
REVENTE
MARKET
Bateau Thibault

TABLE DES MATIÈRES

INTRODUCTION

Notre équipe AKEA, composée d'Axel ACHY, Khalil ABBIOUI, Eric BARNET et Anas MOUMEN, 4 développeurs full-stack travaillant au sein de l'entreprise InnoTech Solutions. Cette entreprise a été fondée en 2019 et elle propose diverses prestations informatiques pour des particuliers ou bien des entreprises.

Dans ce projet, nous avions eu pour objectif de développer un back-office pour un point de revente nommé ReventeMarket. ReventeMarket sont notre client et ils proposent des produits issus de la mer. Leur fournisseur Bateau Thibault les réapprovisionnent souvent chaque semaine.

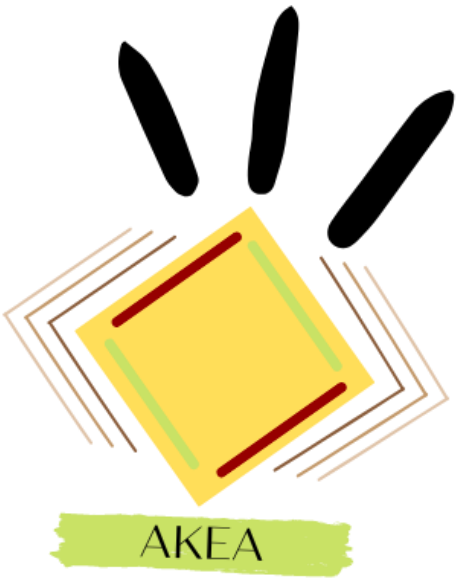
Le travail commence à 0 (zéro), contrairement à d'autres projets précédemment accomplis, il n'y a pas de PMV (Product Minimum Valuable). On ne part pas sur une refonte mais sur le développement complet d'un système.

1. ENTREPRISE : SA RAISON D'ÊTRE ET LE BESOIN DE REVENTEMARKET

ReventeMarket est un point de revente localisé un peu partout en France, ils proposent aux particuliers des produits frais toute la saison. C'est un intermédiiaire entre les consommateurs et le fournisseur. A ce jour, ReventeMarket gère toujours les stocks, les prix etc sur papier. Ils veulent numériser ce travail.

Le développement d'un back-office pour ReventeMarket est très utile et très important, utile dans un sens où le back-office permettra à ReventeMarket de connaître ses stocks, gérer ces derniers, modifier, supprimer, quand ils le souhaitent. Et important d'une part car c'est directement dessus que ReventeMarket pourra analyser son chiffre d'affaires, ses résultats comptables, les impôts sur les sociétés, les trimestres négatives (au niveau de la vente).

Donc ce projet est particulièrement intéressant car il va permettre d'accroître la productivité et le gain de temps de ReventeMarket.



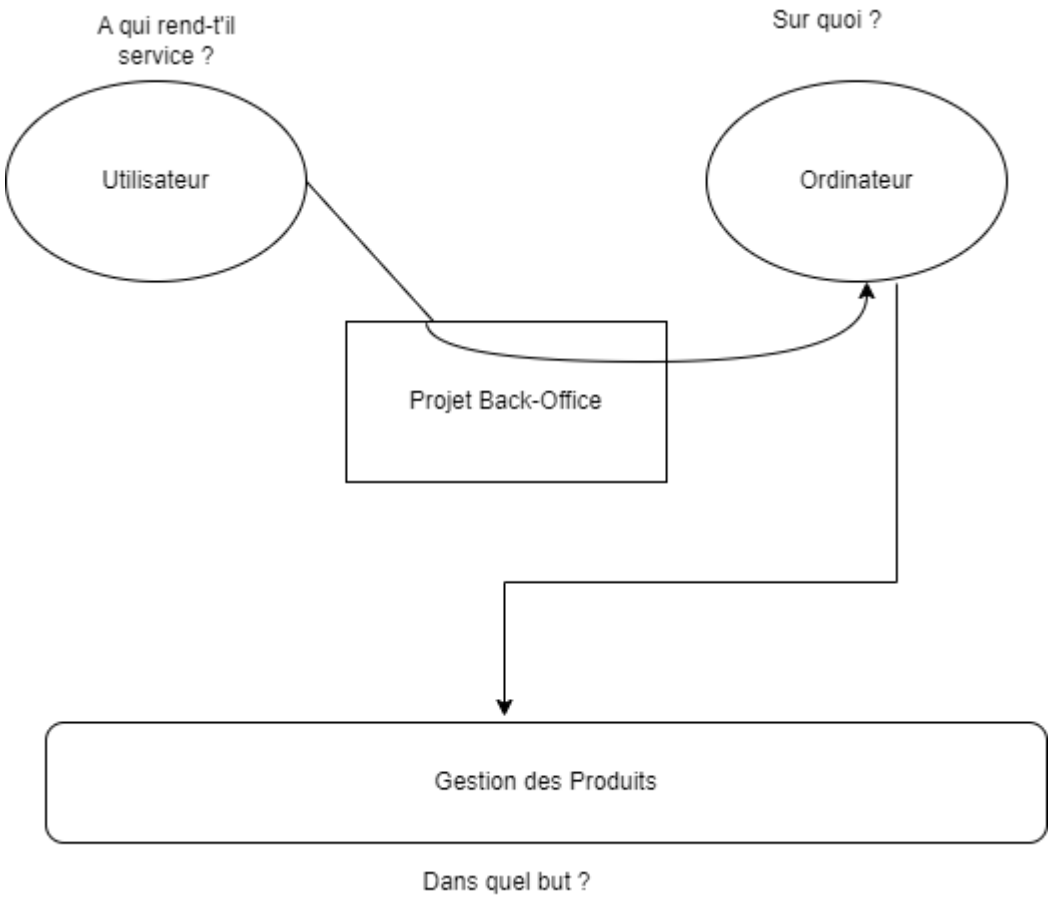
2. ANALYSE FONCTIONELLE

L'analyse fonctionnelle est d'abord une approche structurante au service de l'équipe en charge de la création ou de l'amélioration du produit, qui est ici une refonte d'une application web.

Pour faire cette analyse fonctionnelle, on va utiliser la méthode APTE, en utilisant le diagramme d'objectif afin démontrer l'utilité du produit en répondant aux attentes du client.

Le diagramme des interacteurs va nous permettre d'avoir vision globale des fonctionnalités de l'applications et de ses contraintes par rapport à son environnement.

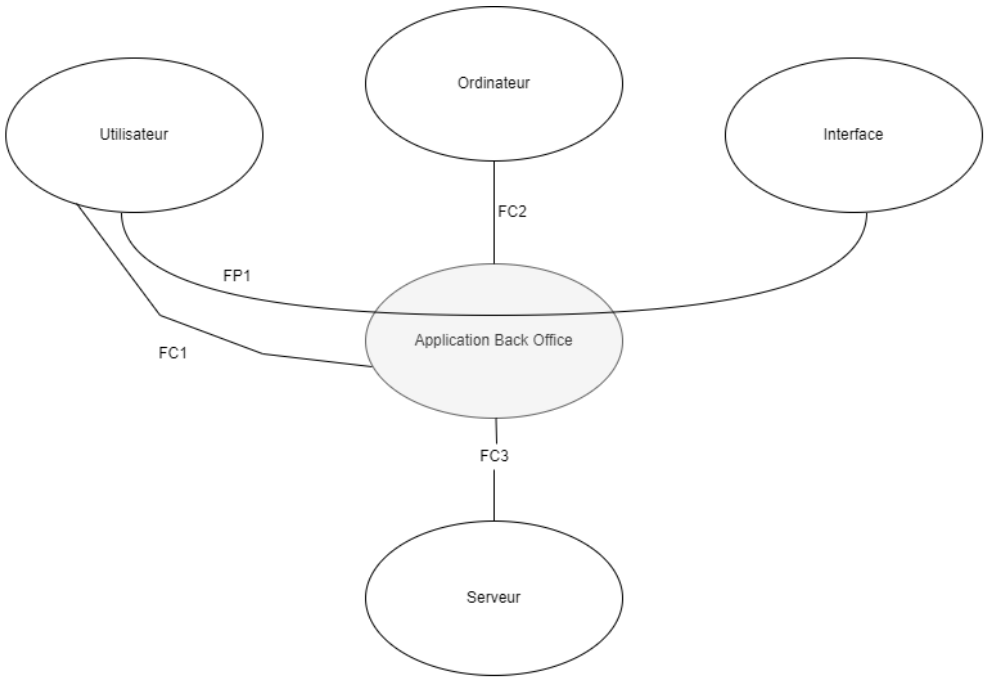
Ici, on applique le diagramme en fonction de notre cas :



Le diagramme d'objectif de notre Back-Office réponds aux question suivantes :

L'application rend service à l'utilisateur utilisant le back-office sur un ordinateur, dans le but de gérer les produits que propose le point de revente

Diagramme d'objectif



Fonctions principales :

FP1 : Gestion des produits via l'interface

Fonctions de contraintes :

FC1 : Il faut que l'utilisateur s'authentifie

FC2 : Le projet doit etre accessible sur tout les appareils

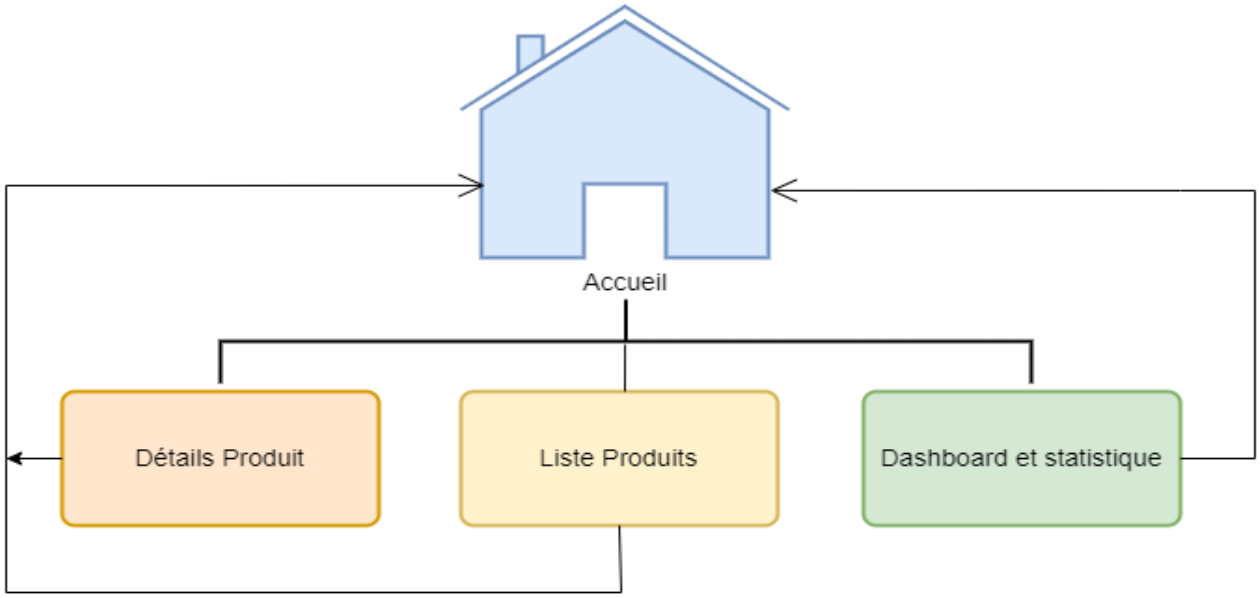
FC3 : Validation des données

.

Diagramme des interactions

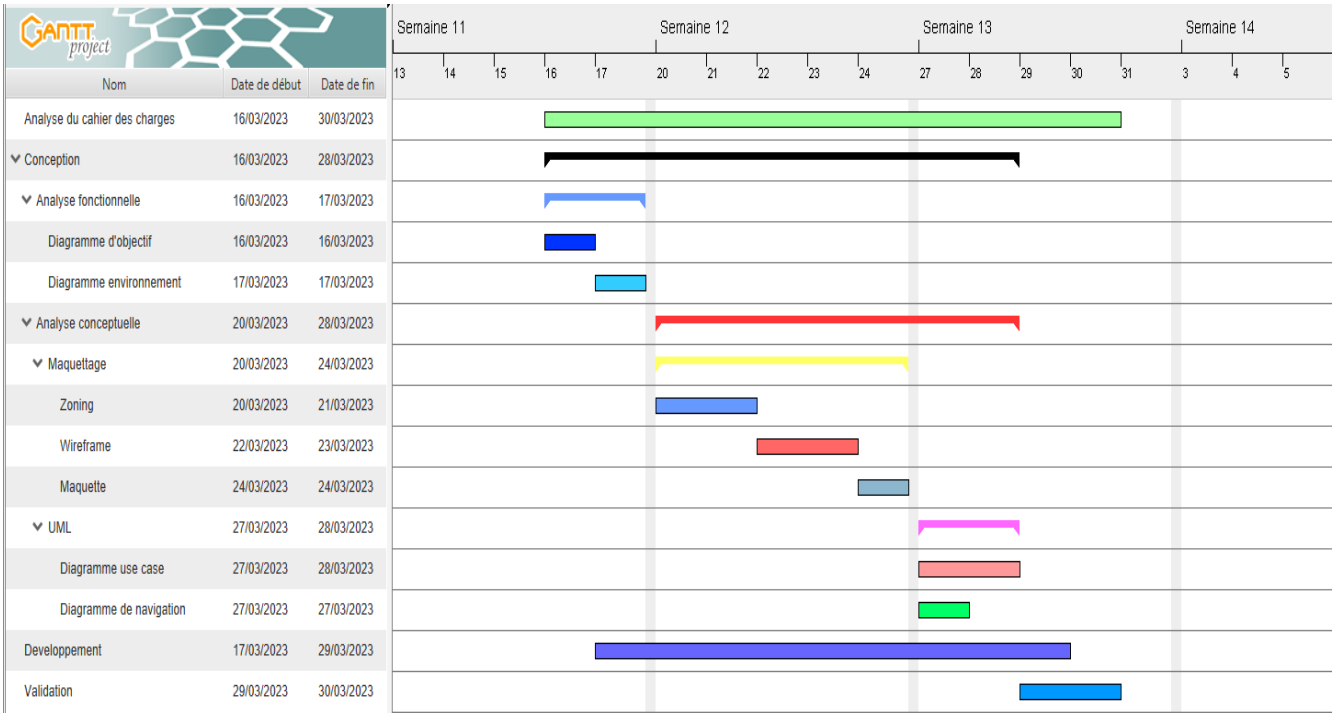
Le diagramme des interactions (aussi appelés diagramme d'environnement), établit des fonctions principales et des contraintes lié à notre projet. Ici pour le site Back Office, nous avons une fonction principale qui permet à l'utilisateur de consulter l'application et de pouvoir gerer des produits. En contrainte, qu'on doit s'authentifier, que se soit accessible sur ordinateur que les données soit valides.

3. ANALYSE CONCEPTUELLE



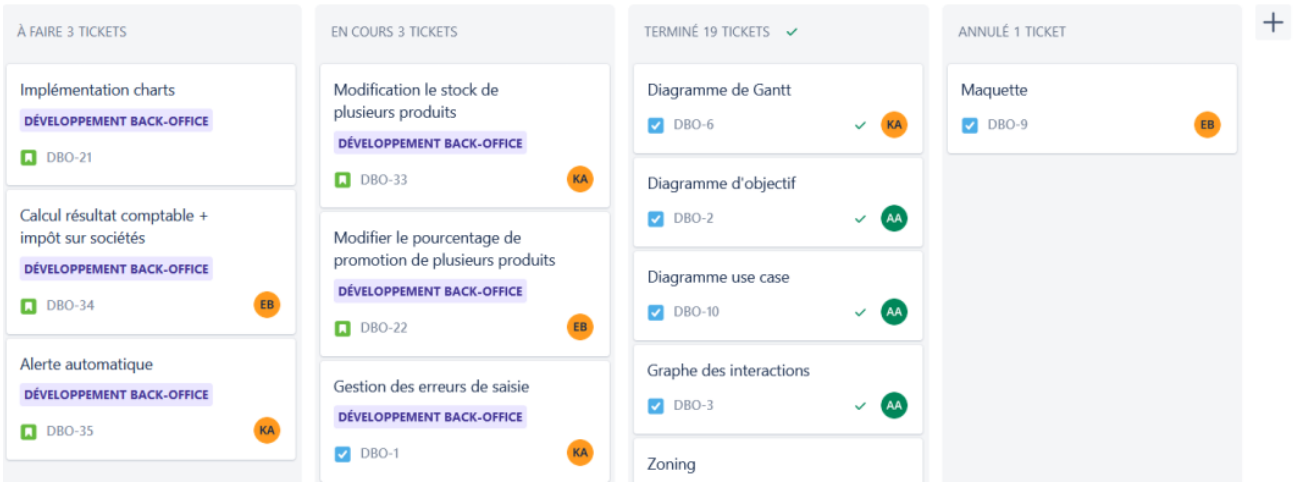
4. Diagramme de navigation

Ensuite, nous avons créé un diagramme de navigation avec le langage UML qui va établir comment l'utilisateur pourra naviguer dans l'application. Il y'aura un home représenter ici par la maison tout en haut. Depuis l'accueil, on pourra accéder aux vues 1-2-3 qui correspondent successivement au Details Produit, Liste Produits et Dashboard & statistique.



Nom	Date de début	Date de fin
Analyse du cahier des charges	16/03/2023	30/03/2023
Conception	16/03/2023	28/03/2023
Analyse fonctionnelle	16/03/2023	17/03/2023
Diagramme d'objectif	16/03/2023	16/03/2023
Diagramme environnement	17/03/2023	17/03/2023
Analyse conceptuelle	20/03/2023	28/03/2023
Maquettage	20/03/2023	24/03/2023
Zoning	20/03/2023	21/03/2023
Wireframe	22/03/2023	23/03/2023
Maquette	24/03/2023	24/03/2023
UML	27/03/2023	28/03/2023
Diagramme use case	27/03/2023	28/03/2023
Diagramme de navigation	27/03/2023	27/03/2023
Developpement	17/03/2023	29/03/2023
Validation	29/03/2023	30/03/2023

Diagramme de Gantt



Jira

Avec le délai imposé, nous avons utilisé la méthode Extreme Programming. Cette méthodologie repose sur des cycles de développement rapide d'où le terme "Extreme". Cette méthode est un choix judicieux dans notre cas car les critères principaux pour l'utilisation de cette méthode son le travail en groupe et les délais courts.

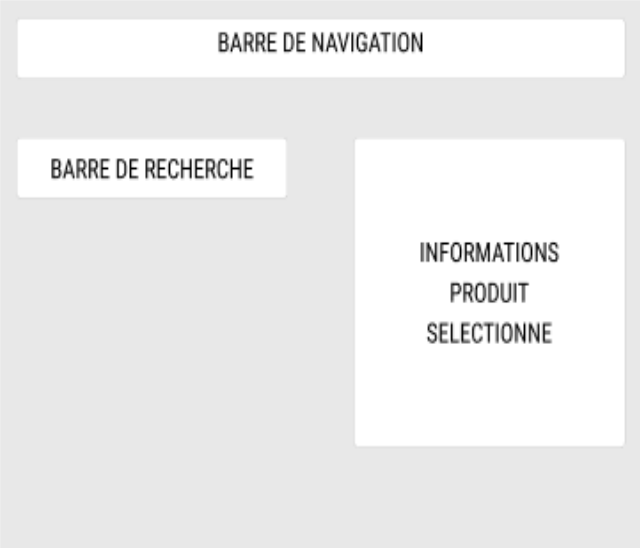
Lors du processus de planification du projet nous avons divisé ce projet en différentes tâches qui représentent généralement soit une page ou une fonctionnalité.

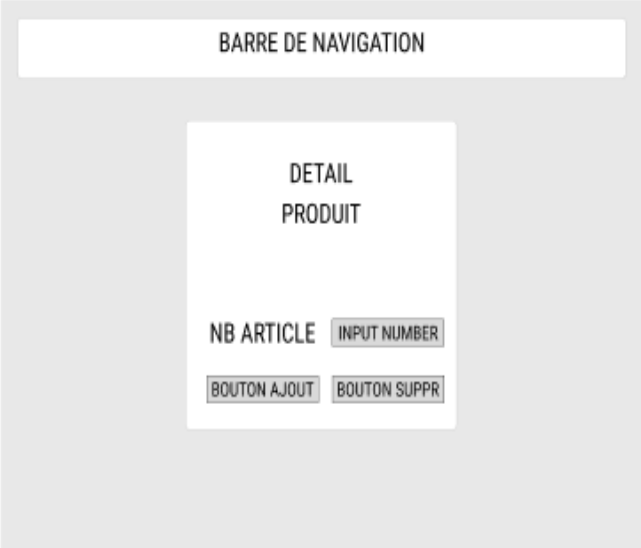
Après la planification faite via Gantt Project il n'y avait plus qu'à attribuer les différentes tâches aux développeurs grâce à Jira A l'aide de ces outils et de cette méthodologie, nous avons pu nous organiser de façon efficace et réagir rapidement lorsqu'un souci se présentait à nous grâce à ces derniers.

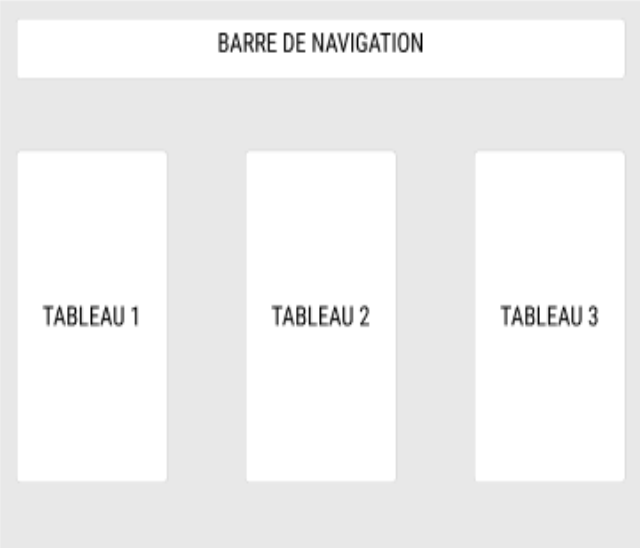
1. Maquettage de l'application

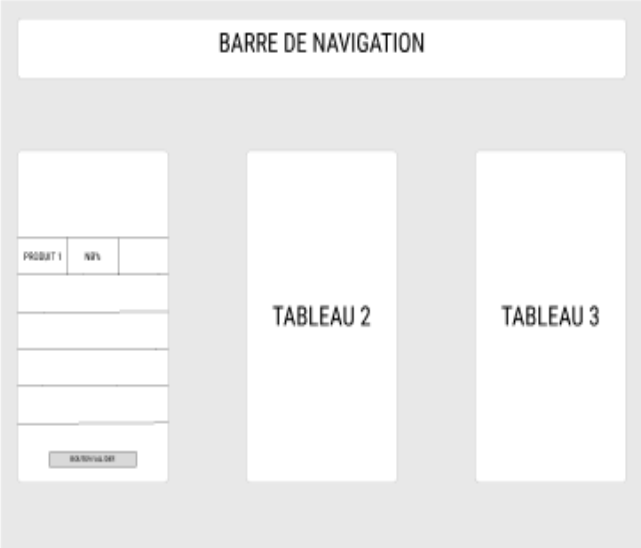
a. Zoning

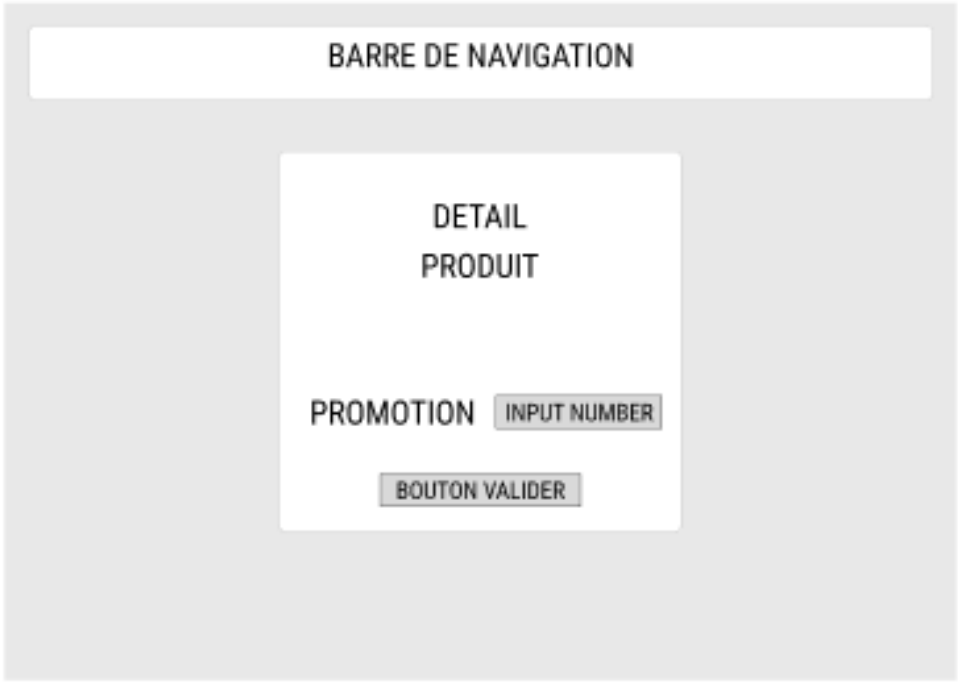
Pour commencer à réaliser l'application nous avons d'abord maquetter l'application en commençant par le zoning. Le zoning va définir les zones de chaque page qu'on établit avec un logiciel appelé Figma.

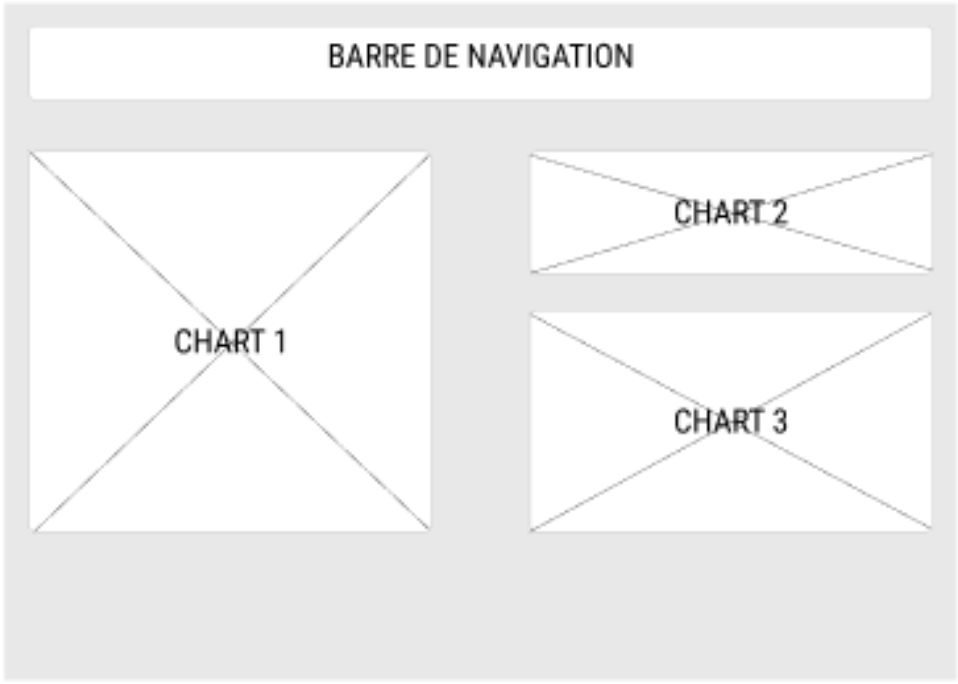

BARRE DE NAVIGATION
BARRE DE RECHERCHE
INFORMATIONS
PRODUIT
SELECTIONNE


BARRE DE NAVIGATION
DETAIL
PRODUIT
NB ARTICLE
INPUT NUMBER
BOUTON AJOUT
BOUTON SUPPR


BARRE DE NAVIGATION
TABLEAU 1
TABLEAU 2
TABLEAU 3

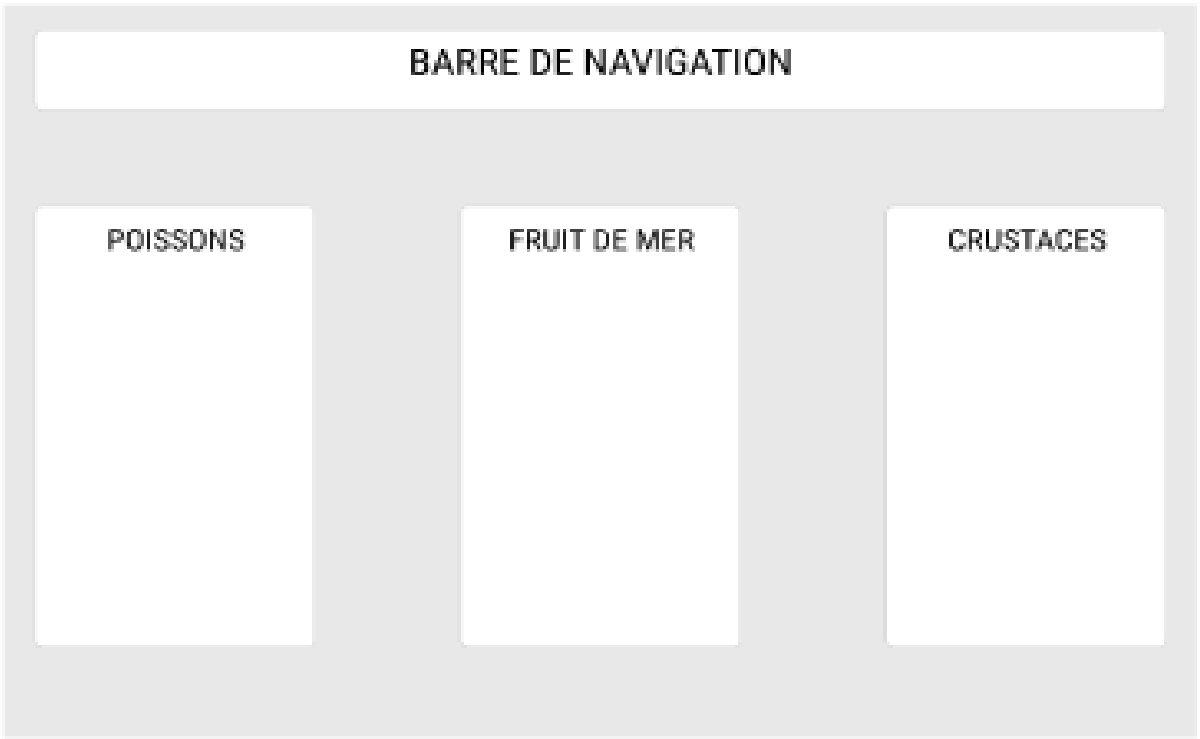

BARRE DE NAVIGATION
TABLEAU 2
TABLEAU 3


BARRE DE NAVIGATION
DETAIL
PRODUIT
PROMOTION
INPUT NUMBER
BOUTON VALIDER

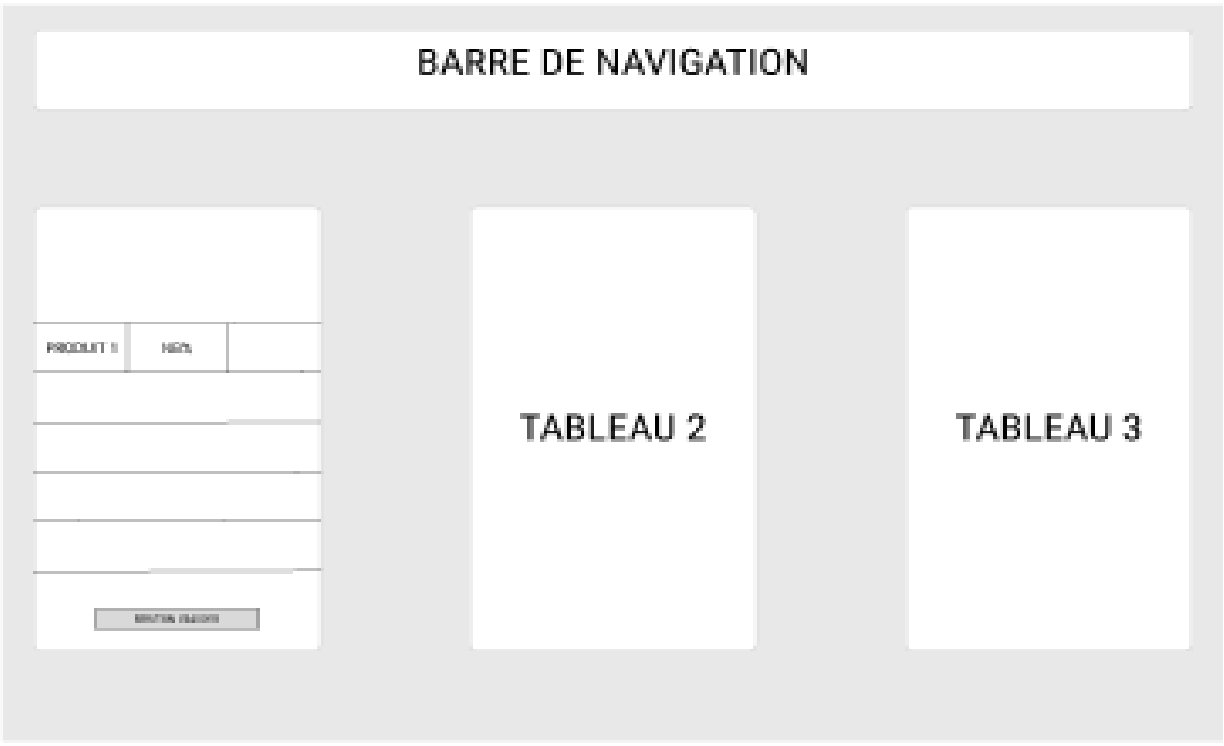

BARRE DE NAVIGATION
CHART 1
CHART 2
CHART 3

a. Wireframe

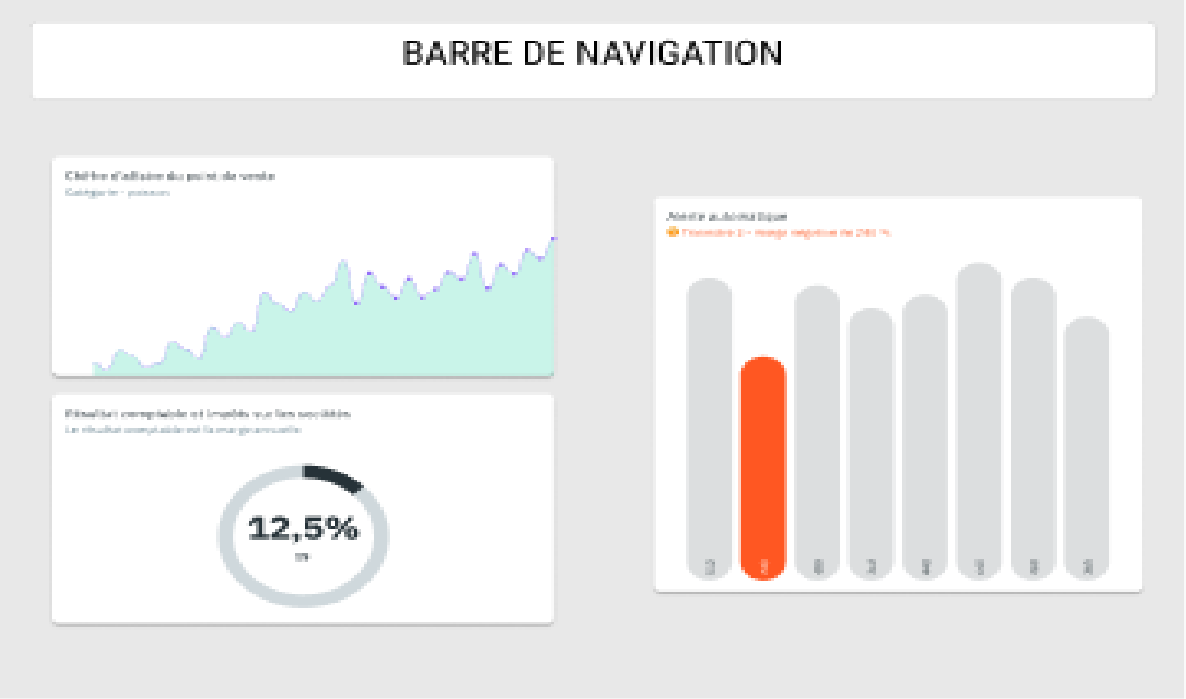

BARRE DE NAVIGATION
Recherchez un produit....
Nom :
Prix :
Prix en promo :
% de promo :
Quantité de stock :
Nombre d'article vendus :
Commentaires :


BARRE DE NAVIGATION
POISSONS
FRUIT DE MER
CRUSTACES

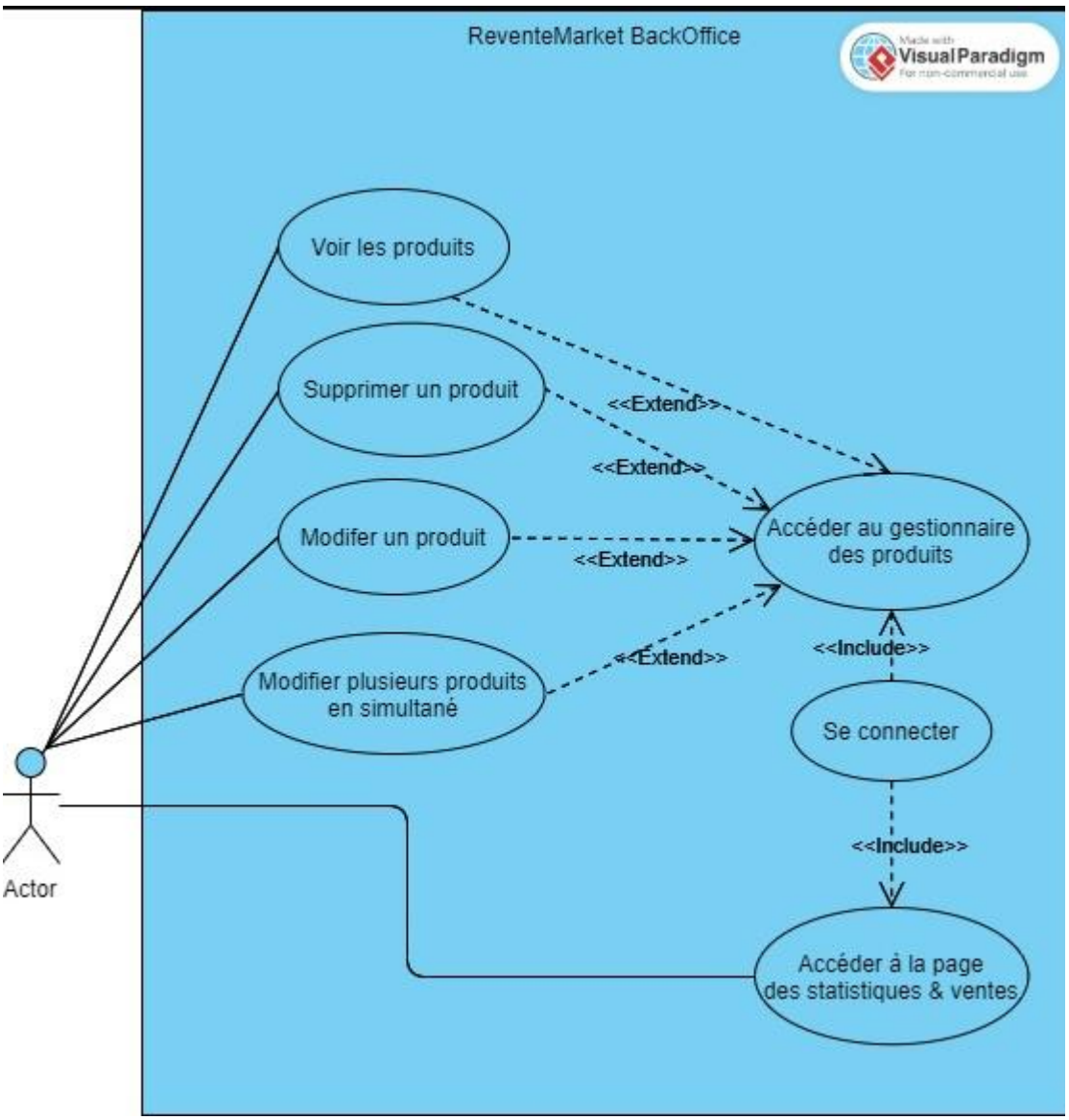

BARRE DE NAVIGATION
Aile de raie
Commentaires :
% de promotion :
Nombre d'article :
Ajouter article(s)
Supprimer article(s)


BARRE DE NAVIGATION
TABLEAU 2
TABLEAU 3


BARRE DE NAVIGATION
Aile de raie
Commentaires :
% de promotion :
Valider promotion
Nombre d'article :
Ajouter article(s)
Supprimer article(s)


BARRE DE NAVIGATION
12,5%

b. UML : Diagramme de cas d'utilisation



Le diagramme de cas d'utilisation de notre application se construit comme l'image ci-dessus. L'utilisateur peut accéder au panier, à la liste des restaurants, à la liste des recettes, la liste des catégories des produits et à la liste des bateaux et il peut aussi accéder à la page contact. Dans panier, l'utilisateur pourra supprimer ou un ajouter un produit dans le panier.

2. DEVELOPPEMENT

Petite introduction concernant Angular :

Vue.js et Python DRF sont deux technologies très populaires pour le développement de sites web. Vue.js c'est un framework front-end open source pour faire des sites web dynamiques. Cela permet de créer des interfaces utilisateur qui sont réactives. Pour Python DRF, est un framework back-end pour le développement d'API RESTful en Python.

Vue.js est un framework basé sur la bibliothèque JavaScript pour la création d'interfaces utilisateur.
Il a été créé pour être simple à utiliser, et il est aussi très performant et flexible. Vue.js utilise système de composants, ce qui permet de créer des interfaces utilisateur réutilisables. Il est lui aussi très flexible, car on peut l'utiliser avec différents outils et bibliothèques pour créer des applications web.

Python DRF, c'est un framework pour la création de services Web RESTful en Python.
Il permet aux développeurs de créer des API RESTful facilement. Il propose des fonctionnalités qui simplifient la gestion des API, comme par exemple la possibilité de gérer l'authentification des utilisateurs, de traiter les requêtes et les réponses.

Un exemple de code représentant la méthode updateQuantityInStock dans le dossier store et le fichier index.ts :

```
actions: {
  async updateQuantityInStock(
    context: ActionContext<RootState, RootState>,
    payload: StockUpdatePayload
  ) {
    const { commit, state, rootGetters } = context;
    const { productId, stockChange } = payload;
    const token = rootGetters["auth/getToken"];

    if (!token) {
      console.error("Accès refusé. Veuillez vous connecter.");
      return;
    }

    const product = state.products.find((p: Product) => p.id === productId);

    if (product) {
      const newQuantityInStock = product.quantityInStock + stockChange;
      console.log(
        "New quantity :",
        newQuantityInStock,
        "Produit de quantité dans la table : ",
        product.quantityInStock,
        "Stockchange:",
        stockChange
      );
      // Envoyer une requête PATCH au backend
      const response = await fetch(
        `http://127.0.0.1:8000/update_product_stock/${productId}/`,
        {
          method: "PATCH",
```

```
    // Envoyer une requête PATCH au backend
    const response = await fetch(
      `http://127.0.0.1:8000/update_product_stock/${productId}/`,
      {
        method: "PATCH",
        headers: {
          "Content-Type": "application/json",
          Authorization: `Bearer ${token}`,
        },

        body: JSON.stringify({ quantityInStock: newQuantityInStock }),
      }
    );

    if (response.ok) {
      const updatedProduct = await response.json();
      console.log("Produit msie à jour après requete : " + updatedProduct);
      commit("updateQuantityStock", {
        productId,
        newQuantityInStock: updatedProduct.quantityInStock,
      });
      product.quantityInStock = updatedProduct.quantityInStock;
    } else {
      const error = await response.json();
      console.error("Error updating product stock:", error);
    }
  }
},
```

La méthode updateQuantityInStock permet de mettre à jour la quantité de stock d'un produit en appelant une API backend. Elle prend en paramètre le contexte Vuex, qui contient notamment le state et les getters, ainsi qu'un objet payload contenant l'ID du produit et la variation de stock à effectuer. La méthode recherche ensuite le produit correspondant dans le state, puis envoie une requête PATCH au backend avec la nouvelle quantité de stock. Si la requête réussit, le produit est mis à jour dans le state Vuex grâce à la mutation "updateQuantityStock".

En conclusion, le développement du back-office pour ReventeMarket est un projet important pour l'entreprise et va permettre une numérisation de leurs activités, ce qui accroîtra leur productivité et leur gain de temps. Il permettra également à l'entreprise de mieux gérer leurs stocks, d'analyser leurs résultats comptables et de prendre des décisions stratégiques en temps réel.

Quant à l'avenir du projet, il est clair que nous avons encore beaucoup de travail à faire. Nous devrons travailler en étroite collaboration avec ReventeMarket pour nous assurer que le système répond à leurs besoins et qu'il est facilement accessible pour eux. Nous devrons également nous assurer que le système est sécurisé et qu'il est évolutif pour répondre aux futurs besoins de l'entreprise.

Enfin, l'équipe AKEA est très satisfaite de ce projet jusqu'à présent. Nous avons réussi à relever un défi complexe et à concevoir un système efficace et facile à utiliser. Nous sommes impatients de continuer à travailler sur ce projet et de le voir se développer pour répondre aux besoins de ReventeMarket.