**Chapitre 3 : Backlog du produit**

**Introduction**

Dans le chapitre précédent, nous sommes intéressés à la stratégie marketing du lancement d'Waste Not.

Dans ce chapitre, nous présenterons les choix technologiques adoptés pour répondre à ces besoins, ainsi que leur spécification en utilisant des diagrammes de cas d'utilisation et de classes. Enfin, nous finaliserons cette phase en créant le backlog de produit, en planifiant les sprints et en établissant les releases.

* 1. **Étude préliminaire**

L'étude préliminaire constitue une étape essentielle du processus de développement. Elle consiste à déterminer les besoins fonctionnels et non fonctionnels, en s'appuyant sur les spécifications du client. L’objectif de cette étape est principalement la spécification détaillée des fonctionnalités attendues d'un service ou d'un produit. Cette étude sera ainsi l'objectif de cette partie.

* + 1. **Identification des exigences fonctionnelles**

Les exigences du système définissent les actions et les réactions que l'application mobile Waste Not doit effectuer en réponse aux demandes de son acteur principal. Elles comprennent la représentation abstraite des services que l'application est censée fournir à ses utilisateurs. Les exigences de notre système sont les suivantes :

* **Gestion de l'authentification :** Elle permet aux utilisateurs de s’inscrire et de s’identifier.
* **Gestion de profil :** L'application doit permettre aux utilisateurs de gérer leurs comptes y compris la modification des informations du profil, la mise à jour des préférences, etc.
* **Gestion des objets :** Cette fonctionnalité permet aux utilisateurs de créer, modifier et supprimer les objets qu'ils souhaitent vendre.
* **Recherche filtré des objets recyclables :** Les utilisateurs peuvent rechercher sur les objets recyclables disponibles dans l`application en utilisant des filtres tels que le type d’objet et l'emplacement.
* **Recherche les points de collecte :** Les utilisateurs peuvent rechercher les points de collecte les plus proches de leur position géographique afin de pouvoir y déposer les objets recyclables.
* **Discussion en temps réel :** Il s'agit de permettre au vendeur et à l'acheteur de discuter en eux afin que ce dernier soit mieux satisfait.
* **Système d’achat :** Cette fonctionnalité permet aux utilisateurs de sélectionner les objets qu'ils souhaitent acheter, de vérifier la disponibilité et la quantité, de les ajouter à leur panier et de passer une commande.
* **Gestion des favoris :** Les utilisateurs peuvent ajouter des objets qu'ils souhaitent acheter à une liste de favoris pour une consultation ultérieure.
* **Gestion de réclamation :** Elle permet aux utilisateurs de signaler un problème.
* **Système de notification :** Elle consiste à informer l'utilisateur par les nouveaux produits les plus récemment ajoutés.
* **Système de paiement :** L'application est équipée d'un système de paiement sécurisé qui permet aux utilisateurs de payer et de recevoir de l'argent.
* **Consultation des statistiques :** Notre système permet à l'administrateur de consulter les statistiques. Cette fonctionnalité permet également de visualiser la répartition géographique des utilisateurs de l'application Waste Not, le nombre de comptes inscrits, le nombre de points de collecte disponibles, ainsi que le nombre de commandes effectuées.
  + 1. **Identification des exigences non fonctionnelles**

Les exigences non fonctionnelles sont celles qui sont considérées comme importantes, voire critiques, aux yeux des utilisateurs, et qui garantissent le bon fonctionnement du système. Pour notre système, ces exigences sont les suivantes :

* **Sécurité :** Assurer la confidentialité et l'intégrité des informations personnelles des utilisateurs et toutes les opérations administratives. L’application doit être sécurisée pour que les données des utilisateurs de l’application ne seront pas divulguées.
* **Performance :** L'application doit réduire le temps de réponse pour le chargement de pages, l'affichage des résultats et les délais de rafraîchissement.
* **Facilité d'utilisation :** Notre système doit être facile à manipuler pour l'utilisateur.
* **Responsive :** la plateforme doit être adaptée aux différents terminaux qu’un utilisateur peut utiliser (ordinateur portable, téléphone mobile, tablette, etc.).
* **Intégrité des données :** Des contrôles de saisie doivent être effectués pour s’assurer que les données soient cohérentes et pertinentes.
* **Ergonomie et Convivialité :** L'application doit être conviviale c’est-à-dire simple, ergonomique et s'adapte sans effort à l’utilisateur en termes de navigation entre les différentes interfaces, les couleurs et les textes de mise en page textes utilisés (clairs et faciles à utiliser).
* **Accessibilité au GPS :** Demander l’autorisation pour accéder à la géolocalisation des utilisateurs et afficher les objets disponibles dans l'application, qui sont positionnés en fonction de rayons définis par rapport à sa position.
  1. **Spécifications techniques**

Après avoir établi les exigences fonctionnelles et non fonctionnelles, nous allons présenter la spécification technique. Puisque les besoins techniques donnent une vision globale sur l'architecture générale de notre projet, ce qui entraîne le développement de l'application mobile.

Cette section est consacrée à l'identification des technologies à utiliser pour garantir le bon fonctionnement de l'application. En effet, la section suivante met l'accent sur le choix de certaines technologies suivante : Les outils de conception, de planification, de design, de base de données, l'architecture logicielle et les technologies de développement.

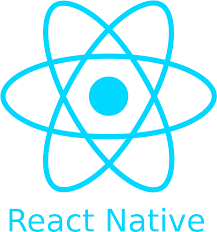
* + 1. **Les technologies de Waste Not**

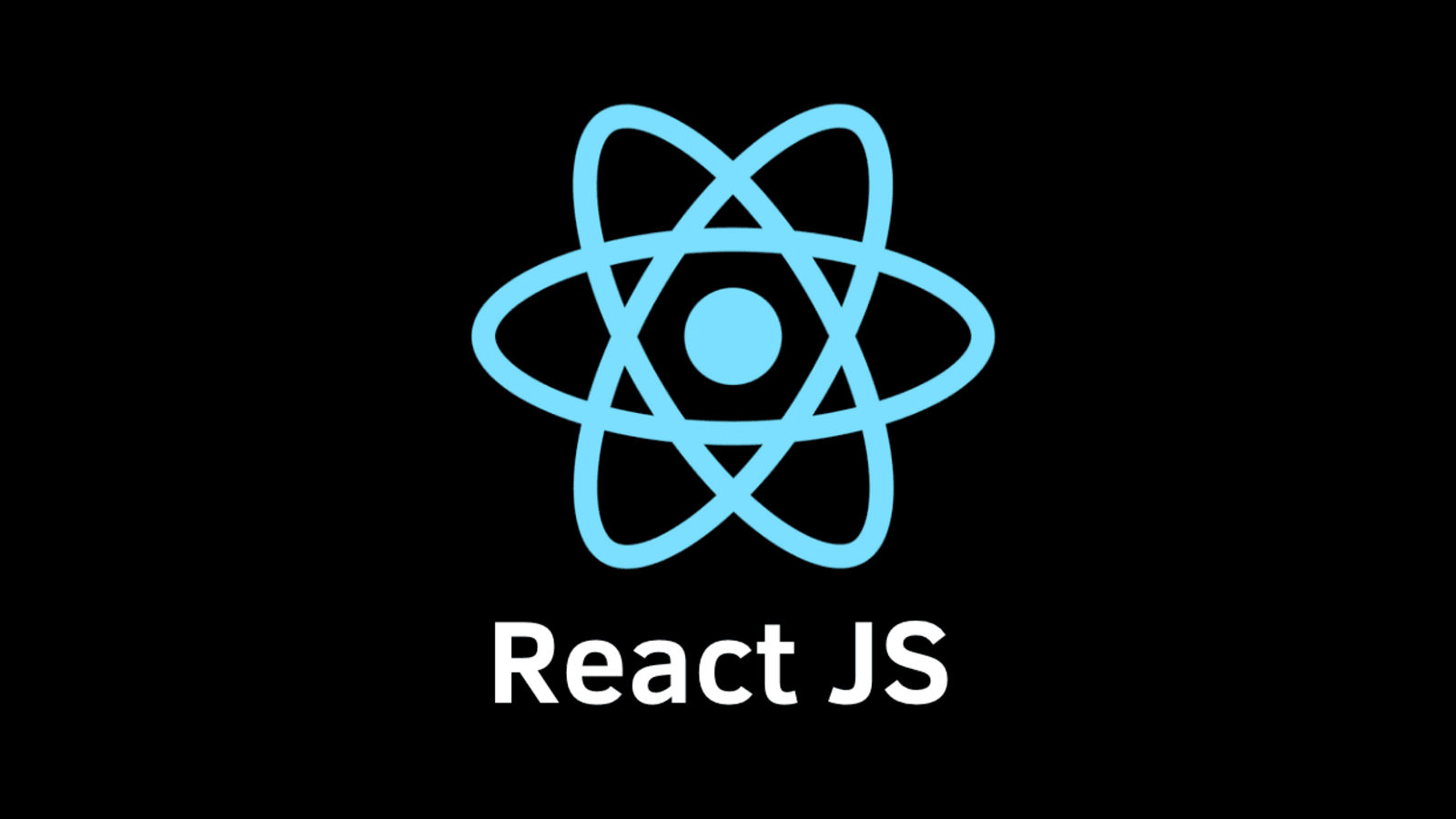
Pour la réalisation de l'application, nous avons utilisé les technologies suivantes :

**HTML5 :** signifie « HyperText Markup Language » qu'on peut traduire par « langage de balises pour l'hypertexte ». Il sert à créer et représenter le contenu d'une page web et sa structure [ref].

 **Bootstrap :** est un framework de développement frontale, gratuite et open source pour la création de sites et d'applications Web. Il repose sur HTML, CSS et JavaScript (JS) pour faciliter le développement de sites et d'applications réactives et tout-mobile [ref].

 **Laravel :** est un framework gratuit et open source qui utilise le langage PHP. Il a été conçu pour rendre le développement d'applications web rapide et facile [ref].

 **React Native** **:** fait référence à un cadre basé sur JavaScript utilisé pour créer des applications mobiles natives sur Android et la plate-forme iOS [ref].

 **React JS** **:** est une bibliothèque JavaScript open-source utilisée pour créer des interfaces utilisateur interactives et dynamiques pour les applications web. Il a été développé par Facebook et est devenu très populaire dans la communauté des développeurs[ref].

* + 1. **Outil de paiement en ligne**

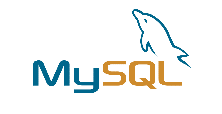
 **Stripe :** c'est une plateforme de traitement des paiements, elle permet de transférer de l'argent à l'aide d'une carte de crédit ou de débit.[ref]

* + 1. **Outil de documentation et de test des services web**

**Postman :** est une application logicielle qui permet aux développeurs de tester, de documenter et de déboguer les API (Interfaces de Programmation d'Application). En d'autres termes, il permet aux développeurs de simuler les requêtes HTTP et de voir les réponses retournées par les serveurs, ce qui facilite le processus de développement et de maintenance des API [ref].

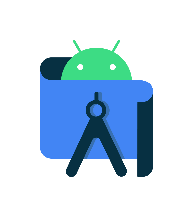
* + 1. **La base de données**

**SQL :** (Structured Query Language) est un langage de « programmation » standardisé utilisé pour gérer des bases de données relationnelles et effectuer diverses opérations sur les données qu’elles contiennent [ref].

**MySQL :** est un système de gestion de bases de données relationnelles utilisant le langage de programmation SQL. Il propose une version open source qui permet à l’utilisateur d’accéder au code source et de le modifier [ref].

* + 1. **Les outils de développement**

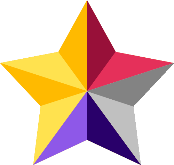
**Visual Studio Code :** est un éditeur de code open-source développé par Microsoft supportant un très grand nombre de langages grâce à des extensions. Il supporte l’auto-complétion, la coloration syntaxique, le débogage, et les commandes git [ref].

**Android Studio :** est l'environnement de développement intégré (IDE) officiel des applications Android. Basé sur le puissant outil de développement et d'édition de code d'IntelliJ IDEA [ref].

 **Composer :** est un outil de gestion des dépendances en PHP. Il vous permet de déclarer les librairies dont dépend votre projet et il les gère (installe/met à jour) [ref].

**XAMPP :** est un ensemble de logiciels qui permet de mettre en place facilement un serveur Web confidentiel, un serveur FTP et un serveur de messagerie électronique. Il contient principalement MYSQL, PHP et PERL [ref].

* + 1. **Les outils de conception**



**StarUML :** est un outil de conception de diagrammes en vue d'une programmation. Il est capable de prendre en charge de nombreux diagrammes commerciaux et techniques comme UML [ref].

* + 1. **Les outils de design**



**Adobe Illustrator :** est un logiciel de création graphique vectorielle qui offre des outils de dessin vectoriel puissants. Les images vectorielles sont constituées de courbes générées par des formules mathématiques [ref].



**Adobe XD :** est un outil de conception et de prototypage vectoriel spécialement conçu pour la conception de sites Web et les applications mobiles. Ce logiciel offre une interface simplifiée, une multitude de fonctionnalités et d’extensions gratuits, une possibilité d’édition très facile ainsi qu’un dynamisme et interactivité aux designs crées avec un service d’hébergement en ligne [ref].

* + 1. **Outil de planification**



**Trello :** est un outil de gestion de projet en ligne. Il permet une excellente visualisation des tâches et de leur avancement, notamment grâce à la création de catégories ou la mise en place de dates limites [ref].

* + 1. **L'architecture logicielle**

**Architecture MVC(MODEL-VIEW-CONTROLLER) :** est une architecture de développement visant à séparer le code source en modules. En effet, ce modèle très répandu, consiste à séparer distinctement l’accès aux données (bases de données), la vue affichée à l’utilisateur et la logique métier [ref].



Figure \*\* : Architecture MVC [ref]

Ainsi, comme le démontre la figure ~~\*\*~~, il y a trois parties distinctes :

* Le modèle qui gère les données.
* La vue affichée à l’utilisateur qui représente l'interface utilisateur.
* Le contrôleur qui traite les actions de l'utilisateur et coordonne l'interaction entre le modèle et la vue.
  1. **Spécifications générales des exigences**

Après avoir défini les spécifications techniques, nous proposons la spécification des exigences pour répondre aux besoins identifiés.

Cette étape implique tout d'abord l'élaboration d'un diagramme de cas d'utilisation général et d'un diagramme de classes pour le projet. Ensuite, nous procéderons à la classification et à la planification des cas d'utilisation en itérations.

* + 1. **Détermination du diagramme de cas d'utilisation global**

Le diagramme de cas d'utilisation est un outil de modélisation UML qui permet de décrire le comportement fonctionnel d'un système logiciel. Bien qu'il soit souvent utilisé pour des présentations auprès de la direction ou des parties prenantes d'un projet, il est plus adapté pour le développement des cas d'utilisation concrets [ref].

Le diagramme de cas d'utilisation aide à organiser les exigences fonctionnelles du système, à définir les acteurs impliqués, généralement des individus, et à établir les relations entre eux.

Selon le diagramme ci-dessus (voir figure \*\*), on distingue 4 acteurs qui sont :

* **Utilisateur :** il peut s’inscrire pour créer un nouveau compte, gérer son profil pour mettre à jour ses informations personnelles, envoyer une réclamation et discuter en temps réel.
* **Administrateur :** il peut gérer la liste des catégories d’objets afin d’ajouter, modifier ou supprimer un objet selon les besoins. Il peut également gérer la liste des points de collectes pour ajouter ou modifier ou supprimer un point de collecte aussi et consulter les statistiques.
* **Vendeur :** il hérite toutes les actions qu’un simple utilisateur peut faire, mais il peut en particulier gérer les objets à vendre afin d’ajouter, modifier ou supprimer un objet. De plus, il peut gérer leurs commandes dans le but de valider ou supprimer une commande.
* **Acheteur :** il hérite également toutes les actions qu’un simple utilisateur peut effectuer. De même, il peut consulter les points de collecte et les objets disponibles, gérer la liste des favoris, gérer l’historique d’achat, recevoir des notifications sur les objets les proches et acheter des objets soit en ligne, soit sur place.

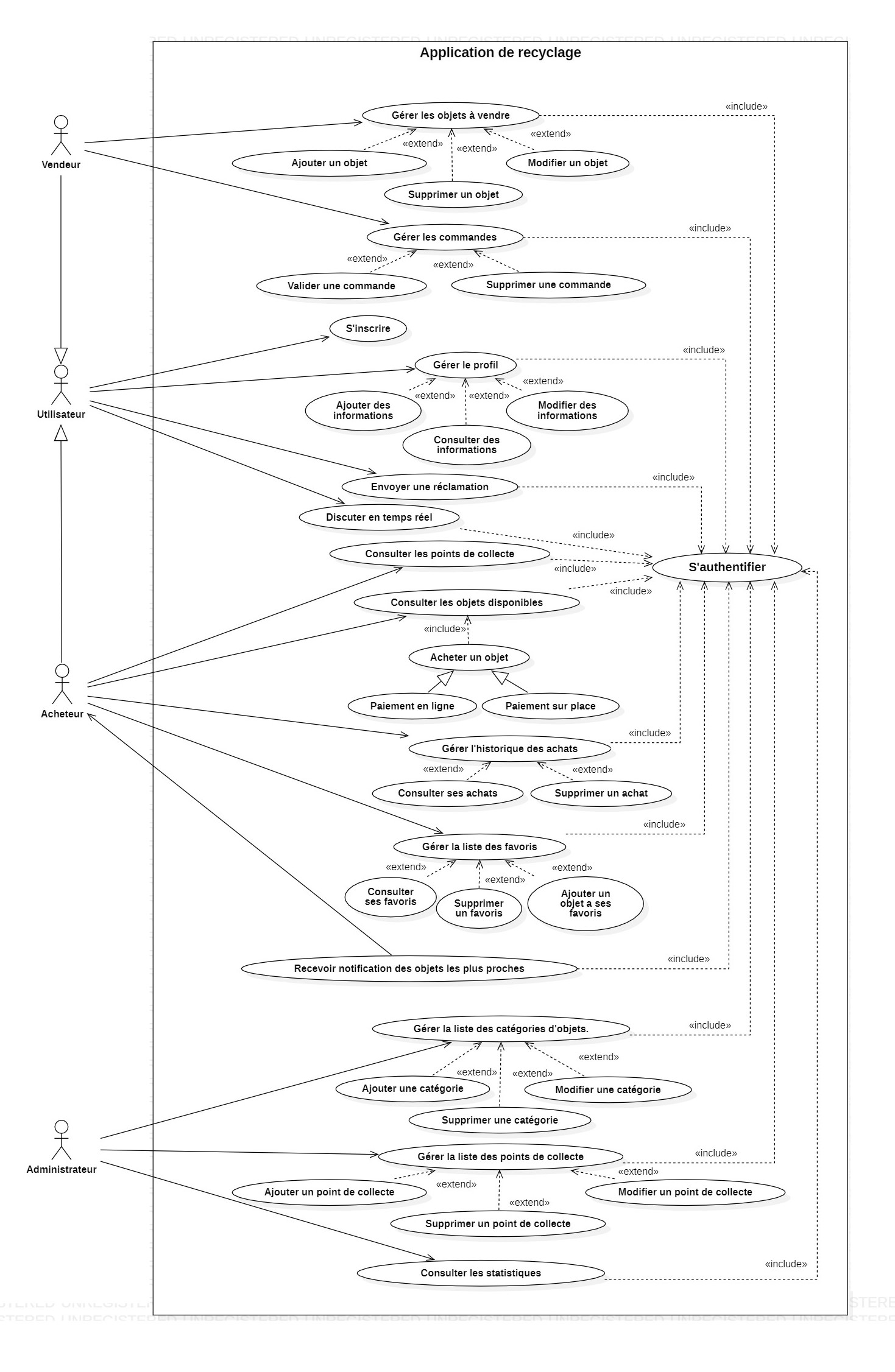


Figure \*\* : Le diagramme de cas d'utilisation global

* + 1. **Détermination du diagramme de classes du projet**

Le diagramme de classe est un diagramme UML qui permet de représenter visuellement les différents types d'objets présents dans un système ainsi que les relations statiques qui existent entre eux. Il est souvent utilisé pour explorer les concepts liés à un domaine, comprendre les exigences logicielles et décrire les conceptions détaillées.

* Un utilisateur peut être un acheteur ou un vendeur.
* Un vendeur peut gère plusieurs objets en les ajoutant, modifiant, supprimant, ou affichant. Et un objet peut être gérée par plusieurs vendeurs.
* Un vendeur gère plusieurs commandes en les validant, supprimant, ou affichant. Et une commande peut être gérée par plusieurs vendeurs.
* Un acheteur peut acheter un ou plusieurs objets et valider un paiement. Et un objet peut être acheté par plusieurs acheteurs.
* Un acheteur peut accéder un ou plusieurs points de collecte. Et un point de collecte peut être accédé par plusieurs acheteurs.
* Un administrateur gère plusieurs catégories en les ajoutant, modifiant, ou supprimant. Et une catégorie ne peut être gérée que par un seul administrateur.
* Un administrateur gère plusieurs points de collecte en les ajoutant, modifiant, affichant, ou supprimant. Et un point de collecte ne peut être gérée que par un seul administrateur.
* Un administrateur peut consulter les statistiques. Et la statistique ne peut être consulté que par un seul administrateur.

La figure numéro \*\* illustre clairement les principales classes de notre application :

- Classe Utilisateur : est la classe qui contient toutes les informations concernant l'utilisateur et contient deux méthodes, une méthode « Inscription » qui permet de créer un compte et une méthode « authentification».

- Classe Vendeur: Il s'agit d'une classe qui englobe toutes les actions possibles pour un vendeur. Elle comprend une méthode « gérer\_profil » permettant de mettre à jour les informations personnelles, une autre méthode « gérer\_commande » pour la validation ou la suppression d'une commande, ainsi qu'une méthode « gérer\_objet » pour gérer les objets mis en vente.

- Classe Administrateur : sert à présenter les principales fonctionnalités d'un administrateur ; un administrateur peut gérer la liste des points de collecte et la liste des catégories des objets ainsi consulter les tableaux de bord des actions d'achats.

- Classe Acheteur: est la classe qui contient toutes les informations concernant l'acheteur et contient quatre méthodes, une méthode nommée « gérer\_profil » pour mettre à jour les informations personnelles, une méthode appelée « Acheter\_objet » pour procéder à l'achat d'un objet, ainsi qu'une méthode nommée « gérer\_favoris » pour gérer la liste des éléments préférés. Enfin, une méthode nommée « gérer\_historique » est envisagée pour permettre la consultation ou la suppression de l'historique des achats.

- Classe Catégorie : cette classe représente la catégorie des objets bien déterminée. Elle possède tois méthodes « Ajouter\_catégorie », « Modifier\_catégorie » et « Supprimer\_catégorie ».

- Classe Points\_collecte : Comporte quatre méthodes qui est « Ajouter\_pc », « Modifier\_pc », « Supprimer\_pc » et « Afficher\_pc ».

- Classe Objet : Inclut quatre méthodes : « Supprimer\_objet », « Modifier\_objet », « Afficher\_objet » et « Ajouter\_objet », qui permettent respectivement de supprimer, modifier, afficher et ajouter un objet.

- Classe Commande : Possède deux méthodes « Valider\_commande » et « afficher\_commande ».

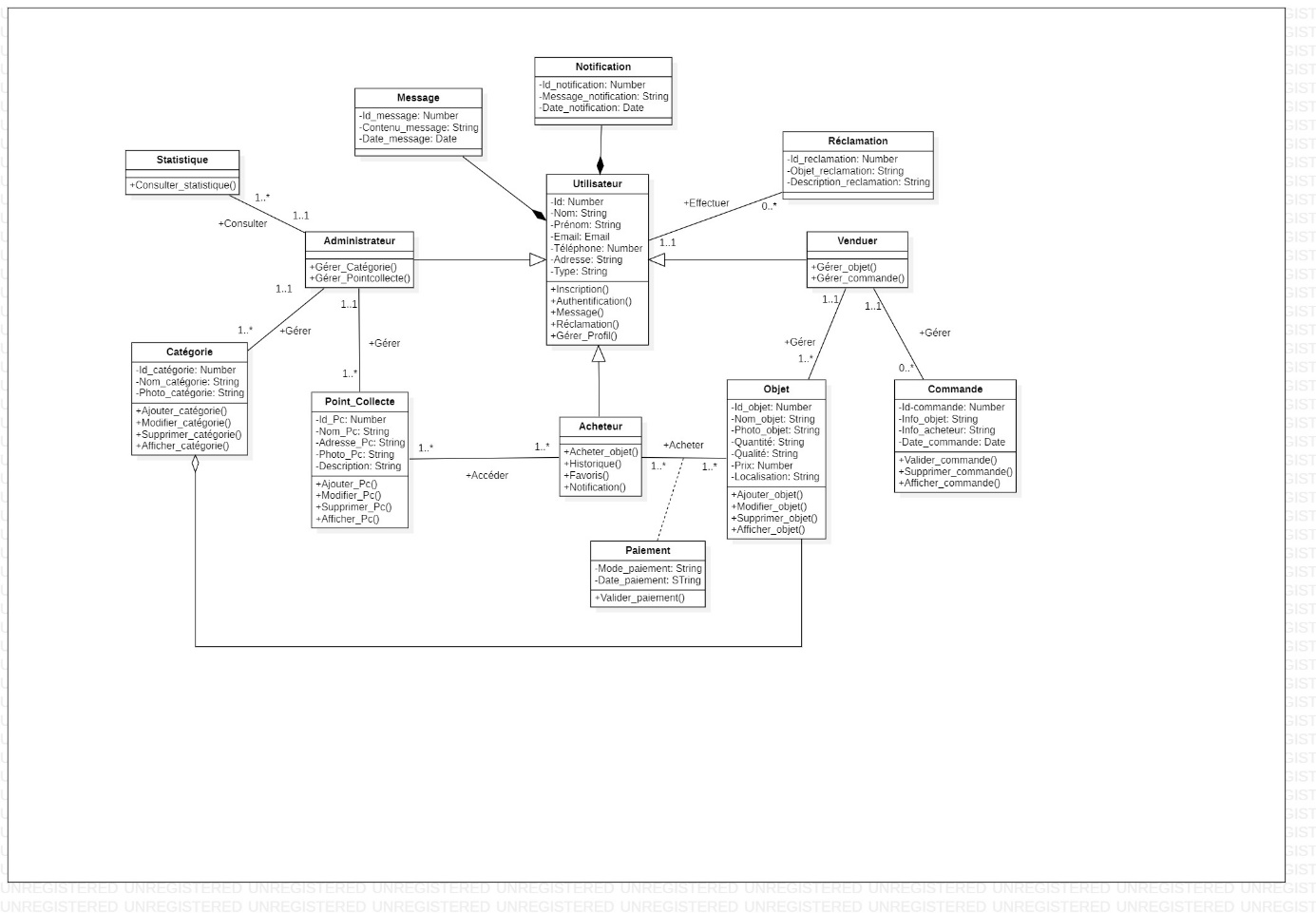


Figure \*\* : Diagramme de classes du projet

* 1. **Planification et Structuration**

La planification de projet consiste à structurer les tâches à effectuer sur une période donnée. Elle vise à déterminer la meilleure façon de planifier toutes les tâches, à estimer les charges et à allouer les ressources nécessaires pour les accomplir. Cette section comprendra d'abord le tableau de backlog du produit, qui énumère les éléments de travail prioritaires à accomplir. Ensuite, nous présenterons les rôles de chaque membre de l'équipe dans la méthodologie Scrum.

**3.4.1. Définition des rôles dans SCRUM**

Le but de cette section a pour but est de déterminer et de planifier les tâches du projet et d'estimer leur charge. Les objectifs du planning sont les suivants :

- Déterminer si les objectifs sont atteints ou dépassés.

- Suivre l'avancement du projet et assurer la communication.

- Allouer les ressources nécessaires pour chaque tâche.

|  |  |
| --- | --- |
| Le propriétaire du produit (Product Owner) | Amira Riahi |
| Le directeur du produit (SCRUM Master) | Abdel Monam Kouka |
| Le membre de l'équipe (SCRUM Team) | Khouloud Chrif  Rania Yousfi |

Tableau \*\*: Les rôles dans SCRUM

**3.4.2. Backlog du produit**

Un backlog produit est l'élément le plus important de Scrum puisqu'il représente l'ensemble des caractéristiques fonctionnelles ou techniques qui constituent le produit souhaité. Les caractéristiques fonctionnelles sont appelées des histoires utilisateur (user story) qui décrivent les interactions de l'utilisateur avec l'application et les caractéristiques techniques sont appelées des histoires techniques (technical story) [ref].

En outre, le backlog du produit dans le tableau ci-dessous, montre la liste des fonctionnalités qui doivent être implémentées dans notre application. Le choix des priorités dans cette section est basé sur la dépendance entre les fonctionnalités de l'application.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **User Story** | **Estimations** | **Descriptions** | **Priorités** |
| 1 | S'inscrire |  | En tant qu'utilisateur, je peux créer un compte. | Haute |
| 2 | S’authentifier |  | En tant qu’administrateur, je peux me connecter à l’application. | Haute |
| 3 | Gérer le profil |  | En tant qu’utilisateur, je peux faire la mise à jour de mes informations personnelles. | Haute |
| 4 | Envoyer une réclamation |  | En tant qu’utilisateur, je peux signaler une réclamation. | Faible |
| 5 | Discuter en temps réel |  | En tant qu’utilisateur, je peux discuter en temps réel. | Faible |
| 6 | Gérer les objets à vendre |  | En tant que vendeur, je peux ajouter ou modifier ou supprimer un objet. | Haute |
| 7 | Gérer les commandes |  | En tant que vendeur, je peux valider ou supprimer une commande. | Haute |
| 8 | Consulter les points de collecte |  | En tant qu'acheteur, je peux avoir des informations sur les différents points de collecte. | Haute |
| 9 | Consulter les objets disponibles |  | En tant qu'acheteur, je peux consulter la liste de tous les objets disponibles selon leurs catégories. | Haute |
| 10 | Acheter un objet |  | En tant qu'acheteur, je peux acheter un objet. | Haute |
| 11 | Gérer l’historique d’achat |  | En tant qu'acheteur, je peux consulter ou supprimer mes achats. | Moyenne |
| 12 | Gérer la liste des favoris |  | En tant qu'acheteur, je peux consulter ou ajouter ou supprimer mes favoris. | Faible |
| 13 | Recevoir notification des objets les plus proches |  | En tant qu' acheteur, je peux visualiser les notifications reçues. | Faible |
| 14 | Gérer la liste des catégories d’objets |  | En tant qu'administrateur, je peux ajouter ou modifier ou supprimer une catégorie d’objets. | Haute |
| 15 | Gérer la liste des points de collecte |  | En tant qu'administrateur, je peux ajouter ou modifier ou supprimer un point de collecte. | Haute |
| 16 | Consulter les statistiques |  | En tant qu'administrateur, je peux accéder aux statistiques. | Moyenne |

Tableau \*\*: Backlog du produit

Le backlog du produit dans le tableau (\*\*), présente la liste des fonctionnalités à implémenter dans notre application mobile. Chaque fonctionnalité est décrite par une histoire utilisateur (User Story) qui se voit attribuer un rang en fonction de sa priorité.

**3.4.3. Planification des sprints**

L'objectif de la planification de sprint est de déterminer ce qui peut être livré durant le sprint et comment y parvenir. Cet événement est crucial dans la méthode de travail Scrum [ref]. Il vise à préparer le planning de travail, à identifier le backlog des sprints, et à choisir la durée des sprints, qui peut varier en fonction de la complexité du projet et de la taille de l'équipe.

- Le premier livrable comprend les quatre premières fonctionnalités de base de l'application de recyclage, qui se concentrent sur la gestion de profil, la gestion des objets à vendre et l'inscription.

- Le deuxième livrable met en avant les fonctionnalités de la partie acheteur, qui concernent la consultation des points de collecte, la gestion des commandes, la consultation la disponibilité des objets et le système d'achat.

- Le livrable final comprend la prise en charge de la fonctionnalité de gestion des favoris ainsi que la mise en place des fonctionnalités administratives de notre application. Cela inclut la gestion des listes de points de collecte, d’historiques d’achats et des catégories d'objets, ainsi que la possibilité de consulter le tableau de bord pour suivre les actions d'achat.

**Conclusion**

Ce chapitre a porté sur l'identification des acteurs impliqués dans notre projet ainsi que sur les exigences fonctionnelles et non-fonctionnelles. Nous avons également étudié les aspects techniques liés au projet et établi notre plan de travail.

Dans le chapitre suivant, nous aborderons l'étude conceptuelle du premier et du deuxième sprints de notre projet. Nous avons ainsi préparé notre plan de travail en termes d'organisation des sprints dans ce chapitre.