

**Republique Tunisienne**

Ministere de l'Enseignement Superieur et de la Recherche Scientifique

**Direction Generale des Etudes Technologiques**



**Institut Supérieur des Etudes Technologiques de  
Rades**

Departement : Technologies de l'Informatique

---

## **Rapport de Projet d'Integration**

### **Conception et Realisation d'une Plateforme E-Commerce Intelligente pour Pieces Detachees Automobiles**

*avec Intelligence Artificielle et Systeme de Livraison Integre*

---

**Realise par :**

Gharsallah Islem

Ben Jemaa Mohamed Malek

Hammi Youssef

**Encadre par :**

Mme. Lamia Mansouri

**Annee Universitaire : 2024 - 2025**

# Dédicace

*À nos chers parents,  
qui nous ont soutenus tout au long de notre parcours,  
et qui ont fait de nous ce que nous sommes aujourd'hui.*

*À nos familles,  
pour leur amour inconditionnel et leur encouragement constant.*

*À nos amis,  
pour les moments partagés et le soutien moral.  
À tous ceux qui ont contribué, de près ou de loin,  
à la réalisation de ce projet.*

**Nous vous dédions ce travail.**

# Remerciements

Au terme de ce projet de fin d'études, nous tenons à exprimer notre profonde gratitude à toutes les personnes qui ont contribué à sa réalisation.

Nous adressons nos sincères remerciements à notre encadrante, **Mme. Lamia Mansouri**, pour sa disponibilité, ses conseils précieux et son accompagnement tout au long de ce projet. Son expertise et ses orientations ont été déterminantes pour la réussite de ce travail.

Nous remercions également l'ensemble du corps enseignant du **Département Technologies de l'Informatique** de l'**ISSET de Radès** pour la formation de qualité qu'ils nous ont dispensée durant ces années d'études.

Nos remerciements vont aussi aux membres du jury qui ont accepté d'évaluer ce travail et de nous faire bénéficier de leurs remarques constructives.

Enfin, nous exprimons notre reconnaissance à nos familles et amis pour leur soutien moral et leurs encouragements constants qui nous ont permis de mener à bien ce projet.

*Gharsallah Islem*  
*Ben Jemaa Mohamed Malek*  
*Hammi Youssef*

# Resume

Ce projet de fin d'études porte sur la conception et la réalisation d'une plateforme e-commerce intelligente dédiée à la vente de pièces détachées automobiles. L'objectif principal est de faciliter l'identification et l'achat de pièces auto grâce à l'intégration de technologies d'Intelligence Artificielle.

La plateforme développée offre une solution complète comprenant :

- Une application web responsive (Angular 18)
- Une application mobile native (Kotlin Android)
- Un panneau d'administration pour la gestion des produits et des commandes
- Un système de paiement sécurisé via Stripe
- Un module de reconnaissance d'images basé sur les réseaux de neurones convolutifs (CNN)
- Un système de recommandation personnalisée
- Un chat de support en temps réel
- Un système de suivi de livraison avec cartographie

Le développement a été réalisé en suivant la méthodologie Scrum, organisée en six sprints couvrant l'authentification, la gestion des produits, les commandes, les paiements, l'intelligence artificielle et le système de livraison.

Les technologies utilisées incluent Angular 18 pour le frontend web, Kotlin pour l'application mobile Android, Spring Boot 3 pour le backend, PostgreSQL pour la base de données, FastAPI et TensorFlow pour le module d'IA, et Leaflet avec OpenRouteService pour la cartographie.

**Mots-cles :** E-commerce, Pièces automobiles, Intelligence Artificielle, Reconnaissance d'images, Spring Boot, Angular, Kotlin, Système de recommandation, Livraison en temps réel.

# Table des matières

Dédicace	i
Remerciements	ii
Resume	iii
<b>1 Introduction Générale</b>	<b>2</b>
1.1 Contexte du Projet . . . . .	2
1.2 Problématique . . . . .	2
1.3 Objectifs du Projet . . . . .	3
1.3.1 Objectifs Fonctionnels . . . . .	3
1.3.2 Objectifs Techniques . . . . .	3
1.4 Périmètre et Limites . . . . .	4
1.4.1 Périmètre du Projet . . . . .	4
1.4.2 Limites . . . . .	4
1.5 Organisation du Rapport . . . . .	4
<b>2 État de l'Art</b>	<b>5</b>
2.1 Introduction . . . . .	5
2.2 Étude des Plateformes E-Commerce Existantes . . . . .	5
2.2.1 Plateformes Internationales . . . . .	5
2.2.2 Plateformes Régionales et Tunisiennes . . . . .	6
2.2.3 Tableau Comparatif . . . . .	6
2.3 Intelligence Artificielle dans l'E-Commerce . . . . .	6
2.3.1 Reconnaissance d'Images . . . . .	6
2.3.2 Systèmes de Recommandation . . . . .	7
2.4 Technologies et Frameworks . . . . .	7
2.4.1 Frontend : Angular . . . . .	7
2.4.2 Mobile : Kotlin Android . . . . .	8
2.4.3 Backend : Spring Boot . . . . .	8
2.4.4 Base de Données : PostgreSQL . . . . .	8
2.4.5 Module IA : FastAPI et TensorFlow . . . . .	9
2.4.6 Autres Technologies . . . . .	9
2.5 Conclusion . . . . .	9

---

<b>3</b>	<b>Méthodologie et Gestion de Projet</b>	<b>10</b>
3.1	Introduction . . . . .	10
3.2	Méthodologie Scrum . . . . .	10
3.2.1	Présentation de Scrum . . . . .	10
3.2.2	Rôles Scrum . . . . .	10
3.2.3	Cérémonies Scrum . . . . .	10
3.2.4	Artefacts Scrum . . . . .	11
3.3	Organisation des Sprints . . . . .	11
3.4	Environnement de Développement . . . . .	11
3.4.1	Outils de Développement . . . . .	11
3.4.2	Configuration des Environnements . . . . .	11
3.5	Architecture Globale du Système . . . . .	12
3.5.1	Architecture en Couches . . . . .	12
3.5.2	Diagramme de Déploiement . . . . .	12
3.6	Stack Technologique Détaillée . . . . .	12
3.6.1	Frontend . . . . .	12
3.6.2	Backend . . . . .	13
3.6.3	Module IA . . . . .	13
3.7	Gestion de la Qualité . . . . .	13
3.7.1	Standards de Code . . . . .	13
3.7.2	Revue de Code . . . . .	14
3.7.3	Tests . . . . .	14
3.8	Conclusion . . . . .	14
<b>4</b>	<b>Sprint 1 : Fondation et Authentification</b>	<b>15</b>
4.1	Presentation du Sprint . . . . .	15
4.1.1	Objectifs . . . . .	15
4.1.2	Objectifs Détaillés . . . . .	15
4.2	Sprint Backlog . . . . .	15
4.3	Analyse et Conception . . . . .	15
4.3.1	Diagramme de Cas d'Utilisation . . . . .	16
4.3.2	Diagramme de Classes . . . . .	17
4.3.3	Diagramme de Sequence . . . . .	19
4.4	Realisation . . . . .	20
4.4.1	Architecture d'Authentification . . . . .	20
4.4.2	Gestion des Roles . . . . .	20
4.5	Bilan du Sprint . . . . .	20

---

<b>5</b>	<b>Sprint 2 : Gestion des Produits et Catalogue</b>	<b>21</b>
5.1	Presentation du Sprint . . . . .	21
5.1.1	Objectifs . . . . .	21
5.2	Sprint Backlog . . . . .	21
5.3	Analyse et Conception . . . . .	21
5.3.1	Diagramme de Cas d'Utilisation . . . . .	21
5.3.2	Diagramme de Classes . . . . .	23
5.3.3	Diagramme de Sequence . . . . .	23
5.4	Realisation . . . . .	24
5.4.1	Architecture du Catalogue . . . . .	24
5.5	Bilan du Sprint . . . . .	24
<b>6</b>	<b>Sprint 3 : Panier et Gestion des Commandes</b>	<b>25</b>
6.1	Presentation du Sprint . . . . .	25
6.1.1	Objectifs . . . . .	25
6.2	Sprint Backlog . . . . .	25
6.3	Analyse et Conception . . . . .	25
6.3.1	Diagramme de Cas d'Utilisation . . . . .	25
6.3.2	Diagramme de Classes . . . . .	27
6.3.3	Diagramme de Sequence . . . . .	29
6.4	Realisation . . . . .	29
6.4.1	Cycle de Vie des Commandes . . . . .	29
6.5	Bilan du Sprint . . . . .	30
<b>7</b>	<b>Sprint 4 : Paiement Stripe et Gestion d'Inventaire</b>	<b>31</b>
7.1	Presentation du Sprint . . . . .	31
7.1.1	Objectifs . . . . .	31
7.2	Sprint Backlog . . . . .	31
7.3	Analyse et Conception . . . . .	32
7.3.1	Diagramme de Cas d'Utilisation . . . . .	32
7.3.2	Diagramme de Classes . . . . .	33
7.3.3	Diagramme de Sequence . . . . .	34
7.4	Realisation . . . . .	35
7.4.1	Integration Stripe . . . . .	35
7.5	Bilan du Sprint . . . . .	35
<b>8</b>	<b>Sprint 5 : Module d'Intelligence Artificielle</b>	<b>36</b>
8.1	Presentation du Sprint . . . . .	36
8.1.1	Objectifs . . . . .	36
8.2	Sprint Backlog . . . . .	36

---

8.3	Analyse et Conception . . . . .	37
8.3.1	Diagramme de Cas d'Utilisation . . . . .	37
8.3.2	Diagramme de Classes - Backend . . . . .	38
8.3.3	Diagramme de Classes - Module Python . . . . .	39
8.3.4	Diagramme de Sequence . . . . .	40
8.4	Realisation . . . . .	40
8.4.1	Modele de Classification . . . . .	40
8.5	Bilan du Sprint . . . . .	40
<b>9</b>	<b>Sprint 6 : Chat Support, Livraison et Validation Finale</b>	<b>42</b>
9.1	Presentation du Sprint . . . . .	42
9.1.1	Objectifs . . . . .	42
9.2	Sprint Backlog . . . . .	42
9.3	Analyse et Conception . . . . .	43
9.3.1	Diagramme de Cas d'Utilisation . . . . .	43
9.3.2	Diagramme de Classes . . . . .	44
9.3.3	Diagramme de Sequence . . . . .	46
9.4	Realisation . . . . .	46
9.4.1	Chat en Temps Reel . . . . .	46
9.4.2	Suivi de Livraison . . . . .	46
9.5	Validation Finale . . . . .	46
9.6	Bilan du Sprint . . . . .	47
<b>10</b>	<b>Tests et Validation</b>	<b>48</b>
10.1	Introduction . . . . .	48
10.2	Strategie de Tests . . . . .	48
10.2.1	Pyramide des Tests . . . . .	48
10.3	Tests Unitaires . . . . .	48
10.3.1	Backend - JUnit et Mockito . . . . .	48
10.3.2	Frontend - Jasmine et Karma . . . . .	49
10.3.3	Module IA - Pytest . . . . .	49
10.4	Tests d'Integration . . . . .	49
10.4.1	Tests API . . . . .	49
10.4.2	Tests avec Postman . . . . .	49
10.5	Tests d'Acceptation Utilisateur . . . . .	50
10.5.1	Scenarios de Test . . . . .	50
10.6	Couverture de Code . . . . .	50
10.7	Validation des Fonctionnalites . . . . .	50
10.8	Conclusion . . . . .	50



---

<b>11 Conclusion et Perspectives</b>	<b>52</b>
11.1 Synthèse des Réalisations . . . . .	52
11.1.1 Objectifs Atteints . . . . .	52
11.1.2 Métriques du Projet . . . . .	52
11.1.3 Fonctionnalités Clés Développées . . . . .	52
11.2 Défis Techniques Surmontés . . . . .	53
11.2.1 Défis Majeurs . . . . .	53
11.3 Limitations Actuelles . . . . .	53
11.4 Perspectives d'Amélioration . . . . .	54
11.4.1 Court Terme (3-6 mois) . . . . .	54
11.4.2 Moyen Terme (6-12 mois) . . . . .	54
11.4.3 Long Terme (12+ mois) . . . . .	55
11.5 Compétences Acquisées . . . . .	55
11.5.1 Compétences Techniques . . . . .	55
11.5.2 Compétences Méthodologiques . . . . .	56
11.5.3 Compétences Transversales . . . . .	56
11.6 Conclusion Générale . . . . .	56
 <b>A Annexe A : Schema de Base de Donnees</b>	 <b>57</b>
A.1 Diagramme Entite-Relation . . . . .	57
A.2 Description des Tables Principales . . . . .	57
A.2.1 Table Users . . . . .	57
A.2.2 Table Products . . . . .	57
A.2.3 Table Orders . . . . .	58
 <b>B Annexe B : Guide d'Installation</b>	 <b>59</b>
B.1 Prerequis . . . . .	59
B.2 Installation du Backend . . . . .	59
B.3 Installation du Frontend . . . . .	59
B.4 Installation du Module IA . . . . .	59

# Table des figures

4.1	Diagramme de cas d'utilisation - Module Authentification . . . . .	16
4.2	Diagramme de classes - Entites User et Role . . . . .	18
4.3	Diagramme de sequence - Processus de connexion JWT . . . . .	19
5.1	Diagramme de cas d'utilisation - Gestion des Produits . . . . .	22
5.2	Diagramme de classes - Entites Product, Category, Brand et Vehicle . . . . .	23
5.3	Diagramme de sequence - Recherche et filtrage de produits . . . . .	24
6.1	Diagramme de cas d'utilisation - Panier et Commandes . . . . .	26
6.2	Diagramme de classes - Entites Cart, Order, OrderItem et Product . . . . .	28
6.3	Diagramme de sequence - Creation d'une commande . . . . .	29
7.1	Diagramme de cas d'utilisation - Paiement et Inventaire . . . . .	32
7.2	Diagramme de classes - Payment, StockMovement, Supplier, PurchaseOrder et Product . . . . .	33
7.3	Diagramme de sequence - Paiement avec Stripe Checkout . . . . .	34
8.1	Diagramme de cas d'utilisation - Module IA . . . . .	37
8.2	Diagramme de classes - Recommendation, UserActivity et entites associees . . . . .	38
8.3	Diagramme de classes - Module IA Python (FastAPI, EfficientNetB0) . . . . .	39
8.4	Diagramme de sequence - Reconnaissance d'image IA . . . . .	40
9.1	Diagramme de cas d'utilisation - Chat et Livraison . . . . .	43
9.2	Diagramme de classes - Conversation, Message, Delivery, Driver et entites associees . . . . .	44
9.3	Diagramme de sequence - Suivi de livraison en temps reel . . . . .	46

# Liste des tableaux

2.1	Comparaison des plateformes existantes . . . . .	6
2.2	Technologies complémentaires utilisées . . . . .	9
3.1	Planification des sprints . . . . .	11
3.2	Outils de développement utilisés . . . . .	12
3.3	Technologies Frontend . . . . .	13
3.4	Technologies Backend . . . . .	13
3.5	Technologies Module IA . . . . .	13
4.1	Fiche du Sprint 1 . . . . .	15
4.2	Sprint Backlog - Sprint 1 . . . . .	16
4.3	Roles et permissions . . . . .	20
4.4	Bilan Sprint 1 . . . . .	20
5.1	Fiche du Sprint 2 . . . . .	21
5.2	Sprint Backlog - Sprint 2 . . . . .	22
5.3	Bilan Sprint 2 . . . . .	24
6.1	Fiche du Sprint 3 . . . . .	25
6.2	Sprint Backlog - Sprint 3 . . . . .	26
6.3	Statuts des commandes . . . . .	30
6.4	Bilan Sprint 3 . . . . .	30
7.1	Fiche du Sprint 4 . . . . .	31
7.2	Sprint Backlog - Sprint 4 . . . . .	31
7.3	Bilan Sprint 4 . . . . .	35
8.1	Fiche du Sprint 5 . . . . .	36
8.2	Sprint Backlog - Sprint 5 . . . . .	36
8.3	Specifications du modele IA . . . . .	40
8.4	Bilan Sprint 5 . . . . .	41
9.1	Fiche du Sprint 6 . . . . .	42
9.2	Sprint Backlog - Sprint 6 . . . . .	42
9.3	Statuts de livraison . . . . .	47
9.4	Scenarios de validation . . . . .	47
9.5	Bilan Sprint 6 . . . . .	47

---

10.1	Resume des tests API . . . . .	49
10.2	Scenarios de test utilisateur . . . . .	50
10.3	Couverture de code par module . . . . .	50
10.4	Validation par plateforme . . . . .	51
11.1	Bilan des objectifs . . . . .	52
11.2	Statistiques du projet . . . . .	53
11.3	Défis techniques rencontrés . . . . .	54
A.1	Structure de la table users . . . . .	57
A.2	Structure de la table products . . . . .	57
A.3	Structure de la table orders . . . . .	58

# Chapitre 1

## Introduction Générale

### 1.1 Contexte du Projet

Le secteur de l'e-commerce connaît une croissance exponentielle à l'échelle mondiale, transformant profondément les habitudes de consommation. En Tunisie, comme dans de nombreux pays, cette transformation numérique s'accélère, offrant de nouvelles opportunités aux entreprises et aux consommateurs.

Le marché des pièces détachées automobiles représente un segment particulièrement dynamique de l'économie. Avec un parc automobile en constante évolution et une demande croissante de pièces de rechange, ce secteur fait face à des défis majeurs : la diversité des modèles de véhicules, la complexité de l'identification des pièces compatibles, et la nécessité de garantir la qualité des produits.

Traditionnellement, les automobilistes doivent se rendre chez des garagistes ou des revendeurs spécialisés pour identifier et acheter les pièces dont ils ont besoin. Ce processus est souvent long, fastidieux et peut conduire à des erreurs coûteuses lorsque la pièce commandée n'est pas compatible avec le véhicule.

L'émergence de l'Intelligence Artificielle (IA) et des technologies de reconnaissance d'images ouvre de nouvelles perspectives pour résoudre ces problématiques. En permettant aux utilisateurs d'identifier des pièces automobiles simplement en prenant une photo, ces technologies peuvent révolutionner l'expérience d'achat dans ce domaine.

### 1.2 Problématique

Les acheteurs de pièces détachées automobiles sont confrontés à plusieurs difficultés majeures :

- **Difficulté d'identification** : La plupart des automobilistes ne connaissent pas le nom exact des pièces dont ils ont besoin, ce qui complique leur recherche.
- **Problème de compatibilité** : Chaque véhicule a des spécifications propres, et une pièce qui semble identique peut ne pas être compatible avec tous les modèles.
- **Fragmentation du marché** : Les pièces sont dispersées entre de nombreux fournisseurs, rendant la comparaison des prix et de la disponibilité difficile.

- **Manque de conseil** : Les plateformes e-commerce classiques ne fournissent pas de recommandations personnalisées basées sur les symptômes ou les besoins spécifiques de l'utilisateur.
- **Suivi de livraison limité** : Le suivi des commandes est souvent rudimentaire, sans visibilité en temps réel sur la position du livreur.

Face à ces constats, la question qui se pose est la suivante : *Comment concevoir une plateforme e-commerce intelligente capable d'assister les utilisateurs dans l'identification, le choix et l'achat de pièces détachées automobiles, tout en offrant une expérience utilisateur optimale ?*

## 1.3 Objectifs du Projet

Notre projet vise à développer une plateforme e-commerce complète et intelligente pour les pièces détachées automobiles. Les objectifs principaux sont :

### 1.3.1 Objectifs Fonctionnels

1. **Système d'authentification sécurisé** : Permettre aux utilisateurs de créer un compte, se connecter et gérer leur profil de manière sécurisée.
2. **Catalogue de produits complet** : Offrir un catalogue riche de pièces automobiles avec des fonctionnalités de recherche avancée et de filtrage.
3. **Gestion du panier et des commandes** : Permettre aux utilisateurs d'ajouter des produits au panier, de passer des commandes et de suivre leur historique.
4. **Paiement sécurisé** : Intégrer un système de paiement en ligne sécurisé via Stripe.
5. **Reconnaissance d'images par IA** : Développer un module de reconnaissance d'images capable d'identifier les pièces automobiles à partir de photos.
6. **Système de recommandation** : Proposer des recommandations personnalisées basées sur le comportement de l'utilisateur et l'analyse des symptômes.
7. **Chat de support** : Mettre en place un système de chat en temps réel pour l'assistance client.
8. **Suivi de livraison** : Offrir un suivi en temps réel des livraisons avec visualisation sur carte.

### 1.3.2 Objectifs Techniques

1. Adopter une architecture moderne et scalable basée sur les microservices.
2. Utiliser les meilleures pratiques de développement (Clean Code, Design Patterns).

3. Assurer la sécurité de l'application (authentification JWT, HTTPS, validation des données).
4. Développer une interface utilisateur responsive et accessible (Web et Mobile).
5. Intégrer des services tiers (Stripe, OpenRouteService) de manière robuste.

## 1.4 Périmètre et Limites

### 1.4.1 Périmètre du Projet

Le projet couvre les aspects suivants :

- Application web complète avec panneau d'administration
- Application mobile native Android (Kotlin) pour les utilisateurs
- Backend API RESTful
- Module d'Intelligence Artificielle pour la reconnaissance d'images
- Système de livraison simulé avec suivi en temps réel

### 1.4.2 Limites

Certaines fonctionnalités ne sont pas incluses dans le périmètre initial :

- Les paiements réels sont simulés via le mode test de Stripe
- Le système de livraison utilise une simulation de mouvement du livreur
- L'application mobile n'inclut pas le panneau d'administration
- Le modèle d'IA est entraîné sur un jeu de données limité (50 classes de pièces)

## 1.5 Organisation du Rapport

Ce rapport est organisé en plusieurs chapitres :

- **Chapitre 2 - État de l'Art** : Présente une étude des solutions existantes et des technologies utilisées.
- **Chapitre 3 - Méthodologie** : Décrit la méthodologie Scrum adoptée et l'environnement de développement.
- **Chapitres 4 à 9 - Sprints** : Détaillent le développement de chaque sprint avec les user stories, les fonctionnalités implémentées et les diagrammes UML.
- **Chapitre 10 - Tests et Validation** : Présente la stratégie de tests et les résultats de validation.
- **Chapitre 11 - Conclusion** : Synthétise les réalisations et propose des perspectives d'amélioration.

# Chapitre 2

## État de l'Art

### 2.1 Introduction

Ce chapitre présente une étude approfondie des solutions existantes dans le domaine de l'e-commerce automobile, ainsi qu'une analyse des technologies qui seront utilisées dans notre projet. Cette étude nous permettra de positionner notre solution et de justifier nos choix technologiques.

### 2.2 Étude des Plateformes E-Commerce Existantes

#### 2.2.1 Plateformes Internationales

##### Amazon Auto Parts

Amazon propose une section dédiée aux pièces automobiles avec plusieurs fonctionnalités intéressantes :

- Recherche par compatibilité véhicule (marque, modèle, année)
- Large catalogue de produits avec avis clients
- Système de recommandation basé sur l'historique d'achat
- Livraison rapide via Amazon Prime

**Limites :** Pas de reconnaissance d'images, interface générique non spécialisée.

##### RockAuto

RockAuto est une référence dans le domaine des pièces automobiles en ligne :

- Spécialisation dans les pièces automobiles
- Catalogue très complet avec diagrammes techniques
- Prix compétitifs avec multiple fournisseurs
- Recherche avancée par numéro de pièce OEM

**Limites :** Interface datée, pas d'IA, pas d'application mobile native.



## eBay Motors

eBay Motors offre une marketplace pour les pièces automobiles :

- Pièces neuves et d'occasion
- Système d'enchères et d'achat immédiat
- Garantie acheteur
- Vendeurs particuliers et professionnels

**Limites :** Qualité variable des vendeurs, pas de reconnaissance d'images.

### 2.2.2 Plateformes Régionales et Tunisiennes

En Tunisie, le marché des pièces automobiles en ligne est encore émergent. Les solutions existantes se limitent généralement à :

- Des pages Facebook de revendeurs
- Des sites vitrine sans fonctionnalités e-commerce complètes
- Des applications basiques sans intelligence artificielle

### 2.2.3 Tableau Comparatif

TABLE 2.1 – Comparaison des plateformes existantes

Fonctionnalité	Amazon	RockAuto	eBay	Local	Notre Solution
Catalogue complet	✓	✓	✓	–	✓
Application mobile	✓	–	✓	–	✓
Reconnaissance IA	–	–	–	–	✓
Recommandations IA	✓	–	–	–	✓
Chat support	✓	–	✓	–	✓
Suivi temps réel	✓	–	–	–	✓

## 2.3 Intelligence Artificielle dans l'E-Commerce

### 2.3.1 Reconnaissance d'Images

La reconnaissance d'images est devenue un outil puissant dans l'e-commerce moderne. Elle permet :

- **Recherche visuelle :** L'utilisateur peut prendre une photo d'un produit pour le rechercher.
- **Identification de pièces :** Particulièrement utile dans l'automobile où les pièces sont difficiles à nommer.
- **Contrôle qualité :** Détection automatique des défauts sur les produits.

## Réseaux de Neurones Convolutifs (CNN)

Les CNN sont la technologie de référence pour la classification d'images. Les architectures les plus utilisées incluent :

- **VGG16/VGG19** : Architecture simple mais efficace, souvent utilisée comme base-line.
- **ResNet** : Introduit les connexions résiduelles pour les réseaux profonds.
- **EfficientNet** : Optimise le compromis entre précision et efficacité computationnelle.
- **MobileNet** : Conçu pour les appareils mobiles avec ressources limitées.

Pour notre projet, nous avons choisi **EfficientNetB0** car il offre un excellent équilibre entre précision et taille du modèle, ce qui est crucial pour un déploiement en production.

### 2.3.2 Systèmes de Recommandation

Les systèmes de recommandation sont essentiels pour personnaliser l'expérience utilisateur. On distingue trois approches principales :

1. **Filtrage collaboratif** : Recommande des produits basés sur les préférences d'utilisateurs similaires.
2. **Filtrage basé sur le contenu** : Recommande des produits similaires à ceux déjà consultés ou achetés.
3. **Approche hybride** : Combine les deux méthodes pour de meilleures recommandations.

Notre système utilisera une approche hybride, combinant l'analyse du comportement utilisateur avec les caractéristiques des produits.

## 2.4 Technologies et Frameworks

### 2.4.1 Frontend : Angular

Angular est un framework développé par Google pour la création d'applications web SPA (Single Page Application).

#### Avantages :

- Architecture modulaire et maintenable
- TypeScript pour un typage fort
- Système de composants réutilisables
- Injection de dépendances intégrée
- Écosystème riche (Angular Material, RxJS)

### 2.4.2 Mobile : Kotlin Android

Kotlin est le langage officiel recommandé par Google pour le développement d'applications Android natives.

**Avantages :**

- Langage moderne et concis
- Interopérabilité complète avec Java
- Null safety intégré
- Coroutines pour la programmation asynchrone
- Support officiel de Google et JetBrains

Notre application mobile Android utilise Kotlin avec les composants :

- Retrofit pour les appels API REST vers le backend Spring Boot
- Coroutines pour la gestion asynchrone
- View Binding pour la liaison des vues
- Material Design 3 pour l'interface utilisateur

### 2.4.3 Backend : Spring Boot

Spring Boot est un framework Java qui simplifie le développement d'applications d'entreprise.

**Avantages :**

- Configuration automatique
- Sécurité intégrée (Spring Security)
- Support JPA/Hibernate pour l'ORM
- Écosystème mature et documenté
- WebSocket pour la communication temps réel

### 2.4.4 Base de Données : PostgreSQL

PostgreSQL est un système de gestion de base de données relationnelle open source.

**Avantages :**

- Fiabilité et robustesse
- Support des types de données complexes (JSON, arrays)
- Performances optimisées pour les grandes bases
- Conformité ACID

### 2.4.5 Module IA : FastAPI et TensorFlow

#### FastAPI :

- Framework Python moderne et rapide
- Documentation automatique (OpenAPI/Swagger)
- Support asynchrone natif
- Validation automatique des données

#### TensorFlow/Keras :

- Bibliothèque de deep learning de Google
- Support GPU pour l'entraînement
- Large communauté et documentation
- Modèles pré-entraînés disponibles

### 2.4.6 Autres Technologies

TABLE 2.2 – Technologies complémentaires utilisées

Composant	Technologie	Utilisation
Païement	Stripe	Transactions sécurisées
Cartographie	Leaflet + OpenRouteService	Suivi de livraison
Temps réel	WebSocket (STOMP)	Chat, notifications
Authentification	JWT	Tokens sécurisés
Conteneurisation	Docker	Déploiement
Versioning	Git/GitHub	Gestion du code source

## 2.5 Conclusion

Cette étude de l'état de l'art nous a permis d'identifier les lacunes des solutions existantes et de définir les caractéristiques différenciatrices de notre plateforme. Notre solution se distingue par l'intégration de l'intelligence artificielle pour la reconnaissance d'images et les recommandations, ainsi que par un système de livraison avec suivi en temps réel.

# Chapitre 3

## Méthodologie et Gestion de Projet

### 3.1 Introduction

Ce chapitre présente la méthodologie de développement adoptée pour notre projet, ainsi que les outils et l'environnement de travail utilisés. L'organisation rigoureuse du projet est essentielle pour garantir la qualité des livrables et le respect des délais.

### 3.2 Méthodologie Scrum

#### 3.2.1 Présentation de Scrum

Scrum est un cadre de travail agile qui permet de gérer des projets complexes de manière itérative et incrémentale. Il est particulièrement adapté aux projets de développement logiciel où les exigences peuvent évoluer.

*scrum\_frameworkCadre detravailScrum*

#### 3.2.2 Rôles Scrum

Notre équipe est organisée selon les rôles Scrum suivants :

- **Product Owner** : Mme. Lamia Mansouri (encadrante) - Définit les priorités et valide les fonctionnalités.
- **Scrum Master** : Gharsallah Islem - Facilite les cérémonies Scrum et résout les obstacles.
- **Équipe de développement** :
  - Gharsallah Islem - Backend, IA
  - Ben Jemaa Mohamed Malek - Frontend Web, Mobile
  - Hammi Youssef - Base de données, Tests

#### 3.2.3 Cérémonies Scrum

Nous avons mis en place les cérémonies suivantes :

1. **Sprint Planning** : Réunion de planification au début de chaque sprint pour définir les objectifs et sélectionner les user stories.
2. **Daily Scrum** : Réunions quotidiennes de 15 minutes pour synchroniser l'équipe.
3. **Sprint Review** : Démonstration des fonctionnalités développées à la fin du sprint.
4. **Sprint Retrospective** : Analyse de ce qui a bien fonctionné et des axes d'amélioration.

### 3.2.4 Artefacts Scrum

- **Product Backlog** : Liste priorisée de toutes les fonctionnalités du projet.
- **Sprint Backlog** : Sous-ensemble du Product Backlog sélectionné pour le sprint en cours.
- **Increment** : Version potentiellement livrable du produit à la fin de chaque sprint.

## 3.3 Organisation des Sprints

Le projet est divisé en **6 sprints** de 2 semaines chacun :

TABLE 3.1 – Planification des sprints

Sprint	Thème	Objectifs principaux
1	Fondation & Authentification	Setup, Auth, Gestion utilisateurs
2	Gestion des Produits	Catalogue, Recherche, Admin produits
3	Panier & Commandes	Panier, Checkout, Gestion commandes
4	Paielement & Inventaire	Stripe, Gestion stocks, Réapprovisionnement
5	Module IA	Reconnaissance images, Recommandations
6	Chat, Livraison & Validation	Support, Suivi livraison, Tests finaux

*gantchartDiagrammedeGanttduprojet*

## 3.4 Environnement de Développement

### 3.4.1 Outils de Développement

### 3.4.2 Configuration des Environnements

Notre projet utilise trois environnements distincts :

1. **Développement** : Configuration locale pour le développement et les tests unitaires.
2. **Staging** : Environnement de pré-production pour les tests d'intégration.
3. **Production** : Environnement final déployé via Docker.

TABLE 3.2 – Outils de développement utilisés

Catégorie	Outil	Version
IDE Backend	IntelliJ IDEA	2024.x
IDE Frontend	Visual Studio Code	1.85+
IDE Python	PyCharm / VS Code	2024.x
SGBD	PostgreSQL	15.x
Admin BDD	pgAdmin	4.x
API Testing	Postman	10.x
Versioning	Git	2.x
Repository	GitHub	-
Conteneurs	Docker	24.x

## 3.5 Architecture Globale du Système

### 3.5.1 Architecture en Couches

Notre application suit une architecture en couches classique :  
*architecture\_globaleArchitectureglobaledusystème*

1. **Couche Présentation :**

- Application Web (Angular 18)
- Application Mobile (Kotlin Android native)

2. **Couche API :**

- Backend Spring Boot (REST API)
- Module IA FastAPI

3. **Couche Métier :**

- Services Spring Boot
- Logique de recommandation

4. **Couche Données :**

- PostgreSQL (base principale)
- Modèles TensorFlow (IA)

### 3.5.2 Diagramme de Déploiement

*diagramme\_deploiementDiagrammededéploiementUML*

## 3.6 Stack Technologique Détaillée

### 3.6.1 Frontend

TABLE 3.3 – Technologies Frontend

Technologie	Utilisation
Angular 18	Framework principal
TypeScript	Langage de programmation
Angular Material	Composants UI
RxJS	Programmation réactive
SCSS	Stylisation
Leaflet	Cartographie
Chart.js	Graphiques analytics

### 3.6.2 Backend

TABLE 3.4 – Technologies Backend

Technologie	Utilisation
Spring Boot 3	Framework principal
Java 17	Langage de programmation
Spring Security	Sécurité, JWT
Spring Data JPA	ORM, accès données
Hibernate	Implémentation JPA
WebSocket (STOMP)	Communication temps réel
Lombok	Réduction boilerplate

### 3.6.3 Module IA

TABLE 3.5 – Technologies Module IA

Technologie	Utilisation
Python 3.10	Langage de programmation
FastAPI	Framework API
TensorFlow 2.12	Deep learning
Keras	API haut niveau
NumPy	Calcul numérique
Pillow	Traitement images

## 3.7 Gestion de la Qualité

### 3.7.1 Standards de Code

Nous avons adopté les pratiques suivantes :

- **Clean Code** : Nommage explicite, fonctions courtes, commentaires pertinents
- **Design Patterns** : Repository, Service, Factory, Singleton



- 
- **SOLID Principles** : Respect des principes de conception orientée objet

### 3.7.2 Revue de Code

Chaque fonctionnalité fait l'objet d'une Pull Request sur GitHub, revue par au moins un autre membre de l'équipe avant fusion.

### 3.7.3 Tests

Notre stratégie de tests comprend :

- Tests unitaires (JUnit, Jasmine)
- Tests d'intégration (Spring Boot Test)
- Tests API (Postman)
- Tests manuels de validation

## 3.8 Conclusion

L'adoption de la méthodologie Scrum et la mise en place d'un environnement de développement structuré nous ont permis de gérer efficacement ce projet complexe. Les six sprints planifiés couvrent l'ensemble des fonctionnalités de la plateforme, de l'authentification jusqu'au système de livraison.

# Chapitre 4

## Sprint 1 : Fondation et Authentification

### 4.1 Presentation du Sprint

#### 4.1.1 Objectifs

Le premier sprint constitue la fondation de notre projet. Il vise à mettre en place l'infrastructure technique et le système d'authentification sécurisé.

TABLE 4.1 – Fiche du Sprint 1

Attribut	Valeur
Duree	2 semaines
Theme	Fondation et Authentification
Priorite	Haute

#### 4.1.2 Objectifs Detailles

1. Configuration de l'environnement de développement (Angular, Spring Boot, PostgreSQL)
2. Mise en place de l'architecture du projet selon le pattern MVC
3. Développement du système d'authentification (inscription, connexion, déconnexion)
4. Gestion des utilisateurs et des rôles (CLIENT, ADMIN, DRIVER, SUPPORT, SUPER\_ADMIN)
5. Sécurisation des API avec JWT (JSON Web Token)
6. Vérification d'email et réinitialisation de mot de passe

### 4.2 Sprint Backlog

### 4.3 Analyse et Conception

TABLE 4.2 – Sprint Backlog - Sprint 1

ID	User Story	Priorite	Points
US-1.1	En tant qu'utilisateur, je veux creer un compte avec mon email et mot de passe	Haute	5
US-1.2	En tant qu'utilisateur, je veux me connecter de maniere securisee	Haute	5
US-1.3	En tant qu'utilisateur, je veux reinitia-liser mon mot de passe oublie	Moyenne	3
US-1.4	En tant qu'admin, je veux gerer les roles des utilisateurs	Haute	5
US-1.5	En tant qu'utilisateur, je veux verifier mon email	Moyenne	3
US-1.6	En tant qu'utilisateur, je veux mettre a jour mon profil	Moyenne	3

### 4.3.1 Diagramme de Cas d'Utilisation

Le diagramme de cas d'utilisation suivant presente les fonctionnalites d'authentification et de gestion des utilisateurs. Il identifie les acteurs principaux : le Visiteur (non authentifie), l'Utilisateur authentifie, l'Administrateur avec privileges etendus, et le Systeme Email pour les notifications.

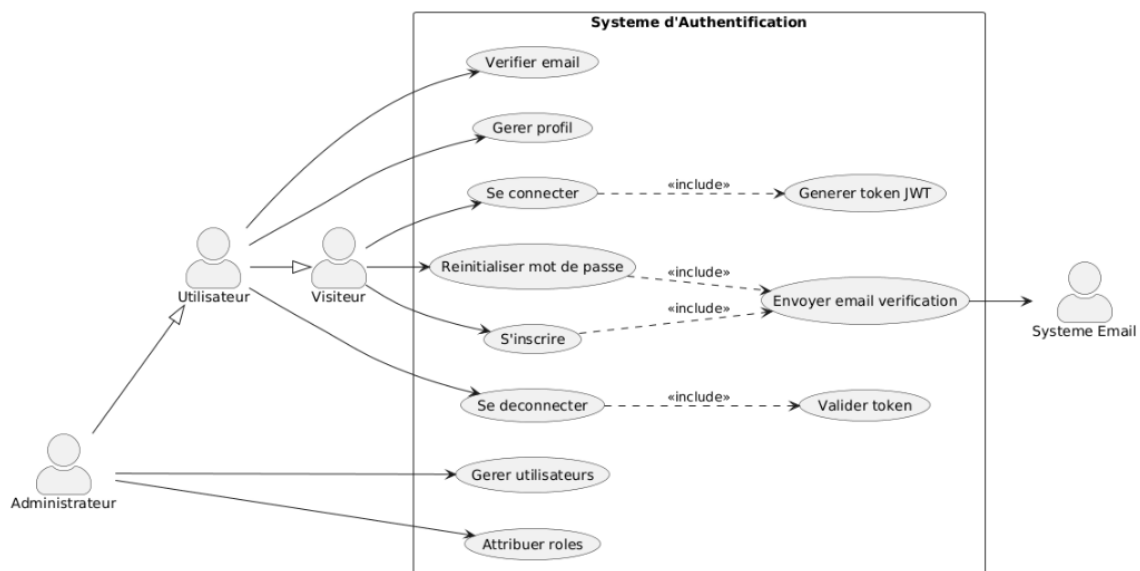


FIGURE 4.1 – Diagramme de cas d'utilisation - Module Authentification

#### Acteurs identifiés :

- **Visiteur** : Utilisateur non authentifié pouvant s'inscrire ou se connecter
- **Utilisateur** : Utilisateur authentifié avec accès aux fonctionnalités de base

- 
- **Administrateur** : Utilisateur avec privileges de gestion complete
  - **Systeme** : Responsable de l'envoi d'emails et de la validation des tokens

### 4.3.2 Diagramme de Classes

Le diagramme de classes ci-dessous presente l'architecture des entites liees a la gestion des utilisateurs et a l'authentification.

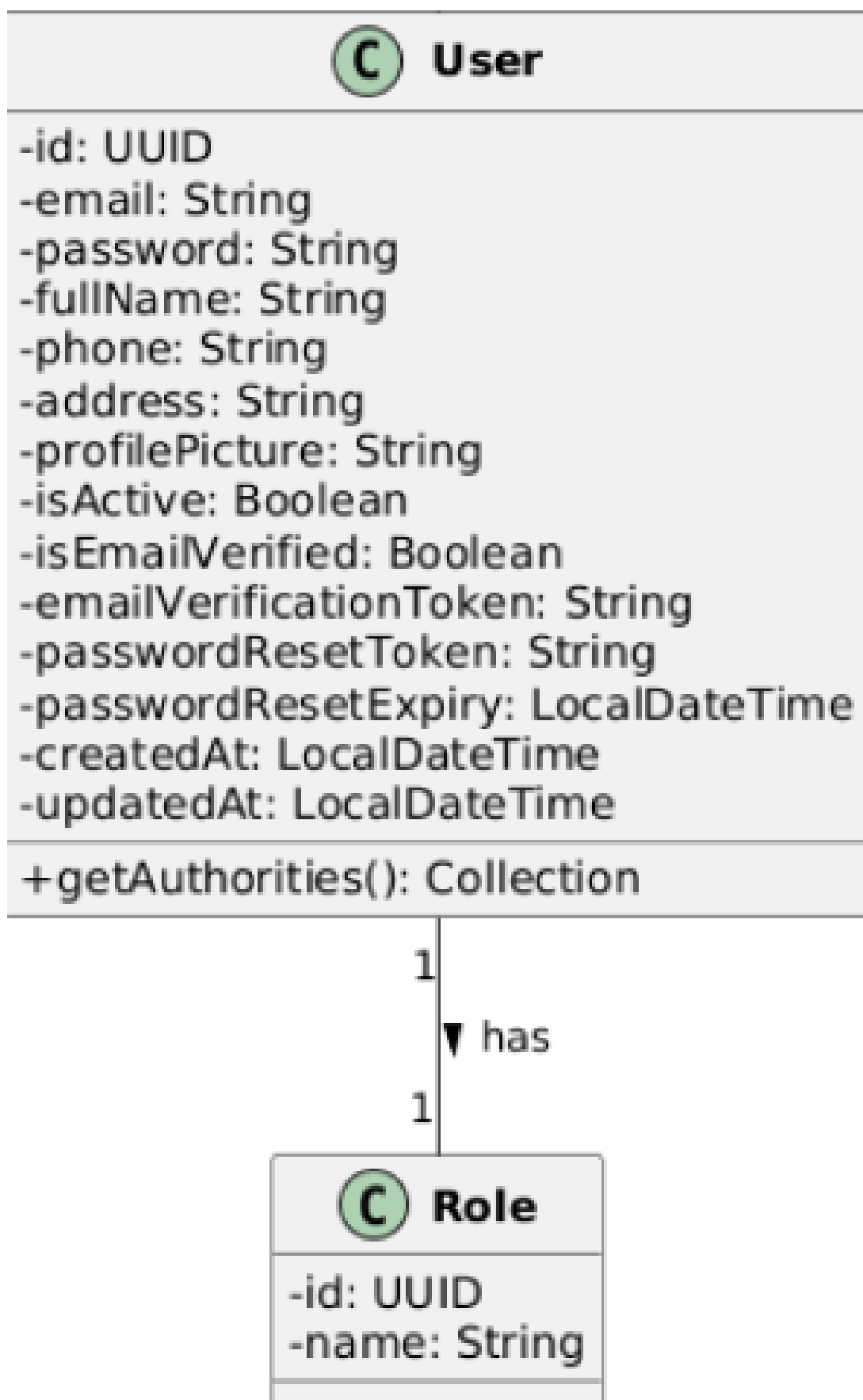


FIGURE 4.2 – Diagramme de classes - Entites User et Role

### Description des entites :

- **User** : Entite principale representant un utilisateur du systeme. Elle contient les attributs : id (UUID), email, password (hash BCrypt), fullName, phone, address, profilePicture, isActive, isEmailVerified, emailVerificationToken, passwordResetToken, et les timestamps de creation/modification.
- **Role** : Entite definissant les differents roles possibles dans le systeme. Les roles disponibles sont : CLIENT (client standard), ADMIN (administrateur), DRIVER (livreur), SUPPORT (agent de support), et SUPER\_ADMIN (super administrateur avec tous les privileges). Relation ManyToOne avec User.

### 4.3.3 Diagramme de Sequence

Le diagramme de sequence suivant illustre le processus complet de connexion avec generation et validation du token JWT.

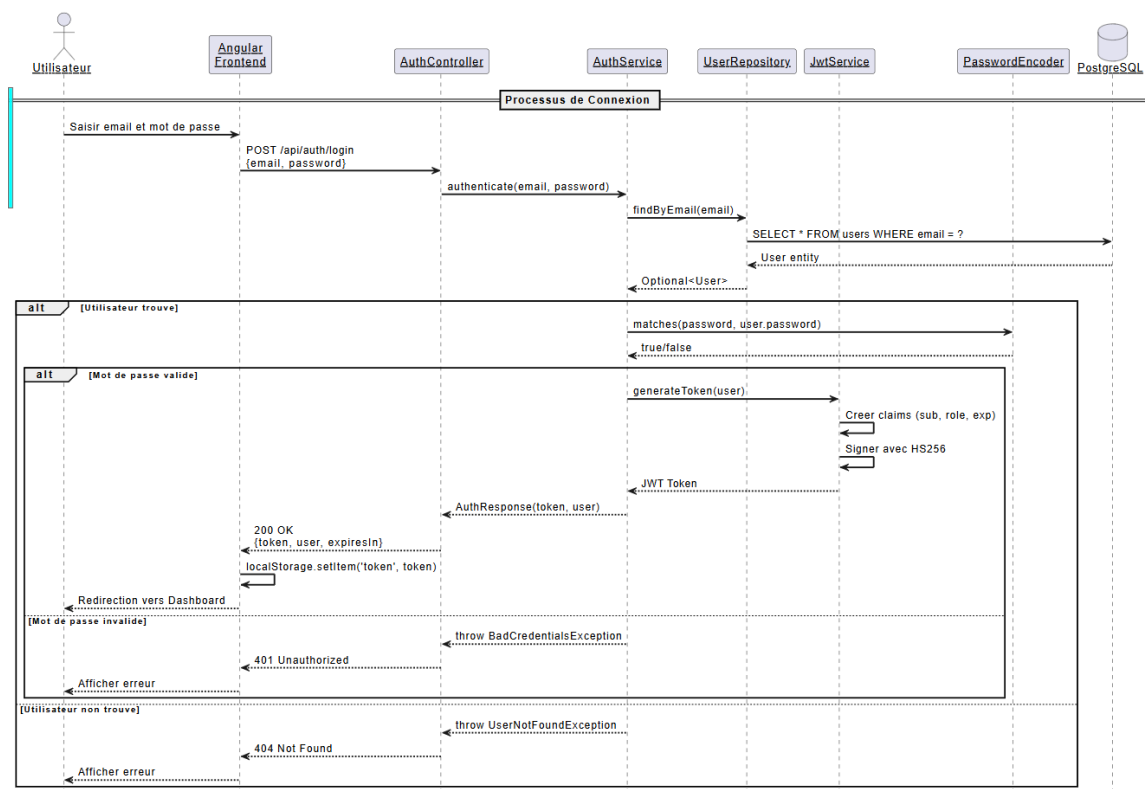


FIGURE 4.3 – Diagramme de sequence - Processus de connexion JWT

### Description du flux de connexion :

1. L'utilisateur saisit ses identifiants (email, mot de passe) dans le formulaire
2. Le frontend Angular envoie une requete POST vers /api/auth/login
3. Le backend verifie les identifiants avec BCrypt
4. Si valides, un token JWT est genere avec les claims (sub, role, exp)

5. Le token est stocke dans localStorage cote client
6. Les requetes suivantes incluent le token dans l'en-tete Authorization

## 4.4 Realisation

### 4.4.1 Architecture d'Authentification

L'architecture d'authentification mise en place repose sur plusieurs composants :

- **SecurityConfig** : Configuration Spring Security avec routes publiques et protegees
- **JwtAuthenticationFilter** : Filtre interceptant chaque requete pour valider le JWT
- **AuthService** : Logique metier d'inscription, connexion et reinitialisation

### 4.4.2 Gestion des Roles

TABLE 4.3 – Roles et permissions

Role	Permissions
CLIENT	Acces au catalogue, panier, commandes, profil
ADMIN	Gestion produits, commandes, utilisateurs, analytics
DRIVER	Acces aux livraisons assignees, mise a jour des statuts
SUPPORT	Acces au chat support, gestion des reclamations
SUPER_ADMIN	Tous les privileges, gestion des admins, configuration systeme

## 4.5 Bilan du Sprint

TABLE 4.4 – Bilan Sprint 1

Metrique	Valeur
User Stories planifiees	6
User Stories terminees	6
Points planifies	24
Points realises	24
Velocite	100%

Le Sprint 1 a ete complete avec succes, etablissant une base solide pour les sprints suivants.

# Chapitre 5

## Sprint 2 : Gestion des Produits et Catalogue

### 5.1 Presentation du Sprint

#### 5.1.1 Objectifs

Le deuxième sprint se concentre sur le développement du catalogue de produits, incluant les fonctionnalités de recherche, de filtrage et l'interface d'administration pour la gestion des produits.

TABLE 5.1 – Fiche du Sprint 2

Attribut	Valeur
Duree	2 semaines
Theme	Gestion des Produits
Priorite	Haute

### 5.2 Sprint Backlog

### 5.3 Analyse et Conception

#### 5.3.1 Diagramme de Cas d'Utilisation

Le diagramme suivant présente les fonctionnalités de gestion du catalogue différenciées par acteur.



TABLE 5.2 – Sprint Backlog - Sprint 2

ID	User Story	Priorite	Points
US-2.1	En tant qu'utilisateur, je veux parcourir les produits par categorie	Haute	5
US-2.2	En tant qu'utilisateur, je veux rechercher des produits par mot-cle	Haute	5
US-2.3	En tant qu'utilisateur, je veux filtrer les produits par prix et marque	Haute	5
US-2.4	En tant qu'admin, je veux ajouter de nouveaux produits	Haute	8
US-2.5	En tant qu'admin, je veux modifier et supprimer des produits	Haute	5
US-2.6	En tant qu'admin, je veux gerer les categories et marques	Moyenne	5
US-2.7	En tant qu'utilisateur, je veux voir les details d'un produit	Haute	3

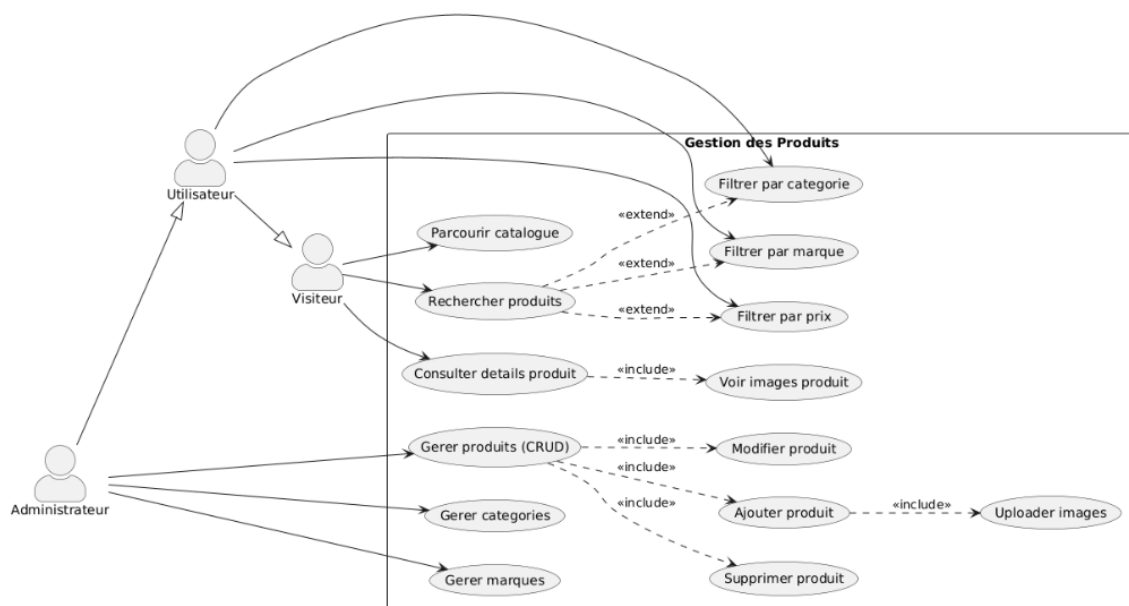


FIGURE 5.1 – Diagramme de cas d'utilisation - Gestion des Produits

### Cas d'utilisation principaux :

- **Parcourir le catalogue** : Navigation dans les produits avec pagination
- **Rechercher des produits** : Recherche textuelle par nom ou description
- **Filtrer par criteres** : Application de filtres (categorie, marque, prix)
- **Gerer les produits** : Operations CRUD sur les produits (Admin)

### 5.3.2 Diagramme de Classes

Ce diagramme presente la structure des entites produit avec leurs relations.

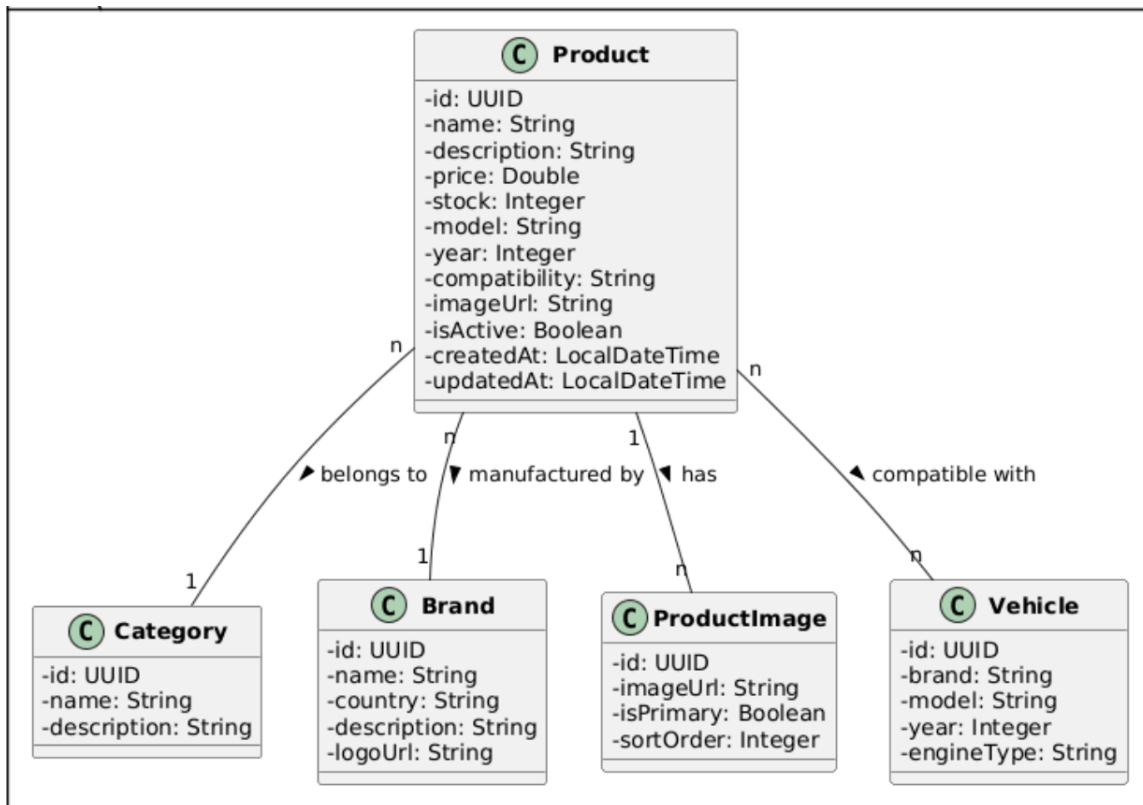


FIGURE 5.2 – Diagramme de classes - Entites Product, Category, Brand et Vehicle

#### Description des entites :

- **Product** : Entite centrale representant une piece automobile. Attributs : id (UUID), name, description, price, stock, model, year, compatibility, imageUrl, isActive, timestamps. Relations ManyToOne avec Category et Brand, et ManyToMany avec Vehicle.
- **Category** : Classification hierarchique des produits (Freinage, Moteur, Electricite, Suspension). Attributs : id, name, description.
- **Brand** : Marques des pieces automobiles (Bosch, Valeo, Brembo). Attributs : id, name, country, description, logoUrl.
- **Vehicle** : Vehicule compatible avec les pieces. Attributs : id, brand, model, year, engineType. Un produit peut etre compatible avec plusieurs vehicules.
- **ProductImage** : Images additionnelles d'un produit. Attributs : id, imageUrl, isPrimary, sortOrder. Relation ManyToOne vers Product.

### 5.3.3 Diagramme de Sequence

Le diagramme illustre le flux de recherche avec filtres multiples.

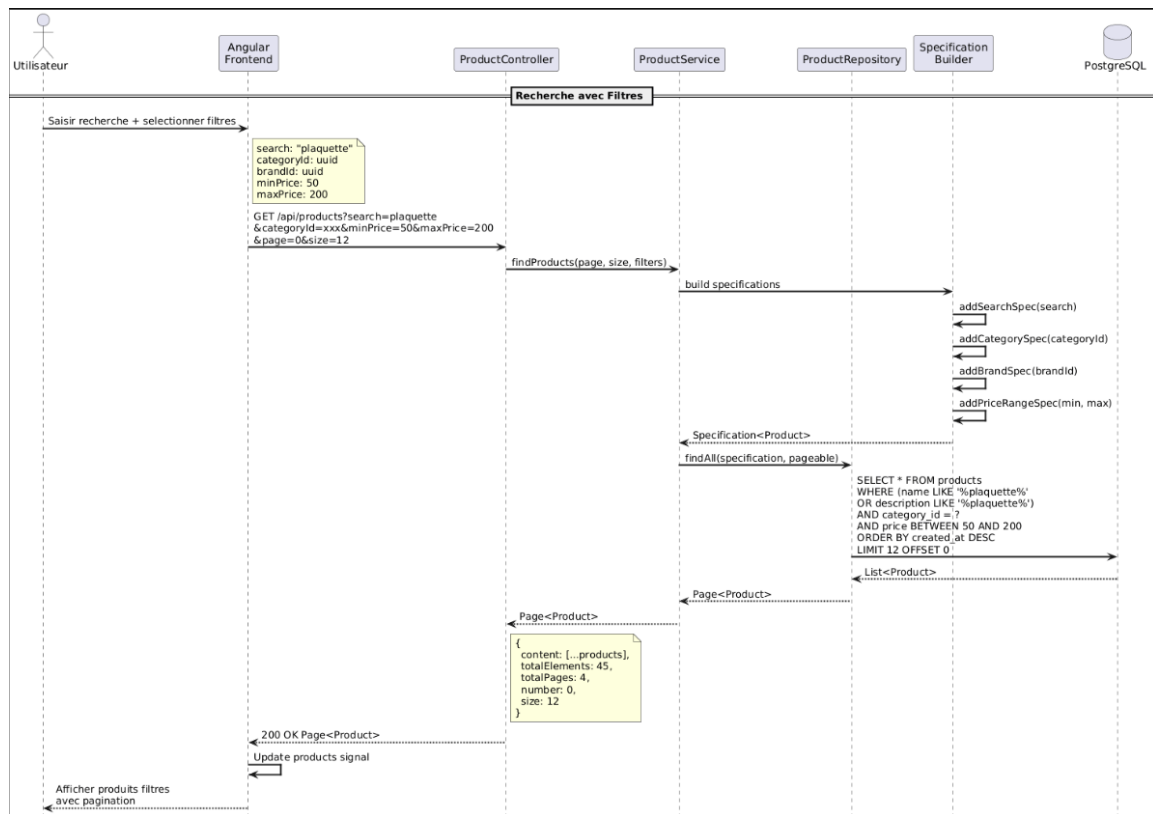


FIGURE 5.3 – Diagramme de sequence - Recherche et filtrage de produits

## 5.4 Realisation

### 5.4.1 Architecture du Catalogue

- **ProductController** : Endpoints REST pour consultation et administration
- **ProductService** : Logique metier incluant recherche avec Specifications JPA
- **ProductRepository** : Interface Spring Data JPA avec methodes personnalisees

## 5.5 Bilan du Sprint

TABLE 5.3 – Bilan Sprint 2

Metrique	Valeur
User Stories terminees	7/7
Points realises	36
Velocite	100%

# Chapitre 6

## Sprint 3 : Panier et Gestion des Commandes

### 6.1 Présentation du Sprint

#### 6.1.1 Objectifs

Le troisieme sprint se concentre sur le developpement des fonctionnalites de panier d'achat et de gestion des commandes.

TABLE 6.1 – Fiche du Sprint 3

Attribut	Valeur
Duree	2 semaines
Theme	Panier et Commandes
Priorite	Haute

### 6.2 Sprint Backlog

### 6.3 Analyse et Conception

#### 6.3.1 Diagramme de Cas d'Utilisation

Le diagramme suivant presente l'ensemble des fonctionnalites liees au panier et aux commandes.

TABLE 6.2 – Sprint Backlog - Sprint 3

ID	User Story	Priorite	Points
US-3.1	En tant qu'utilisateur, je veux ajouter des produits a mon panier	Haute	5
US-3.2	En tant qu'utilisateur, je veux modifier les quantites dans mon panier	Haute	3
US-3.3	En tant qu'utilisateur, je veux supprimer des articles de mon panier	Haute	2
US-3.4	En tant qu'utilisateur, je veux passer une commande	Haute	8
US-3.5	En tant qu'utilisateur, je veux voir l'historique de mes commandes	Haute	5
US-3.6	En tant qu'admin, je veux gerer les statuts des commandes	Haute	5
US-3.7	En tant qu'utilisateur, je veux recevoir une confirmation de commande	Moyenne	3

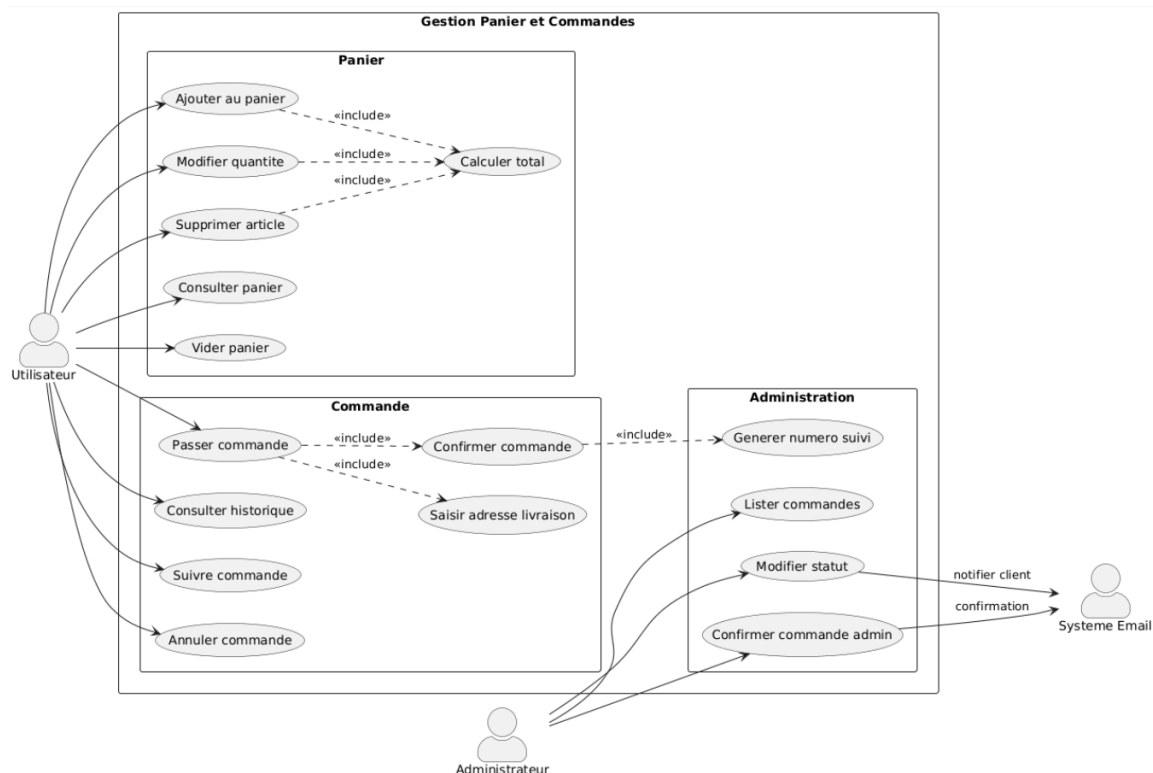


FIGURE 6.1 – Diagramme de cas d'utilisation - Panier et Commandes

### Cas d'utilisation - Panier :

- **Ajouter au panier :** Ajout d'un produit avec quantite specifiee
- **Modifier quantite :** Augmentation ou diminution de la quantite
- **Supprimer du panier :** Retrait d'un article du panier
- **Consulter panier :** Visualisation du contenu et du total

- **Vider panier** : Suppression de tous les articles

**Cas d'utilisation - Commande :**

- **Passer commande** : Finalisation de l'achat avec adresse de livraison
- **Consulter historique** : Visualisation de toutes les commandes passees
- **Suivre commande** : Consultation du statut actuel d'une commande
- **Annuler commande** : Annulation d'une commande en attente

**Cas d'utilisation - Administration :**

- **Lister commandes** : Vue d'ensemble de toutes les commandes
- **Modifier statut** : Changement du statut d'une commande
- **Confirmer commande** : Validation d'une commande en attente

### 6.3.2 Diagramme de Classes

Le diagramme suivant illustre les relations entre les entites du module panier et commandes.

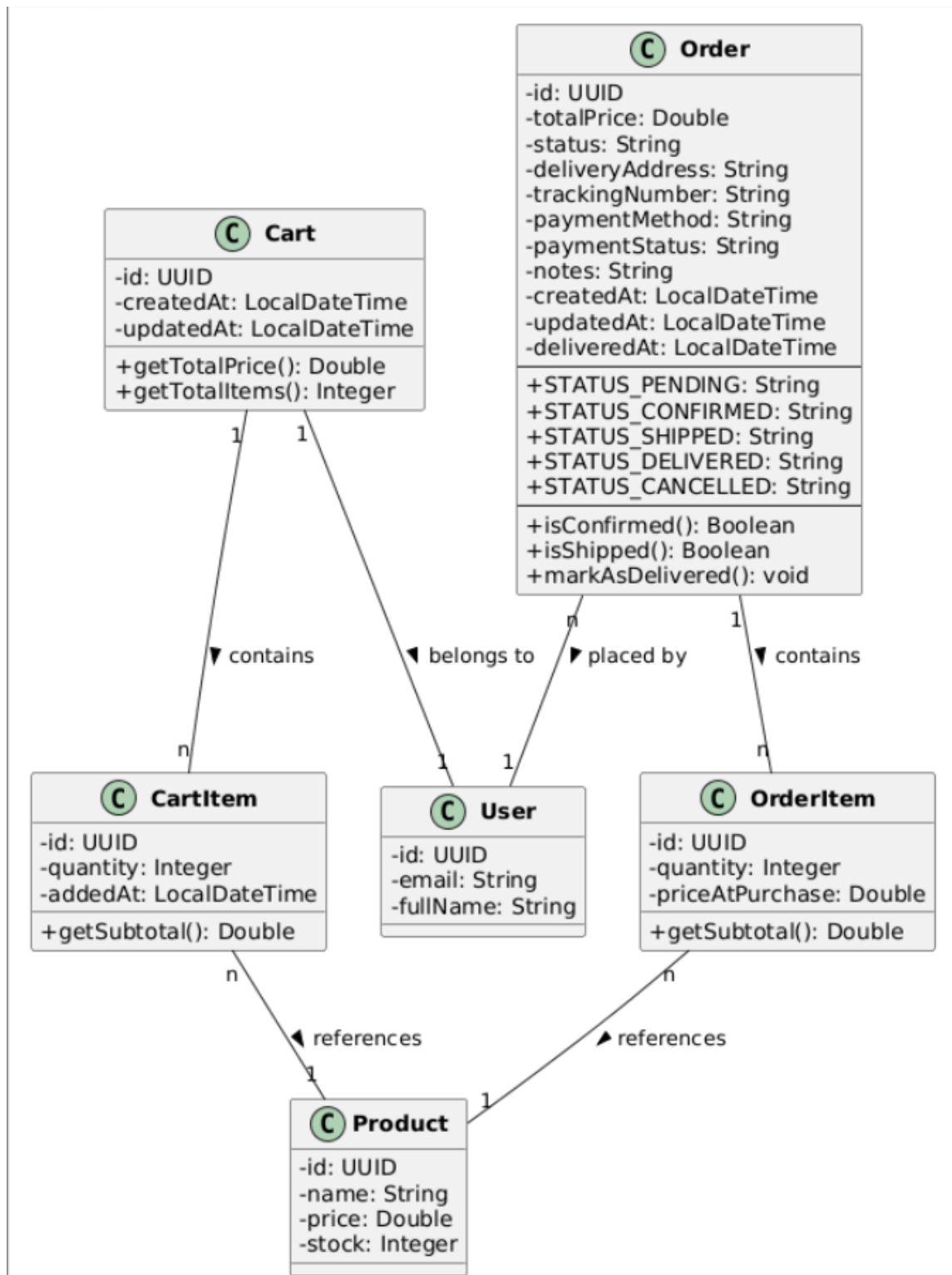


FIGURE 6.2 – Diagramme de classes - Entites Cart, Order, OrderItem et Product

**Description des entites :**

- **Cart** : Panier d'achat d'un utilisateur. Attributs : id, createdAt, updatedAt. Relation OneToOne avec User, OneToMany avec CartItem. Methodes : getTotalPrice(), getTotalItems().

- **CartItem** : Article dans le panier. Attributs : id, quantity, addedAt. Relations ManyToOne avec Cart et Product. Methode : getSubtotal().
- **Order** : Commande validee. Attributs : id, totalPrice, status, deliveryAddress, trackingNumber, paymentMethod, paymentStatus, notes, timestamps. Relation ManyToOne avec User, OneToMany avec OrderItem.
- **OrderItem** : Article d'une commande avec prix fige. Attributs : id, quantity, priceAtPurchase. Relations ManyToOne avec Order et Product.
- **Product** : Produit reference par CartItem et OrderItem. Fournit les informations de prix et stock pour le calcul du panier.

### 6.3.3 Diagramme de Sequence

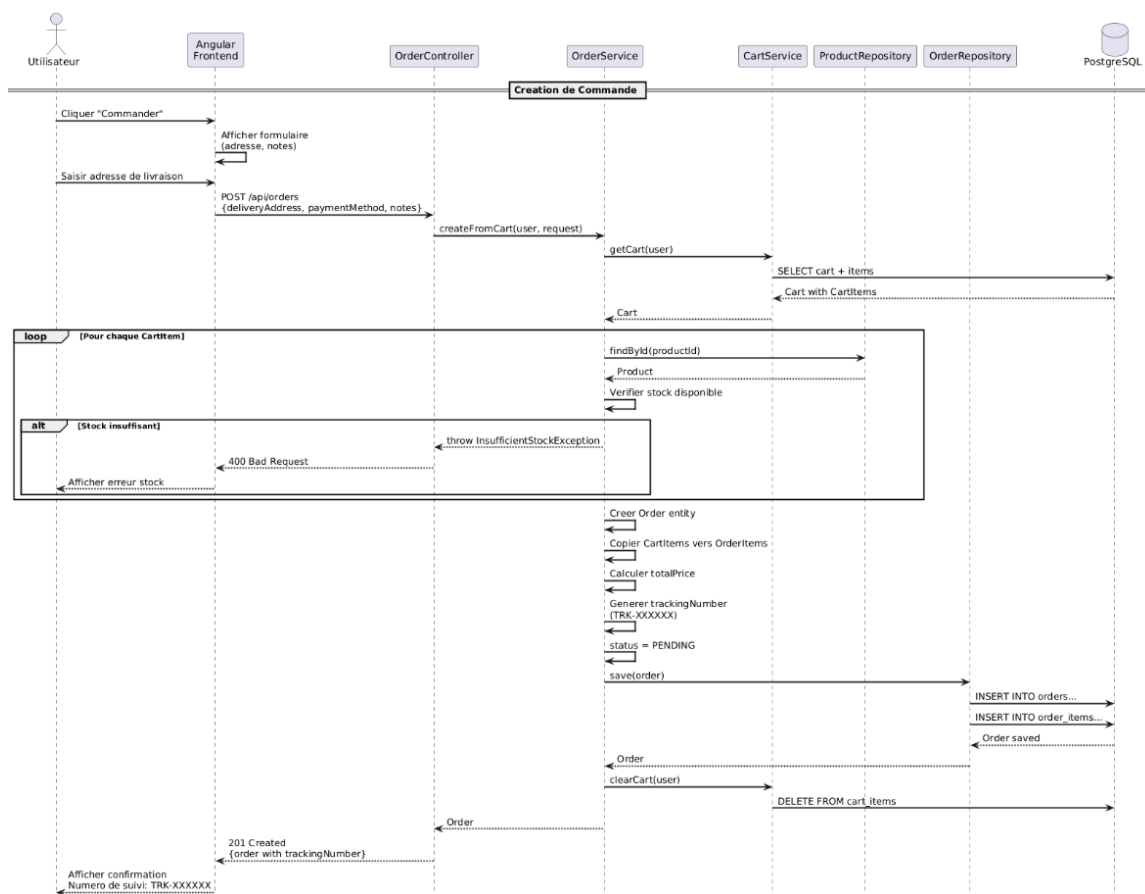


FIGURE 6.3 – Diagramme de sequence - Creation d'une commande

## 6.4 Realisation

### 6.4.1 Cycle de Vie des Commandes



TABLE 6.3 – Statuts des commandes

Statut	Description
PENDING	Commande creee, en attente de confirmation
CONFIRMED	Commande validee par l'administrateur
SHIPPED	Commande expediee, livraison en cours
DELIVERED	Commande livree au client
CANCELLED	Commande annulee

## 6.5 Bilan du Sprint

TABLE 6.4 – Bilan Sprint 3

Metrique	Valeur
User Stories terminees	7/7
Points realises	31
Velocite	100%

# Chapitre 7

## Sprint 4 : Paiement Stripe et Gestion d'Inventaire

### 7.1 Presentation du Sprint

#### 7.1.1 Objectifs

Le quatrieme sprint vise a integrer le systeme de paiement securise Stripe et a developper un module de gestion d'inventaire.

TABLE 7.1 – Fiche du Sprint 4

Attribut	Valeur
Duree	2 semaines
Theme	Paiement et Inventaire
Priorite	Haute

### 7.2 Sprint Backlog

TABLE 7.2 – Sprint Backlog - Sprint 4

ID	User Story	Priorite	Points
US-4.1	En tant qu'utilisateur, je veux payer par carte bancaire	Haute	8
US-4.2	En tant qu'utilisateur, je veux un paiement securise	Haute	5
US-4.3	En tant qu'admin, je veux suivre les niveaux de stock	Haute	5
US-4.4	En tant qu'admin, je veux des alertes de stock bas	Moyenne	5
US-4.5	En tant qu'admin, je veux gerer les fournisseurs	Moyenne	5
US-4.6	En tant qu'admin, je veux creer des commandes de reapprovisionnement	Moyenne	5

## 7.3 Analyse et Conception

### 7.3.1 Diagramme de Cas d'Utilisation

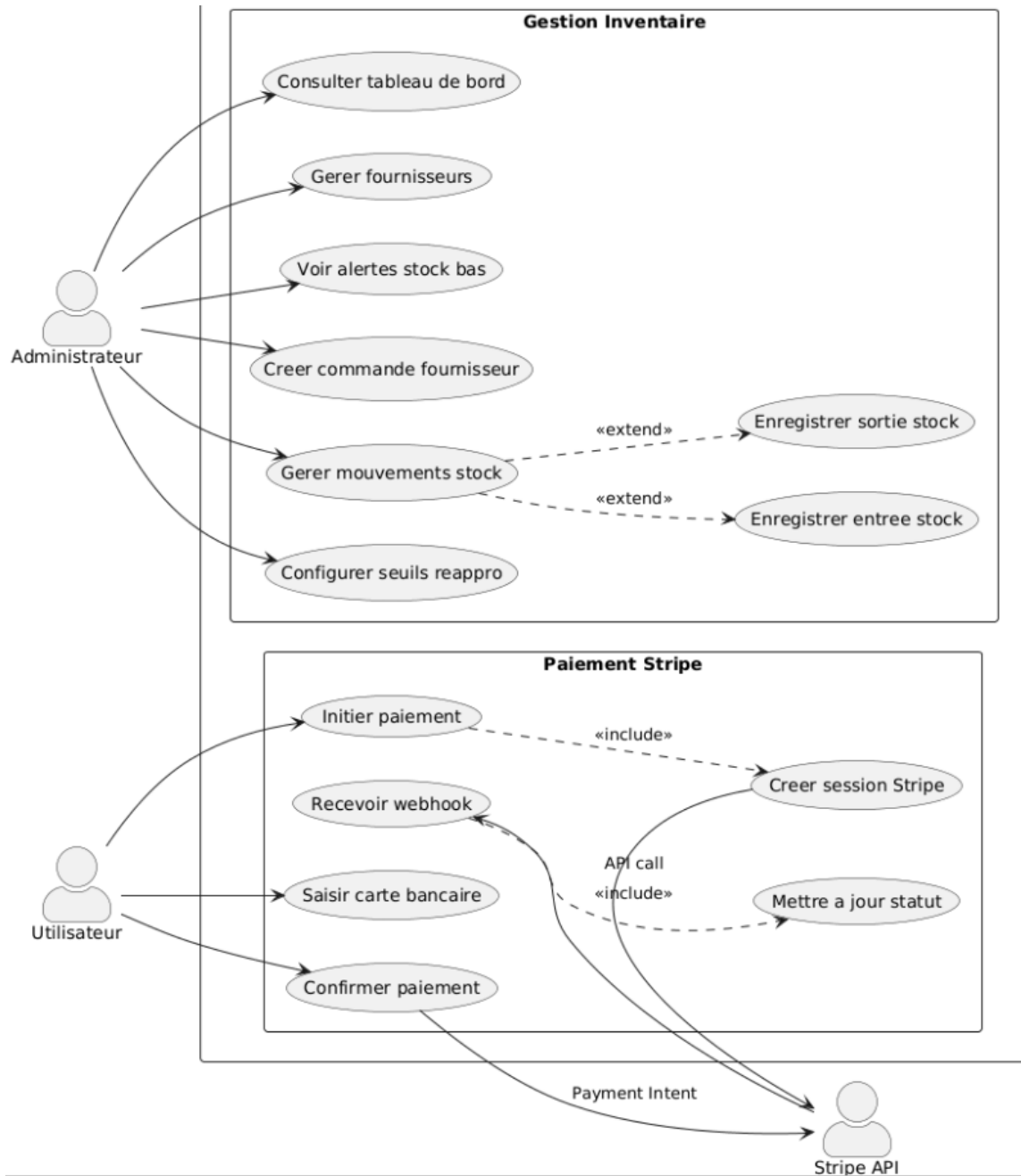


FIGURE 7.1 – Diagramme de cas d'utilisation - Paiement et Inventaire

#### Cas d'utilisation - Paiement :

- **Initier paiement** : Demarrage du processus de paiement
- **Saisir carte** : Entree des informations bancaires via Stripe
- **Confirmer paiement** : Validation de la transaction

**Cas d'utilisation - Inventaire :**

- **Consulter tableau de bord** : Vue d'ensemble des stocks
- **Gerer mouvements** : Enregistrement des entrees/sorties de stock
- **Gerer fournisseurs** : CRUD sur les fournisseurs
- **Commander reapprovisionnement** : Creation de commandes fournisseurs

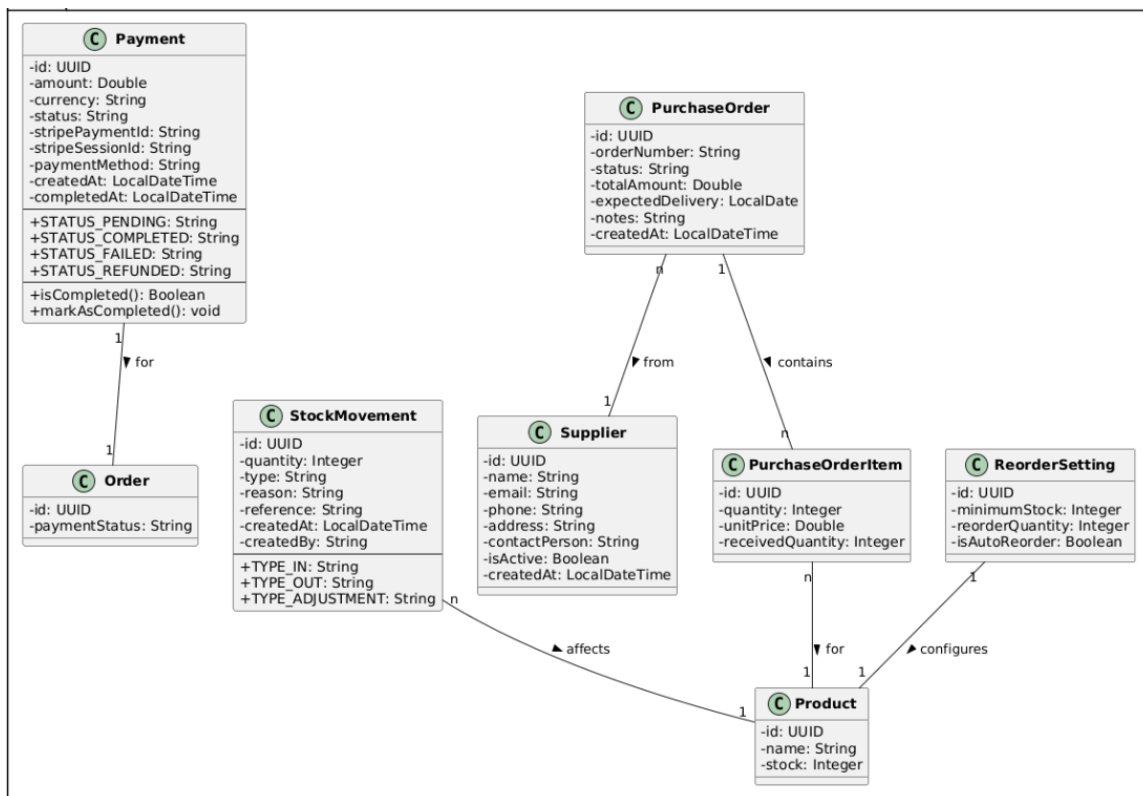
**7.3.2 Diagramme de Classes**

FIGURE 7.2 – Diagramme de classes - Payment, StockMovement, Supplier, PurchaseOrder et Product

**Description des entites :**

- **Payment** : Enregistrement d'un paiement. Attributs : id, amount, currency (TND), status (PENDING, COMPLETED, FAILED, REFUNDED), stripePaymentId, stripeSessionId, paymentMethod, timestamps. Relation OneToOne avec Order.
- **Order** : Commande associee au paiement. Le paiement reference la commande pour laquelle il a ete effectue.
- **StockMovement** : Historique des mouvements de stock. Attributs : id, quantity, type (IN, OUT, ADJUSTMENT), reason, reference, createdBy, createdAt. Relation ManyToOne avec Product.

- **Product** : Produit dont le stock est affecté par les mouvements. Le stock est mis à jour automatiquement.
- **Supplier** : Fournisseur de pièces. Attributs : id, name, email, phone, address, contactPerson, isActive.
- **PurchaseOrder** : Commande de reapprovisionnement. Attributs : id, orderNumber, status, totalAmount, expectedDelivery, notes. Relation ManyToOne avec Supplier, OneToMany avec PurchaseOrderItem.
- **PurchaseOrderItem** : Article d'une commande fournisseur. Attributs : id, quantity, unitPrice, receivedQuantity. Relation ManyToOne avec Product.

### 7.3.3 Diagramme de Sequence

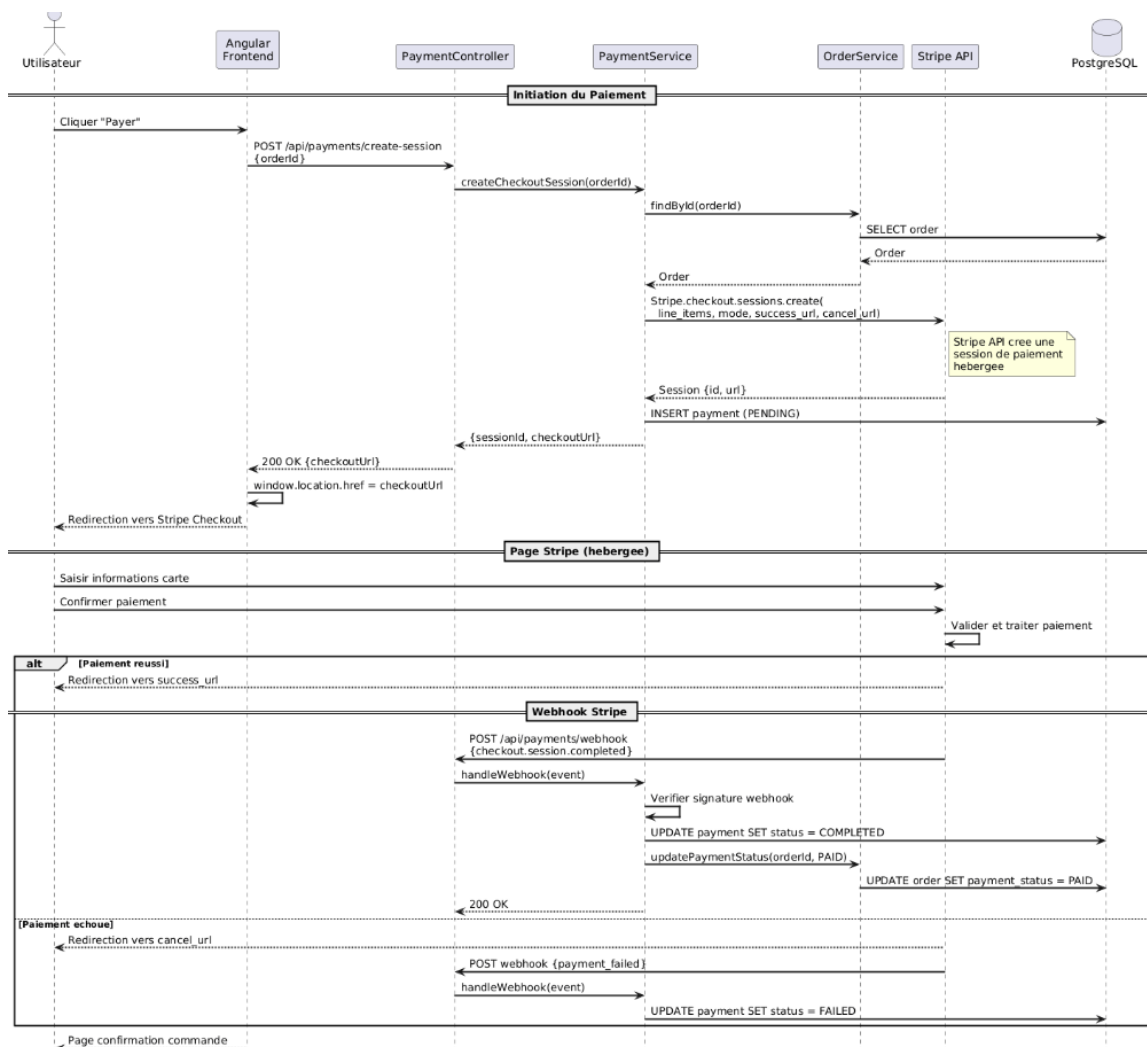


FIGURE 7.3 – Diagramme de sequence - Paiement avec Stripe Checkout

## 7.4 Realisation

### 7.4.1 Integration Stripe

- **Stripe Checkout** : Interface hebergee par Stripe, conforme PCI-DSS
- **Webhooks** : Endpoint securise pour confirmation asynchrone
- **Mode Test** : Cles API de test pour simulation

## 7.5 Bilan du Sprint

TABLE 7.3 – Bilan Sprint 4

Metrique	Valeur
User Stories terminees	6/6
Points realises	33
Velocite	100%

# Chapitre 8

## Sprint 5 : Module d'Intelligence Artificielle

### 8.1 Présentation du Sprint

#### 8.1.1 Objectifs

Le cinquieme sprint est consacre au developpement du module d'Intelligence Artificielle, permettant la reconnaissance d'images et les recommandations personnalisées.

TABLE 8.1 – Fiche du Sprint 5

Attribut	Valeur
Duree	2 semaines
Theme	Module IA
Priorite	Haute

### 8.2 Sprint Backlog

TABLE 8.2 – Sprint Backlog - Sprint 5

ID	User Story	Priorite	Points
US-5.1	En tant qu'utilisateur, je veux identifier une piece a partir d'une photo	Haute	13
US-5.2	En tant qu'utilisateur, je veux des recommandations basees sur mon historique	Haute	8
US-5.3	En tant qu'utilisateur, je veux decrire des symptomes et obtenir des suggestions	Moyenne	8
US-5.4	En tant qu'admin, je veux voir les statistiques d'utilisation de l'IA	Moyenne	5

## 8.3 Analyse et Conception

### 8.3.1 Diagramme de Cas d'Utilisation

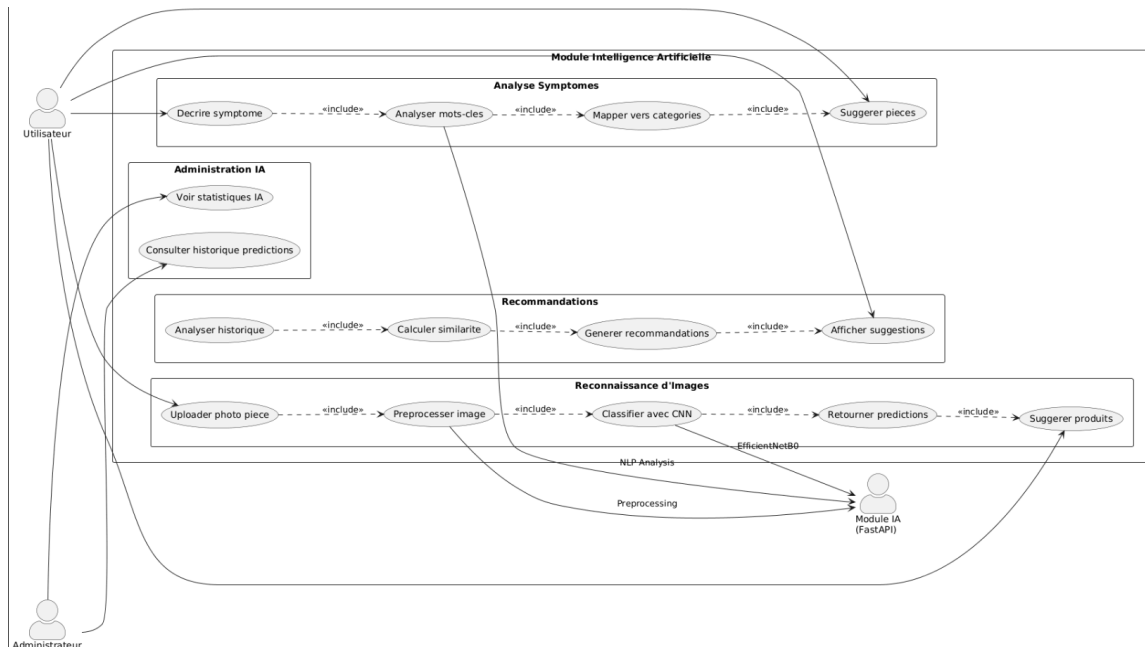


FIGURE 8.1 – Diagramme de cas d'utilisation - Module IA

#### Cas d'utilisation principaux :

- **Identifier piece** : L'utilisateur uploade une photo pour identification
- **Obtenir recommandations** : Suggestions basees sur l'historique d'achat
- **Analyser symptomes** : Description textuelle pour suggestion de pieces



### 8.3.2 Diagramme de Classes - Backend

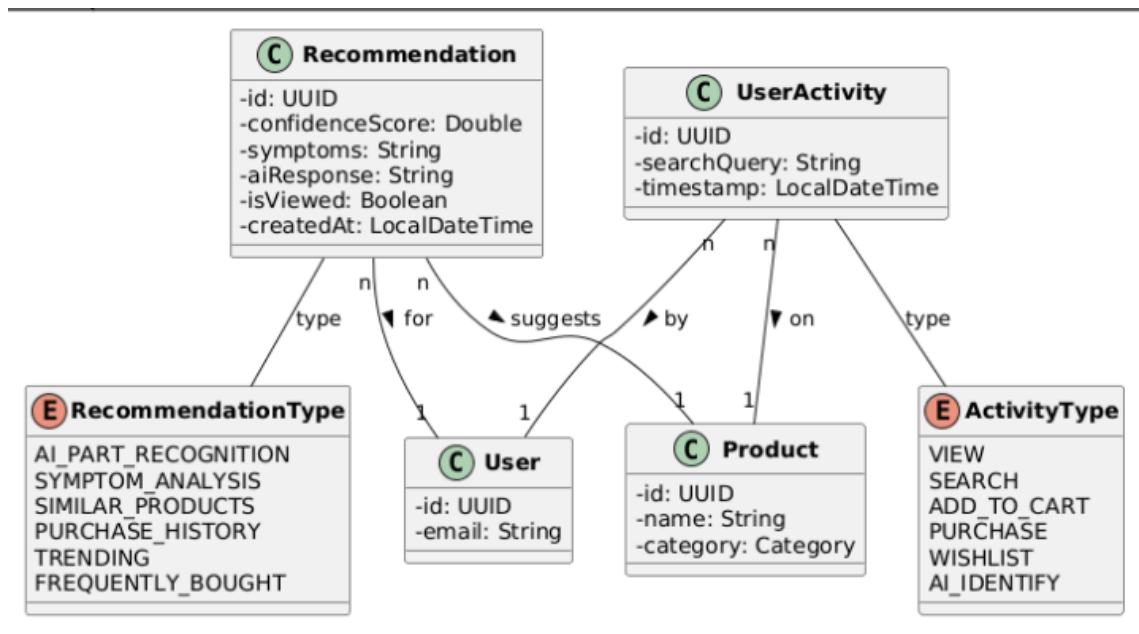


FIGURE 8.2 – Diagramme de classes - Recommendation, UserActivity et entites associees

#### Description des entites :

- **Recommendation** : Enregistrement d'une recommandation IA. Attributs : id, confidenceScore, symptoms, aiResponse, isViewed, createdAt. Relations ManyToOne avec User et Product.
- **RecommendationType** : Enumeration definissant les types de recommandations : AI\_PART\_RECOGNITION, SYMPTOM\_ANALYSIS, SIMILAR\_PRODUCTS, PURCHASE\_HISTORY, TRENDING, FREQUENTLY\_BOUGHT.
- **UserActivity** : Historique des activites utilisateur. Attributs : id, searchQuery, timestamp. Relations ManyToOne avec User et Product.
- **ActivityType** : Enumeration des types d'activites : VIEW, SEARCH, ADD\_TO\_CART, PURCHASE, WISHLIST, AI\_IDENTIFY.
- **User** : Utilisateur pour lequel les recommandations sont generees et les activites enregistrees.
- **Product** : Produit recommande ou concerne par l'activite utilisateur.

### 8.3.3 Diagramme de Classes - Module Python

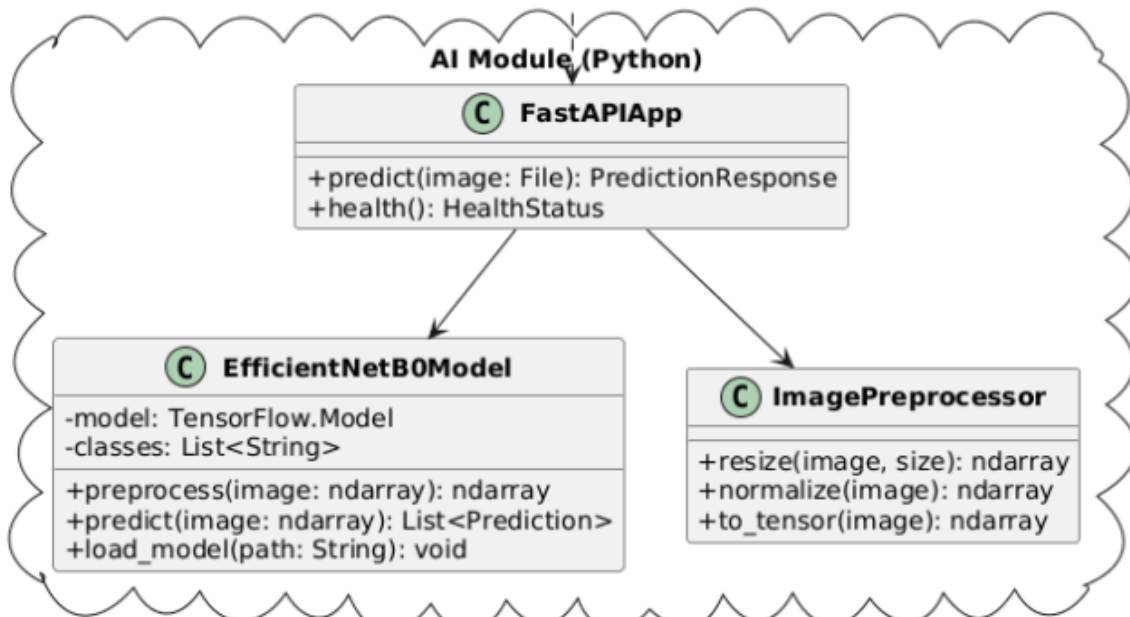


FIGURE 8.3 – Diagramme de classes - Module IA Python (FastAPI, EfficientNetB0)

#### Composants IA :

- **FastAPIApp** : Application Python exposant les endpoints `/predict` et `/health`
- **EfficientNetB0Model** : Modèle CNN pré-entraîné avec 50 classes de pièces
- **ImagePreprocessor** : Traitement des images (resize 224x224, normalisation)

### 8.3.4 Diagramme de Sequence

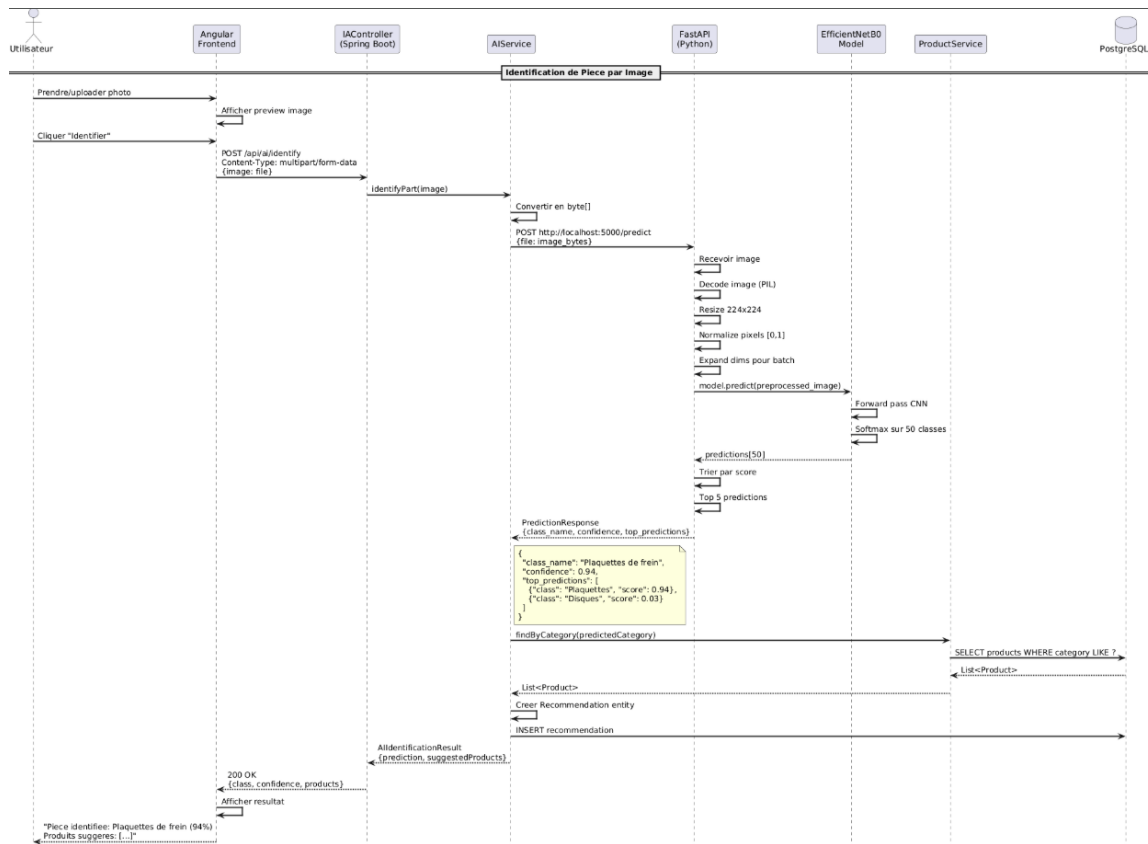


FIGURE 8.4 – Diagramme de sequence - Reconnaissance d'image IA

## 8.4 Realisation

### 8.4.1 Modele de Classification

TABLE 8.3 – Specifications du modele IA

Caracteristique	Valeur
Architecture	EfficientNetB0 (pre-entraine ImageNet)
Nombre de classes	50 types de pieces
Taille d'entree	224 x 224 pixels
Accuracy (validation)	94.2%
Temps d'inference	50ms

## 8.5 Bilan du Sprint

TABLE 8.4 – Bilan Sprint 5

Metrique	Valeur
User Stories terminees	4/4
Points realises	34
Velocite	100%

# Chapitre 9

## Sprint 6 : Chat Support, Livraison et Validation Finale

### 9.1 Presentation du Sprint

#### 9.1.1 Objectifs

Le sixieme et dernier sprint couvre le developpement du systeme de chat en temps reel, du suivi de livraison avec cartographie, et la validation finale du projet.

TABLE 9.1 – Fiche du Sprint 6

Attribut	Valeur
Duree	2 semaines
Theme	Chat, Livraison, Validation
Priorite	Haute

### 9.2 Sprint Backlog

TABLE 9.2 – Sprint Backlog - Sprint 6

ID	User Story	Priorite	Points
US-6.1	En tant qu'utilisateur, je veux contacter le support par chat	Haute	8
US-6.2	En tant qu'utilisateur, je veux suivre ma livraison sur une carte	Haute	8
US-6.3	En tant qu'admin, je veux gerer les livraisons	Haute	5
US-6.4	En tant que livreur, je veux mettre a jour le statut de livraison	Moyenne	5
US-6.5	En tant qu'utilisateur, je veux soumettre une reclamation	Moyenne	5
US-6.6	Tests d'integration et validation finale	Haute	8

## 9.3 Analyse et Conception

### 9.3.1 Diagramme de Cas d'Utilisation

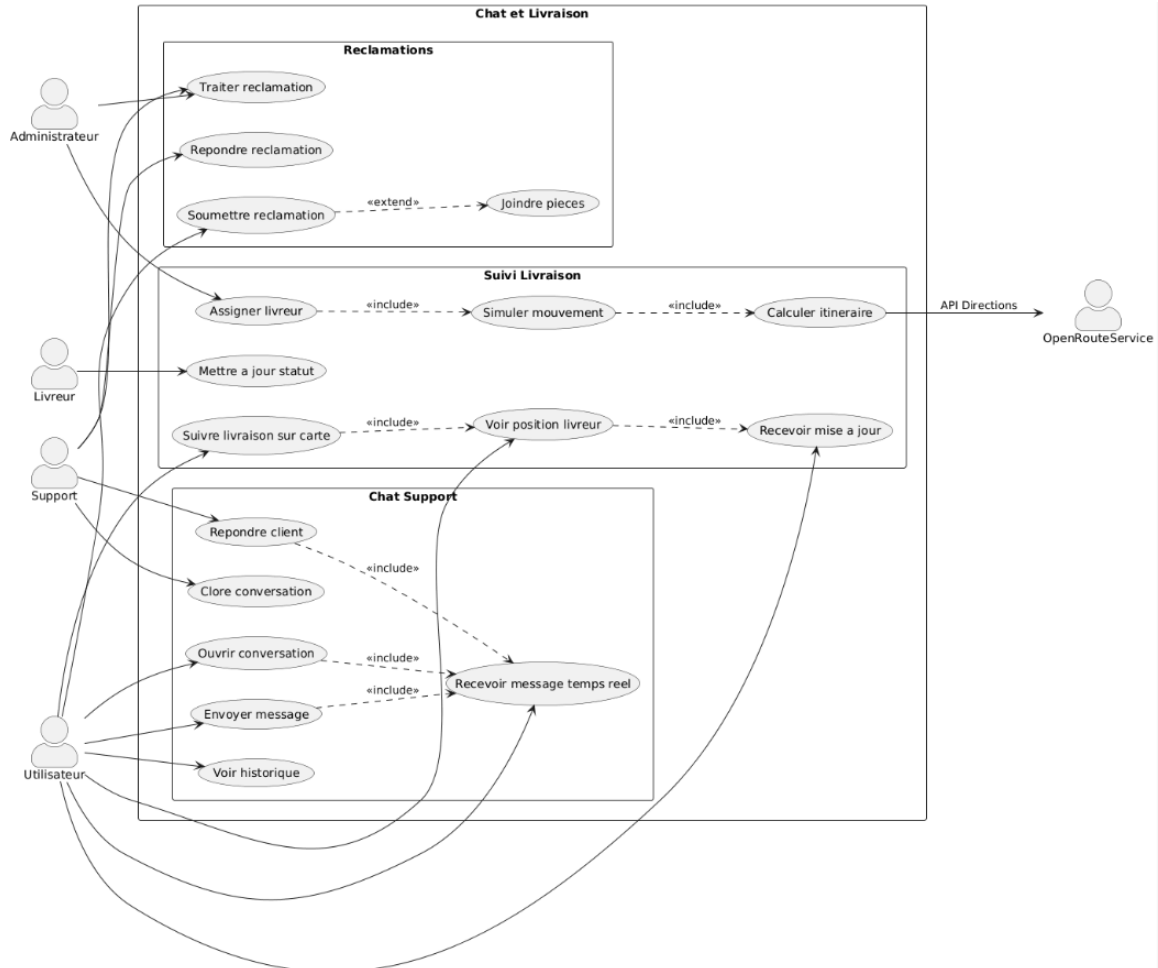


FIGURE 9.1 – Diagramme de cas d'utilisation - Chat et Livraison

#### Acteurs :

- **Utilisateur** : Chat avec support, suivi livraison, réclamations
- **Support** : Repondre aux conversations, traiter réclamations
- **Administrateur** : Assigner livreurs, gerer livraisons
- **Livreur** : Mettre a jour statut, transmettre position

### 9.3.2 Diagramme de Classes

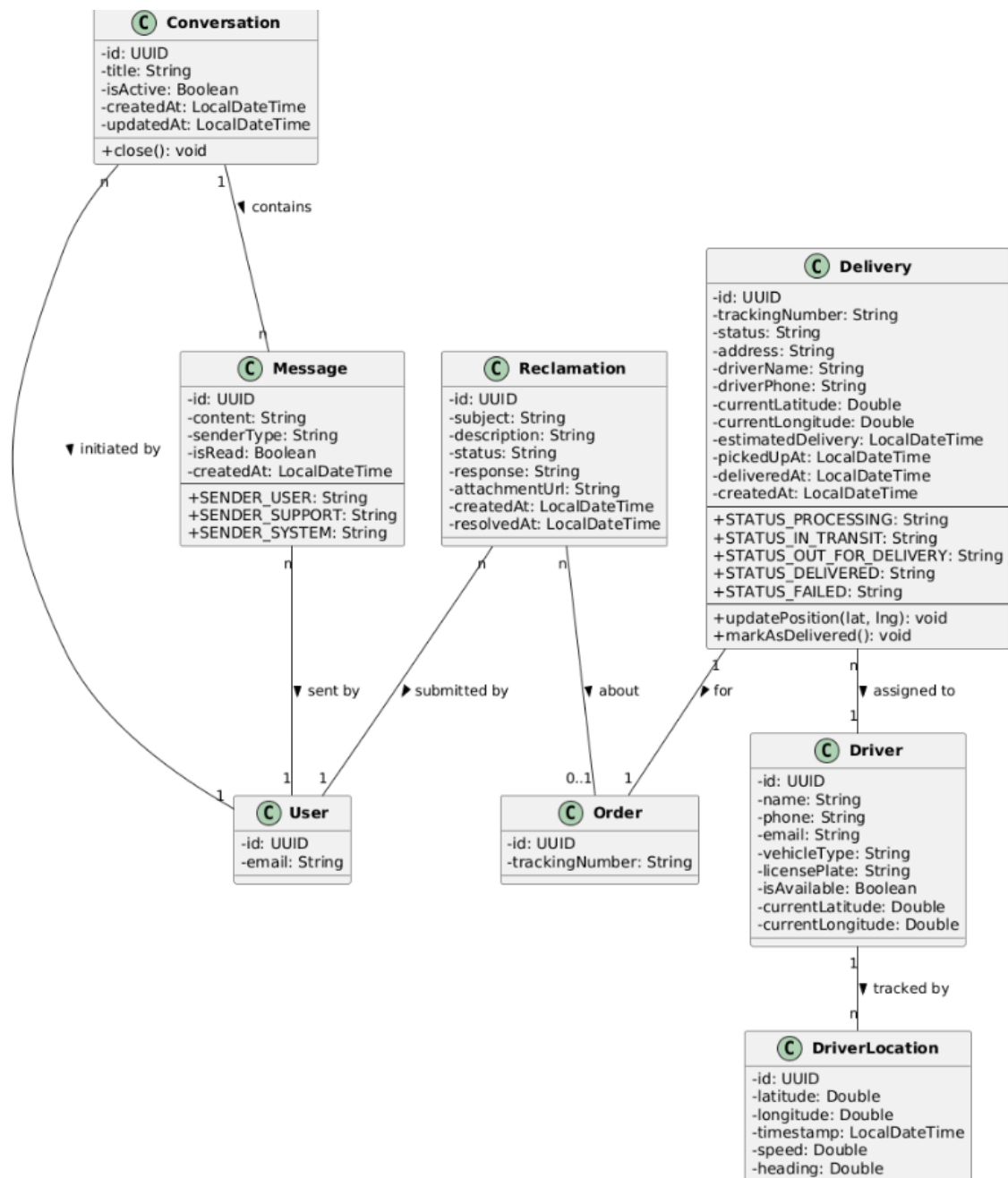


FIGURE 9.2 – Diagramme de classes - Conversation, Message, Delivery, Driver et entites associees

#### Description des entites :

- **Conversation** : Discussion entre un utilisateur et le support. Attributs : id, title, isActive, createdAt, updatedAt. Relations ManyToOne avec User, OneToMany avec Message.
- **Message** : Message dans une conversation. Attributs : id, content, senderType (USER,

---

SUPPORT, SYSTEM), isRead, createdAt. Relation ManyToOne avec Conversation et User.

- **User** : Utilisateur initiant la conversation ou envoyant des messages.
- **Delivery** : Livraison associee a une commande. Attributs : id, trackingNumber, status, address, driverName, driverPhone, currentLatitude, currentLongitude, estimatedDelivery, pickedUpAt, deliveredAt. Relation OneToOne avec Order, ManyToOne avec Driver.
- **Order** : Commande pour laquelle la livraison est creee. Chaque commande expediee genere une livraison.
- **Driver** : Livreur assigne a la livraison. Attributs : id, name, phone, email, vehicleType, licensePlate, isAvailable, currentLatitude, currentLongitude.
- **DriverLocation** : Historique des positions du livreur. Attributs : id, latitude, longitude, timestamp, speed, heading. Relation ManyToOne avec Driver.
- **Reclamation** : Plainte soumise par un utilisateur. Attributs : id, subject, description, status, response, attachmentUrl, createdAt, resolvedAt. Relations ManyToOne avec User et Order.



### 9.3.3 Diagramme de Sequence

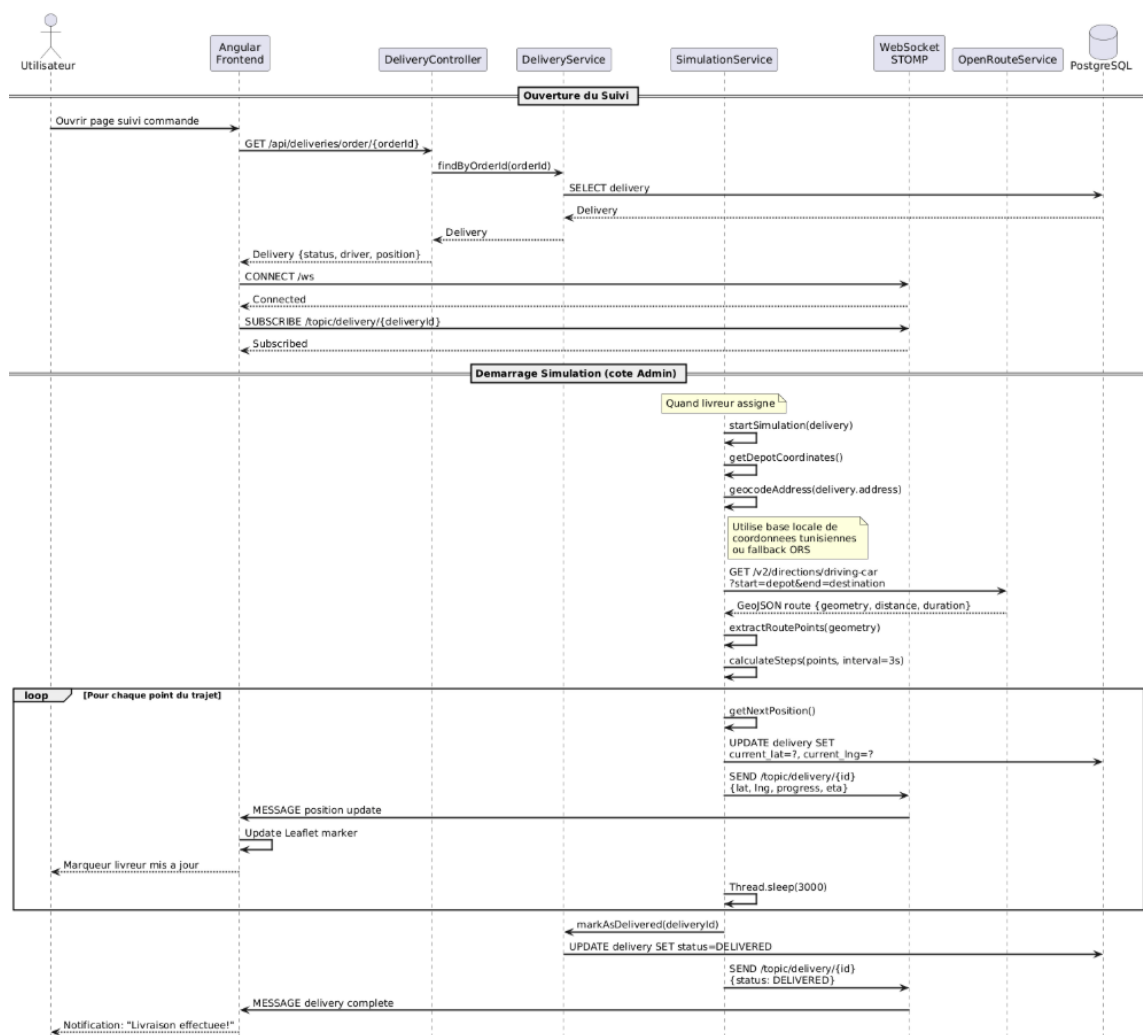


FIGURE 9.3 – Diagramme de sequence - Suivi de livraison en temps reel

## 9.4 Realisation

### 9.4.1 Chat en Temps Reel

- **WebSocket avec STOMP** : Communication bidirectionnelle
- **SockJS** : Fallback pour navigateurs incompatibles
- Messages persistants avec historique

### 9.4.2 Suivi de Livraison

## 9.5 Validation Finale

TABLE 9.3 – Statuts de livraison

Statut	Description
PROCESSING	Livraison creee, en attente d'assignation
IN_TRANSIT	Livreur en route vers la destination
OUT_FOR_DELIVERY	Livreur proche de l'adresse
DELIVERED	Colis livre avec succes
FAILED	Echec de livraison

TABLE 9.4 – Scenarios de validation

Scenario	Resultat
Parcours complet : inscription, achat, paiement, livraison	✓
Chat temps reel entre utilisateur et support	✓
Suivi de livraison avec mise a jour en temps reel	✓
Identification IA et ajout au panier	✓

## 9.6 Bilan du Sprint

TABLE 9.5 – Bilan Sprint 6

Metrique	Valeur
User Stories terminees	6/6
Points realises	39
Velocite	100%

Ce dernier sprint a permis de completer toutes les fonctionnalites et de valider l'integration de l'ensemble des modules.

# Chapitre 10

## Tests et Validation

### 10.1 Introduction

Ce chapitre presente la strategie de tests mise en place pour assurer la qualite et la fiabilite de notre plateforme. Nous avons adopte une approche multi-niveaux couvrant les tests unitaires, d'integration et de validation utilisateur.

### 10.2 Strategie de Tests

#### 10.2.1 Pyramide des Tests

Notre strategie suit la pyramide des tests classique :

*test<sub>pyramid</sub>Pyramidedestests*

1. **Tests Unitaires (Base)** : Testent les composants individuels en isolation
2. **Tests d'Integration (Milieu)** : Verifient l'interaction entre les modules
3. **Tests E2E (Sommet)** : Valident les parcours utilisateur complets

### 10.3 Tests Unitaires

#### 10.3.1 Backend - JUnit et Mockito

Les tests unitaires du backend couvrent :

- **Services** : Logique metier (AuthService, ProductService, OrderService, etc.)
- **Utilitaires** : Fonctions de generation de tokens, validation, etc.
- **Entites** : Methodes de calcul et validations

**Outils utilises :**

- JUnit 5 pour les assertions et le cycle de vie des tests
- Mockito pour le mocking des dependances
- Spring Boot Test pour le contexte d'application

### 10.3.2 Frontend - Jasmine et Karma

Les tests unitaires Angular couvrent :

- **Services** : Appels HTTP, gestion d'état
- **Composants** : Logique de presentation, interactions
- **Pipes et Directives** : Transformations de donnees

### 10.3.3 Module IA - Pytest

Les tests du module Python couvrent :

- Verification du chargement du modele
- Tests de prediction avec images de reference
- Validation des endpoints API (health, predict)

## 10.4 Tests d'Integration

### 10.4.1 Tests API

Les tests d'integration verifient le bon fonctionnement des endpoints REST :

TABLE 10.1 – Resume des tests API

Module	Nombre de tests	Reussis
Authentication	8	8
Products	12	12
Cart	6	6
Orders	10	10
Payments	4	4
Delivery	8	8
Chat	6	6
AI	4	4
<b>Total</b>	<b>58</b>	<b>58</b>

### 10.4.2 Tests avec Postman

Une collection Postman complete a ete creee pour tester manuellement et automatiquement toutes les API. La collection comprend :

- Variables d'environnement (URLs, tokens)
- Pre-request scripts pour l'authentification automatique
- Tests automatises pour valider les reponses
- Collection Runner pour l'execution en lot

## 10.5 Tests d'Acceptation Utilisateur

### 10.5.1 Scenarios de Test

Les scenarios suivants ont ete valides avec des utilisateurs reels :

TABLE 10.2 – Scenarios de test utilisateur

Scenario	Resultat
Un utilisateur s'inscrit, verifie son email et se connecte	✓
Un utilisateur recherche et filtre des produits par categorie	✓
Un utilisateur ajoute des produits au panier et passe commande	✓
Un utilisateur effectue un paiement via Stripe	✓
Un utilisateur utilise l'IA pour identifier une piece	✓
Un utilisateur suit sa livraison en temps reel	✓
Un utilisateur contacte le support via chat	✓
Un admin gere les produits et commandes	✓

## 10.6 Couverture de Code

TABLE 10.3 – Couverture de code par module

Module	Lignes	Branches	Fonctions
Backend Services	78%	72%	85%
Backend Controllers	82%	75%	90%
Frontend Services	75%	68%	80%
Module IA	70%	65%	75%

## 10.7 Validation des Fonctionnalités

## 10.8 Conclusion

La strategie de tests mise en place nous a permis d'assurer un niveau de qualite eleve pour notre plateforme. Avec un taux de reussite de 100% sur les tests automatises et une validation complete des scenarios utilisateur, nous avons confiance dans la stabilite et la fiabilite de l'application.

TABLE 10.4 – Validation par plateforme

Module	Web	Mobile	Admin
Authentification	✓	✓	✓
Catalogue produits	✓	✓	✓
Panier et commandes	✓	✓	✓
Paiement Stripe	✓	✓	-
Module IA	✓	✓	-
Chat support	✓	✓	✓
Suivi livraison	✓	✓	✓

# Chapitre 11

## Conclusion et Perspectives

### 11.1 Synthèse des Réalisations

Ce projet de fin d'études a abouti à la conception et au développement d'une plateforme e-commerce intelligente pour les pièces détachées automobiles, intégrant des technologies d'Intelligence Artificielle innovantes.

#### 11.1.1 Objectifs Atteints

Nous avons réussi à atteindre l'ensemble des objectifs fixés au début du projet :

TABLE 11.1 – Bilan des objectifs

Objectif	Statut
Système d'authentification sécurisé (JWT, RBAC)	✓
Catalogue de produits avec recherche et filtres	✓
Gestion du panier et des commandes	✓
Intégration du paiement Stripe	✓
Module IA de reconnaissance d'images (94% précision)	✓
Système de recommandation personnalisé	✓
Chat de support en temps réel	✓
Suivi de livraison avec cartographie	✓
Application mobile utilisateur	✓

#### 11.1.2 Métriques du Projet

#### 11.1.3 Fonctionnalités Clés Développées

##### Module d'Intelligence Artificielle

Le module IA représente l'innovation principale de notre plateforme :

- **Reconnaissance d'images** : Classification de 50 types de pièces auto avec une précision de 94.2%

TABLE 11.2 – Statistiques du projet

Métrique	Valeur
Nombre de sprints	6
Durée totale	12 semaines
User Stories livrées	40
Points Story réalisés	197
Entités backend	30
Endpoints API	85+
Composants frontend	50+
Tests automatisés	58
Couverture de code moyenne	76%

- **Analyse de symptômes** : Suggestions de pièces basées sur la description de problèmes
- **Recommandations personnalisées** : Algorithme hybride combinant filtrage collaboratif et content-based

### Système de Livraison Intelligent

- Simulation de mouvement du livreur en temps réel
- Intégration OpenRouteService pour les itinéraires
- Visualisation sur carte Leaflet
- Notifications WebSocket

### Architecture Robuste

- Architecture microservices avec séparation Backend/IA
- API RESTful sécurisée
- Base de données PostgreSQL optimisée
- Frontend Angular 18 avec Signals

## 11.2 Défis Techniques Surmontés

### 11.2.1 Défis Majeurs

## 11.3 Limitations Actuelles

Malgré les fonctionnalités implémentées, certaines limitations persistent :

- **Paiements simulés** : Le système utilise le mode test de Stripe, les vrais paiements nécessiteraient une validation supplémentaire.



TABLE 11.3 – Défis techniques rencontrés

Défi	Solution Adoptée
Précision du modèle IA	Utilisation de EfficientNetB0 pré-entraîné avec fine-tuning
Communication temps réel	WebSocket avec STOMP pour chat et suivi
Géocodage d'adresses en Tunisie	Base de coordonnées locales + fallback ORS
Sécurisation des paiements	Mode test Stripe + webhooks sécurisés
Performances avec gros volumes	Pagination, lazy loading, indexation BDD

- **Livraison simulée** : Le mouvement du livreur est simulé, l'intégration avec de vrais livreurs GPS reste à faire.
- **Dataset IA limité** : Le modèle est entraîné sur 50 classes, une extension serait nécessaire pour couvrir toutes les pièces.

## 11.4 Perspectives d'Amélioration

### 11.4.1 Court Terme (3-6 mois)

#### 1. Passage en production

- Activation du mode live Stripe
- Déploiement sur serveur dédié avec HTTPS
- Configuration des environnements de production

#### 2. Amélioration du modèle IA

- Extension du dataset à 200+ classes
- Entraînement avec augmentation de données
- Optimisation pour mobile (TensorFlow Lite)

#### 3. Intégration livreurs réels

- Application mobile pour livreurs
- Tracking GPS temps réel
- Optimisation des itinéraires

### 11.4.2 Moyen Terme (6-12 mois)

#### 1. Amélioration application mobile

- Optimisation des performances Kotlin

- Notifications push Firebase
- Mode hors ligne pour le catalogue

## 2. Chatbot IA

- Assistant conversationnel pour le diagnostic
- Intégration NLP avancé
- Support multilingue (Français, Arabe, Anglais)

## 3. Marketplace multi-vendeurs

- Intégration de vendeurs tiers
- Système de commissions
- Tableau de bord vendeur

### 11.4.3 Long Terme (12+ mois)

#### 1. Expansion régionale

- Déploiement multi-pays (Maghreb)
- Adaptation aux réglementations locales
- Réseau de partenaires logistiques

#### 2. IA prédictive

- Prédiction des besoins de maintenance
- Alertes personnalisées basées sur le véhicule
- Analyse de la durée de vie des pièces

#### 3. Réalité augmentée

- Identification de pièces via caméra AR
- Guide d'installation visuel
- Vérification de compatibilité 3D

## 11.5 Compétences Acquises

Ce projet nous a permis de développer et renforcer de nombreuses compétences :

### 11.5.1 Compétences Techniques

- Développement full-stack (Angular, Spring Boot)
- Intelligence Artificielle et Machine Learning
- Architecture microservices
- Bases de données relationnelles
- API RESTful et WebSocket
- Intégrations tierces (Stripe, ORS)

### 11.5.2 Compétences Méthodologiques

- Méthodologie Scrum
- Gestion de projet agile
- Tests et assurance qualité
- Documentation technique

### 11.5.3 Compétences Transversales

- Travail en équipe
- Résolution de problèmes complexes
- Communication technique
- Gestion du temps et des priorités

## 11.6 Conclusion Générale

Ce projet de fin d'études nous a permis de mettre en pratique les connaissances acquises durant notre formation et d'explorer de nouvelles technologies innovantes. La plateforme développée répond aux besoins identifiés et propose une solution originale grâce à l'intégration de l'Intelligence Artificielle.

Le développement en suivant la méthodologie Scrum nous a permis de livrer un produit fonctionnel et de qualité, capable d'évoluer pour répondre aux besoins futurs du marché des pièces automobiles.

Cette expérience constitue une base solide pour notre carrière professionnelle et nous a préparés aux défis du développement logiciel moderne.

# Annexe A

## Annexe A : Schema de Base de Donnees

### A.1 Diagramme Entite-Relation

*erd<sub>c</sub>completeDiagrammeentite – relationcompletde labasededonnees*

### A.2 Description des Tables Principales

#### A.2.1 Table Users

TABLE A.1 – Structure de la table users

Colonne	Type	Description
id	UUID	Identifiant unique
email	VARCHAR(255)	Adresse email (unique)
password	VARCHAR(255)	Mot de passe hashe
full_name	VARCHAR(255)	Nom complet
phone	VARCHAR(20)	Numero de telephone
role_id	UUID	Reference vers le role
is_active	BOOLEAN	Compte actif

#### A.2.2 Table Products

TABLE A.2 – Structure de la table products

Colonne	Type	Description
id	UUID	Identifiant unique
name	VARCHAR(255)	Nom du produit
description	TEXT	Description detaillee
price	DECIMAL(10,2)	Prix unitaire
stock	INTEGER	Quantite en stock
category_id	UUID	Reference vers la categorie
brand_id	UUID	Reference vers la marque

### A.2.3 Table Orders

TABLE A.3 – Structure de la table orders

Colonne	Type	Description
id	UUID	Identifiant unique
user_id	UUID	Reference vers l'utilisateur
total_price	DECIMAL(10,2)	Prix total de la commande
status	VARCHAR(50)	Statut de la commande
delivery_address	TEXT	Adresse de livraison
tracking_number	VARCHAR(50)	Numero de suivi

# Annexe B

## Annexe B : Guide d'Installation

### B.1 Prerequis

Pour installer et executer le projet, les outils suivants sont necessaires :

- Java 17 ou superieur
- Node.js 18 ou superieur
- Python 3.10 ou superieur
- PostgreSQL 15 ou superieur
- Maven 3.8 ou superieur
- Git

### B.2 Installation du Backend

1. Cloner le repository du projet
2. Configurer les variables d'environnement dans application.properties
3. Creer la base de donnees PostgreSQL
4. Executer mvn clean install pour compiler
5. Lancer avec mvn spring-boot :run

### B.3 Installation du Frontend

1. Naviguer vers le dossier frontend-web
2. Executer npm install pour installer les dependances
3. Lancer avec ng serve pour le developpement
4. Compiler avec ng build --configuration production pour la production

### B.4 Installation du Module IA

1. Naviguer vers le dossier ai-module

- 
2. Créer un environnement virtuel Python
  3. Installer les dépendances avec `pip install -r requirements.txt`
  4. Lancer avec `uvicorn src.api.main :app --port 5000`