen d'afficher les produits de notre site. Tous les membres du groupe ont dû s'atteler à la tâche pour résoudre ce problème.

### Utiliser du JSON ?

La première solution que l'on a essayée de mettre en place était l'utilisation d'**appel d'API**. Ces appels d'API auraient permis d'exploiter des objets JSON contenant les informations d'un ou plusieurs produits. Cependant, le manque de documentation et le nombre important d'informations contenu dans un objet JSON de l'API nous ont dissuadés d'utiliser ce moyen.

### Wrapper PHP

Un **wrapper** (ou *adaptateur* en français) permet, dans le cas de notre API, de récupérer les produits via des fonctions spécifiques, et donc de les rendre exploitables par le langage de programmation utilisé (dans notre cas, c'est le langage PHP).

On a pu ajouter le wrapper à notre projet via la commande suivante :

```
composer require openfoodfacts/openfoodfacts-php
```

Après quelques recherches, nous avons trouvé une solution envisageable : un **wrapper**. Il se trouve que l'API en proposait pour certains langages de programmation et nous avons pris celui proposé pour le langage PHP. Le choix de prendre un wrapper nous a fait réviser notre schéma de base de données. En effet, le wrapper permet de se passer de l'entité **Produit** que nous avons créé et donc de la table **Produit** dans notre base. Ce choix nous a aussi obligé à changer plusieurs sections de notre projet, comme la notation, les fonctions se basant sur l'entité **Produit** ou encore la recherche de produit.

Bien que pratique d'utilisation, le wrapper de l'API manquait de documentation, ce qui nous a obligé, pendant le reste du développement du projet, à revoir le code utilisé pour pallier aux erreurs qui apparaissaient ponctuellement.

Exemple d'utilisation du wrapper de l'API :

```
$api = new Api("food", "fr");

// On peut récupérer un produit dans l'API via son ID
$produit = $api->getProduct("3057640385148")
```

## Ajout des fonctionnalités de la partie "Forum"

Le forum représente la seconde moitié de notre projet, la première étant la notation et les commentaires de produit. Ce sont Martin et Thomas L. qui ont implémenté ces fonctionnalités.

Le forum a nécessité la création de plusieurs éléments :

- Des templates pour afficher la liste des sujets de discussion, le fil des réponses aux sujets et le formulaire de création de sujet
- Un contrôleur pour gérer les fonctions PHP relatives au forum
- Les entités Discussion et Reponse

### Discussion et message

Pour permettre aux utilisateurs de discuter, nous avons ajouté un formulaire pour la création de sujet de discussion. Un sujet de discussion contient un titre, un sujet, et le message du créateur du sujet, auquel d'autres utilisateurs peuvent répondre.

En ce qui concerne les réponses, leur affichage est basé sur la classe "Media Objects" de Bootstrap et comporte le nom de l'utilisateur, sa photo de profil (s'il en possède une), son message et l'heure à laquelle il a été posté.

Après l'ajout de ces fonctionnalités, notre base de données ressemble au schéma suivant :

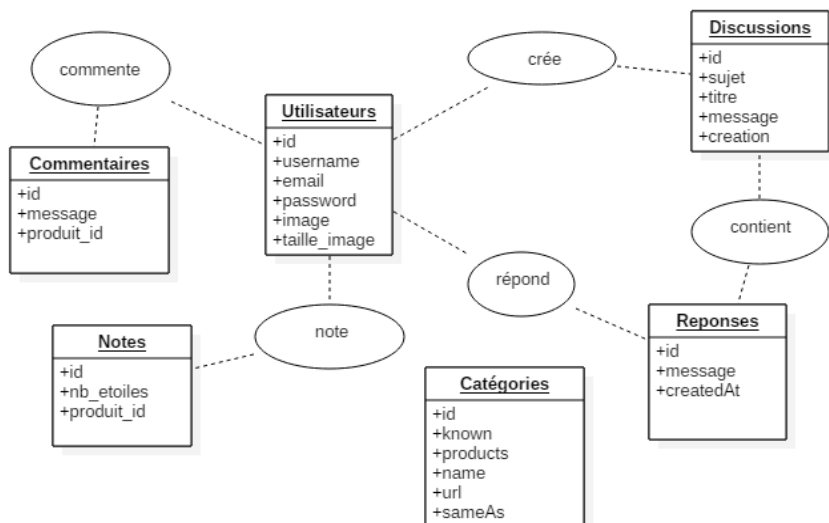


Figure 10 : Base de données V2



## Recherche de produit : Révision

Après avoir ajouté le wrapper PHP, il a fallu revoir la fonctionnalité de recherche de notre site. Ce sont Martin, Matthieu et Thomas M. qui ont travaillé sur la révision de cette fonctionnalité.

Jusqu'ici, la fonction de recherche était une requête sur la base de données. Cependant, avec le wrapper de l'API, on a accès à la fonction `search()`, qui permet de rechercher tous les produits contenant dans leurs mots-clés le mot passé en paramètre de la fonction.

## Problème rencontré : Limite de pagination

La recherche nous a posé problème car le résultat de la fonction `search()` était déjà paginé et notre pagination entraînait en conflit avec. C'est en comparant les résultats de notre site avec le site de l'API que l'on s'est aperçu de cela. Pour résoudre ce problème, il a fallu paginer en prenant en compte le nombre de résultats retourné par la fonction et le nombre de produits par page retourné par l'API.

## Ajout d'une liste des catégories

Pour faciliter la recherche de produit, nous avons décidé d'implémenter une page qui recense les catégories de l'API. Cette page contient une liste paginée de liens menant à une liste des produits appartenant à cette catégorie. C'est Martin qui a implémenté cette fonctionnalité.

## Système de commentaire

Après que le système de notation ait été achevé, il a fallu ajouter les commentaires. C'est Matthieu et Martin qui ont implémenté cette fonctionnalité.

Le système de commentaire fait qu'un commentaire est toujours relié à une note. La gestion des commentaires repose sur les mêmes principes que pour une note. On peut ajouter un commentaire, ou simplement le supprimer en vidant la zone de texte.

Pour l'affichage des commentaires, nous avons opté pour la classe "Card" de Bootstrap. Cela nous permet de montrer plusieurs notes et commentaires de manière simple et condensée.

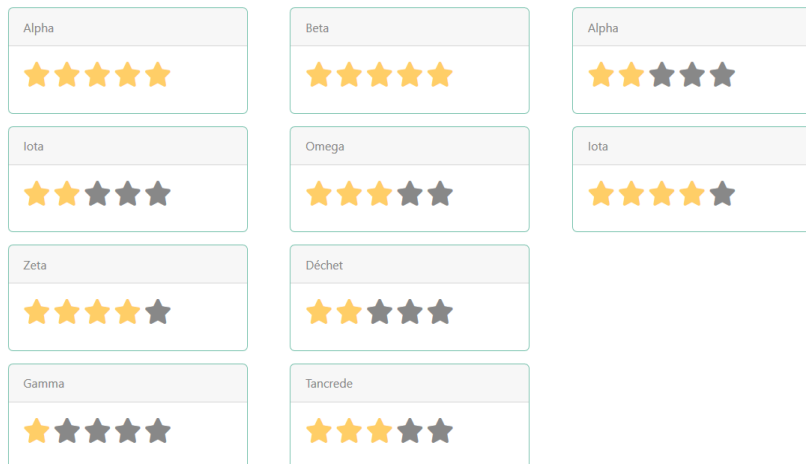


Figure 11: Sans commentaire

Une autre fonctionnalité que nous avons voulu implémenter est le "like" d'un commentaire. Ainsi, un utilisateur peut "liker" un commentaire, ce qui signifie son utilité et peut lui permettre d'apparaître dans la section "Meilleurs commentaires" de la page d'accueil.

### **Problème rencontré : Positionnement des cartes**

Un problème d'affichage s'est posé lorsque l'on a commencé à avoir plusieurs commentaires. En effet, les cartes dépassaient du conteneur et s'alignaient étrangement. Nous avons résolu ce problème en utilisant une classe de Bootstrap appelée "card-columns".

### **Carrousel de la page d'accueil**

Sur la page d'accueil, plusieurs carrousels sont censés apparaître. Ces carrousels se basent sur la classe "Carousel" de Bootstrap. Ce sont Martin et Matthieu qui ont implémenté ces fonctionnalités.

Le premier carrousel contient des produits tirés au hasard. Chaque produit affiché montre son image, ainsi que sa catégorie et son nom. De plus, le nom du produit et l'image sont des liens vers la page du produit.

Le second carrousel, "Les meilleurs produits", affiche les produits les mieux notés sur notre site. La présentation pour ce carrousel est la même que celle du premier carrousel.

Le troisième carrousel, "Les meilleurs commentaires", affiche les commentaires les plus "likés" de notre site.

### **Problème rencontré : La moyenne irrécupérable**

Au départ, la moyenne était un simple calcul qui se faisait à l'affichage d'une page de produit. Cependant, refaire ce calcul dans la page d'accueil s'avérait être une tâche trop complexe. Nous avons opté pour une solution plus simple : créer une table pour les moyennes des notes des produits.

Une fois cela fait, on a pu afficher les produits avec les meilleurs moyennes.

### **Ajout de la photo de profil**

Nous avons donné la possibilité à l'utilisateur de pouvoir ajouter une photo de profil. Pour cela, il était question d'enregistrer directement l'image dans la base de données (type blob). Ensuite il fallait la récupérer dans le contrôleur afin de pouvoir l'afficher avec Twig. Pour l'afficher correctement en HTML, il fallait décoder l'image avec une fonction PHP. Cependant, cette manipulation ne fonctionnait pas sur toutes les pages. Matthieu a donc eu l'idée d'enregistrer directement l'encodage de l'image créé par la fonction PHP dans la base de données, ce qui permet d'éviter la conversion sur toutes les pages.

Martin s'est ensuite occupé de la mise en forme de la fonctionnalité.

## Fonctionnalité des produits similaires

Dans la page des produits, à côté des notes, un carrousel montre les produits similaires au produit affiché. C'est Martin qui a implémenté cette fonctionnalité.

Ce carrousel, comme pour les carrousels de la page d'accueil, montre une image ainsi que le nom du produit, et sa catégorie. L'image et le nom du produit sont des liens vers le produit affiché dans le carrousel.

## Tableau de bord

Le tableau de bord est une fonctionnalité importante pour l'utilisateur, car il lui permet de voir les éléments avec lesquels il a interagi. C'est Matthieu qui a implémenté cette fonctionnalité.

### Partie "Produit"

Pour conserver les informations concernant les produits visités ou notés, le moyen utilisé est la création de fichier CSV. Chaque utilisateur connecté possède un dossier contenant les fichiers CSV sur les produits qu'il a visités. Il a fallu gérer le cas d'informations en double lorsque l'utilisateur visite plusieurs fois le même produit.

Le principe est le même pour les produits commentés.

### Partie "Forum"

Le principe est le même pour la partie "Forum" que pour la partie "Produit".

## Lien entre les utilisateurs

Cette fonctionnalité concerne les demandes d'amis. Pour bien faire fonctionner cette fonctionnalité, il a été nécessaire de créer deux nouvelles tables : une pour les demandes d'amis et une autre pour ceux qui sont effectivement amis.

Après l'ajout du tableau de bord, notre base de données correspondait au schéma suivant :

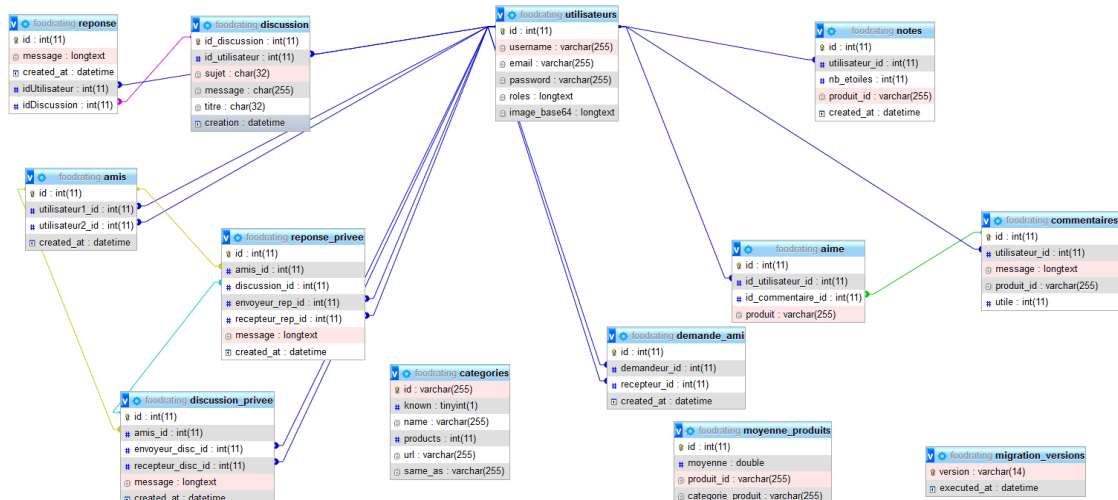


Figure 12 : Base de données V3

## Message entre les utilisateurs

Avec l'implémentation des demandes d'amis, les messages entre utilisateurs ont aussi été implémentés. Cela a nécessité la création de deux tables : une pour les discussions privées et une autre pour les réponses contenues dans ces discussions.

Quand un utilisateur supprime son compte ou retire un autre utilisateur de ses amis, les discussions concernés par l'un ou l'autre cas sont supprimées.

## Mail de notification pour la création d'un compte

Cette fonctionnalité existe sur la plupart des sites internet de nos jours. C'est Thomas L. qui a implémenté cette fonctionnalité.

Cette fonctionnalité, bien que constitutrice de beaucoup de sites internet, n'a pas été facile à implémenter. Il nous fallait déjà un serveur SMTP, élément que nous n'avions pas utilisé jusqu'à présent, et un objet permettant d'envoyer des mails. Si un bundle dans Symfony permet de gérer l'envoi de mail, le serveur SMTP n'a pas été trouvé de suite.

Pour pouvoir mettre en place cette fonctionnalité, il a fallu avoir recours à Node.js et à un serveur de messagerie en local (*maildev*). Une fois la configuration faite, on peut tester l'envoi de mail après la création d'un compte. Le mail va demander à l'utilisateur s'il a bien créé un compte sur notre site, et si oui, va le rediriger vers notre site en vérifiant son compte.

## Formulaire de contact

En plus du mail de notification, Thomas L. a aussi dû ajouter la fonctionnalité "Formulaire de contact", qui repose sur l'envoi de mail et donc l'utilisation d'un serveur SMTP.

## Fonctionnalités d'administration

Les fonctionnalités d'administration (et donc le rôle d'administrateur), sont des fonctionnalités importantes en développement Web. Il a donc fallu intégrer ce rôle et les droits qu'il possède dans notre projet. C'est Thomas M. qui a été chargé d'implémenter ces fonctionnalités.

Pour commencer, le plus important était de différencier les utilisateurs lambda d'un administrateur. Pour cela, il a d'abord fallu ajouter un champ rôle dans l'entité Utilisateur (et donc la même colonne dans la table Utilisateur). Ensuite, il a fallu gérer ce qui s'affichait pour l'administrateur, notamment dans son tableau de bord où il peut voir tous les comptes des utilisateurs et les bannir ou les supprimer de la base de données.

L'ajout de la fonctionnalité de bannissement a d'ailleurs nécessité l'ajout d'une table "Bannis" dans notre base de données. Cette table répertorie simplement les mails des utilisateurs qui ont été bannis du site, et uniquement leur mail. L'utilisateur ne pourra donc plus se connecter et s'inscrire avec cette adresse mail, même s'il met un autre pseudo.

## Petits ajouts

Pour que le site soit plus propre, nous avons ajouté des messages de confirmation quand l'utilisateur est sur le point de réaliser une action, et également des toasts permettant d'informer l'utilisateur sur une action.

Matthieu s'est occupé des messages de confirmations et Thomas M. des toasts. Matthieu a également créé la charte de confidentialité du site, disponible sur l'onglet mentions légales.

## Conclusion du projet : retards et dernières fonctionnalités

Ce projet a permis, à toute l'équipe, de progresser en développement Web et de découvrir de nouvelles technologies. Cependant, plusieurs problèmes récurrents sont apparus durant le développement de ce projet. De plus, le contexte actuel (COVID-19) n'a pas facilité le travail sur ce projet car la communication fut beaucoup plus difficile, causant des retards et nous forçant à mettre certaines fonctionnalités de côté.

## Problème récurrent : Redirection

Notre site utilise le wrapper de l'API Open Food Facts pour obtenir ses produits. Cependant, la plupart de nos fonctionnalités se base sur la recherche de l'API via les catégories du produit. Le problème ici est que le nom de la catégorie est nécessaire pour cette recherche, mais n'est pas facilement accessible. Pour obtenir cela, il a fallu importer les catégories de l'API dans notre base, puis prendre dans l'url de la catégorie la partie qui nous intéressait.

Exemple :

```
// La partie intéressante est la partie après "categorie/"  
https://fr.openfoodfacts.org/categorie/vinaigres-d-alcools
```

Cependant, les url de catégorie peuvent comporter des caractères particuliers comme des accents par exemple, ce qui empêche le bon fonctionnement de la redirection de l'API. Il a donc fallu créer une fonction comprenant plusieurs expressions régulières pour venir à bout de ce problème.

Ce problème est d'ailleurs revenu plusieurs fois au cours du développement du projet, nous obligeant à prendre en compte toujours plus de cas particuliers pour la redirection.

### **Problème récurrent : Hétérogénéité des données des produits**

Le problème de l'hétérogénéité des données des produits de l'API est ce qui nous a pris le plus de temps et ce qui a pas mal complexifié le projet. En effet, l'API est open-source, ce qui fait que l'ajout des produits est libre et non contrôlé. Cela a posé problème pour nous qui voulions récupérer et exploiter les produits car certains produits possédaient des informations que les autres n'avaient pas. De plus, au lieu de ne pas exister, certaines valeurs étaient vides (null), ce qui posait aussi problème.

Pour pallier à ce handicap, il a fallu mettre un grand nombre de conditions et de vérifications dans les fonctions, et ce, au fur et à mesure que l'on découvrait les différents cas d'erreurs. Ce problème est d'ailleurs la source des nombreux problèmes que l'on a rencontré durant le développement de ce projet.

### **Problème récurrent : Vérification de doublon dans un tableau multi-dimensionnel**

Ce problème a été rencontré lors de l'ajout de données dans un fichier CSV. En effet, pour éviter l'ajout de doublon, il fallait déterminer si la donnée existait dans un tableau à plusieurs dimensions en PHP, la fonction `in_array()` de PHP n'était pas suffisante. Il a donc fallu créer une fonction que l'on a appelé `multi_in_array()` et qui permet de vérifier si un élément est présent ou non dans un tableau à plusieurs dimensions.

### **Fonctionnalités inachevées**

Contrairement à ce que nous avons prévu au départ, nous n'avons pas pu développer toutes les fonctionnalités que nous avons prévues. Cela est dû à plusieurs facteurs notamment la difficulté d'implémenter certains éléments au projet (le wrapper par exemple) ou encore les problèmes de redirection, mais aussi au contexte du confinement, qui a perturbé notre rythme de travail et notre organisation.

Cela a causé du retard et parfois l'abandon de certaines fonctionnalités, comme les fonctionnalités suivantes :

- Citation de message (abandon)
- Tri sur les produits recherchés (abandon)
- Ajout de produit dans la base par un utilisateur (abandon)
- Création d'un "Compte producteur" (abandon)
- Fonction de la partie "Administrateur" (retard)

## Table des figures

FIGURE 1 : MAQUETTE DE LA PAGE D'ACCUEIL	9
FIGURE 2 : MAQUETTE DES FORMULAIRES	9
FIGURE 3 : MAQUETTE DE L'ICÔNE D'UTILISATEUR	10
FIGURE 4 : MAQUETTE DE LA PAGE D'UN PRODUIT	10
FIGURE 5 : MAQUETTE D'UN RÉSULTAT DE RECHERCHE	10
FIGURE 6 : MAQUETTE DE L'ACCUEIL DU FORUM	11
FIGURE 7 : MAQUETTE D'UN SUJET DE DISCUSSION	11
FIGURE 8 : BASE DE DONNÉES V1	16
FIGURE 9 : UN PEU TROP DE BOUTONS...	18
FIGURE 10 : BASE DE DONNÉES V2	20
FIGURE 11: SANS COMMENTAIRE	21
FIGURE 12 : BASE DE DONNÉES V3	24