

République Tunisienne  
Ministère de l'Enseignement  
Supérieur  
et de la Recherche Scientifique  
Université de Sousse



Institut Supérieur des  
Sciences Appliquées  
et de Technologie  
de Sousse



# DÉPARTEMENT INFORMATIQUE

## RAPPORT DE PROJET DE FIN D'ÉTUDES

Présenté pour l'obtention du :  
**Diplôme National d'Ingénieur en Informatique**

*Option : Génie Logiciel - Architecture des Logiciels*

**Conception et Développement d'un  
Marketplace avec un backoffice OMS et un  
workflow DevSecOps**

Réalisé par :

**Ameni Gharbi**

Soutenu le **-/10/2025** devant le jury composé de :

Président : M -

ISSAT de Sousse

Examinateur : M -

ISSAT de Sousse

Encadreur Académique : M.Houssem Chtioui

ISSAT de Sousse

Encadreur Industriel : M. Sabri Zouari

Kriga Technologies

**Année Universitaire : 2024/2025**

*Code du sujet : -*



## *Dédicace*



# Introduction générale

Au début du XXI<sup>e</sup> siècle, le développement des technologies s'est beaucoup accéléré mais le développement exponentiel se poursuit sans relâche aujourd'hui.

Au cours des deux dernières années, la transformation numérique s'est intensifiée, notamment à cause de la pandémie, poussant de nombreux secteurs à adopter des solutions numériques pour améliorer leur efficacité et garantir leur viabilité dans un environnement en constante évolution.

Le commerce de détail subit des changements importants en raison de l'augmentation de l'activité en ligne et de l'émergence de modèles de vente innovants. Les mêmes trains restent quel que soit le format de magasin, qu'il s'agisse d'un point de vente physique ou d'un point de vente virtuel proposant des produits variés disponibles rapidement et efficacement.

Les deux derniers facteurs dépendants des distributeurs qui jouent un rôle de plus en plus essentiel dans cet égard en général. La plupart des distributeurs tunisiens s'appuient encore sur des méthodes traditionnelles utilisant largement des supports papier ou des outils peu adaptés pour gérer les flux de marchandises. Bien que ces méthodes fonctionnant à petite échelle elles deviennent rapidement dépassées avec l'augmentation des commandes dans des environnements commerciaux de plus en plus complexes. Cela freine considérablement la croissance des distributeurs et commerces qu'ils approvisionnent de manière significative.

Dans ce contexte Kamioun une startup tunisienne spécialisée dans la distribution de produits de grande consommation cherche à transformer le secteur avec une plate-forme mobile-first qui sert de guichet unique pour les petits détaillants. Plus de 1 600 produits alimentaires et produits de nettoyage etc .. sont disponibles sur l'application permettant aux détaillants de gérer leur approvisionnement de façon vraiment plus fluide et tout à fait efficace..

Kamioun permet aux utilisateurs de passer des commandes à tout moment avec une livraison rapide en 24 heures et des prix stables. Cependant, l'entreprise fait face à des défis tels que la stagnation du marché et une lente croissance du nombre d'utilisateurs, avec des dépenses stagnantes sur la plateforme. Pour se démarquer de la concurrence, Kamioun doit offrir une expérience innovante. Les détaillants montrent un intérêt pour une plateforme centralisée avec une large gamme de produits, similaire au succès de Jumia en Tunisie. Face à ces défis notre projet vise à créer une solution stratégique en intégrant un modèle de marché qui transforme Kamioun en plateforme.

Ce rapport présente tout travail réalisé sous plusieurs angles dans ce projet. Il est divisé en sept chapitres décrits ainsi le premier chapitre présente sommairement le contexte du projet en examinant l'étude préliminaire.

( à completer)

En fin de compte, une conclusion générale récapitule le travail accompli et suggère des pistes d'amélioration pour le développement futur de la plateforme Kamioun, d'une manière ou d'une autre.

# Chapitre 1

## Contexte général

### Introduction

Nous commençons ce chapitre par la présentation de l'organisme d'accueil "Kriga Technologies". Ensuite, nous exposons le problème identifié ainsi que la solution proposée suivis par une étude de l'existant afin de déterminer la valeur ajoutée et de mettre en évidence les insuffisances. Enfin, nous concluons ce chapitre par la description de la méthodologie de gestion de notre projet.

#### 1.1. Présentation de l'organisme d'accueil

Kamioun est une startup tunisienne spécialisée dans la distribution de produits de grande consommation. Elle propose une application mobile intuitive permettant aux petits détaillants d'accéder à plus de 1 600 produits du quotidien, de gérer facilement leurs stocks et de passer commande à tout moment, 24h/24. Grâce à une logistique optimisée, Kamioun assure une livraison rapide en 24 heures, avec la possibilité pour le client de choisir la date qui lui convient.

L'entreprise se distingue également par des prix stables, qui aident les détaillants à mieux gérer leur fonds de roulement et à planifier leurs achats en toute sérénité.

Il est à noter que l'entreprise d'accueil, Kriga Technologie, fait partie intégrante de la même structure que Kamioun. Tandis que Kamioun est dédiée aux opérations de distribution, Kriga constitue le pôle technologique (IT) de l'entreprise.



## 1.2. Présentation du projet

Nous détaillons dans cette section la problématique du projet, la solution proposée et les objectifs envisagés.

### 1.2.1. Cadre du projet

Le projet s'inscrit dans une transformation majeure du modèle économique de Kamioun, passant d'une plateforme de distribution centralisée à une marketplace B2B connectant directement les commerçants avec un réseau élargi de fournisseurs. Actuellement, Kamioun opère comme un distributeur unique, proposant environ 1 500 références via son application mobile. La nouvelle architecture envisagée intégrera plusieurs partenaires (Sokadi, AgriTable, etc.) et marques locales (Jadida, Bonna, Président, etc.), permettant d'élargir significativement le catalogue tout en maintenant Kamioun comme acteur clé de l'écosystème.

Cette évolution nécessitera une refonte technique de la plateforme, avec notamment la mise en place d'un système d'ordre management (OMS) capable de gérer des flux multi-fournisseurs, une interface unifiée pour les commerçants, et des outils analytiques pour les partenaires.

### 1.2.2. Problématique

Kamioun rencontre trois défis structurels majeurs :

Une attractivité commerciale limitée : Avec seulement 1 500 SKUs disponibles (sur un marché potentiel de 30 000+), la plateforme ne couvre pas les besoins complets des commerçants. Les prix, bien que compétitifs, ne suffisent pas à compenser la faible variété de l'offre.

Des relations client-fournisseur difficiles à disrupter : Les commerçants tunisiens entretiennent des liens historiques avec leurs fournisseurs, qui proposent souvent du crédit et une livraison directe en magasin. Ces habitudes rendent complexe l'adoption d'une solution purement digitale.

Des limites techniques : L'application actuelle, bien qu'intuitive, ne permet pas

de gérer des commandes multi-fournisseurs ni d'offrir des fonctionnalités avancées comme la comparaison de prix en temps réel ou le suivi unifié des livraisons.

Ces contraintes expliquent la stagnation du nombre de nouveaux utilisateurs et la faible progression du panier moyen des clients existants.

### 1.2.3. Solution proposée

L'opportunité du Marketplace B2B

La transition vers une marketplace B2B représente une réponse adaptée à ces enjeux. Cette nouvelle plateforme permettra :

Un élargissement radical de l'offre : Jusqu'à 30 000 références via l'intégration de fournisseurs tiers, tout en conservant les produits Kamioun.

Une expérience client optimisée : Processus de commande unifié, comparaison des prix, gestion centralisée des livraisons et possibilité de crédit différé.

Un modèle gagnant-gagnant : Pour les commerçants (choix élargi, transparence), les fournisseurs (accès à de nouveaux clients sans investissement digital) et Kamioun (commission sur les transactions).

Les retours préliminaires des utilisateurs confirment le potentiel disruptif de cette approche, plusieurs commerçants comparant déjà Kamioun à une alternative B2B à Jumia.

### 1.2.4. Objectifs :

#### Objectifs Techniques

La solution technique vise à créer une plateforme performante et évolutive permettant :

— **Une offre produit élargie via un réseau multi-partenaires :**

- Intégration transparente de plusieurs fournisseurs (Kamioun, Sokadi, Agri-Table, etc.)
- Catalogue unifié permettant aux commerçants de choisir leurs produits parmi différents partenaires et dépôts d'approvisionnement
- Optimisation automatique de la chaîne logistique en fonction de la localisation et des stocks disponibles

— **Une expérience utilisateur fluide et efficace :**

- Processus de commande simplifié (réduction des étapes)
- Outils de gestion intégrés (suivi des stocks, historique des achats)
- Interface intuitive adaptée aux besoins des petits commerçants

## Objectifs Commerciaux

- Atteindre 15% de parts de marché sur le segment des petits commerçants tunisiens en 24 mois
- Élargir le catalogue à plus de 30 000 références via les partenaires intégrés
- Fidéliser les utilisateurs avec des outils de gestion simplifiés et des prix compétitifs

## Objectifs Opérationnels

- Disponibilité élevée : Taux de disponibilité > 99,5%
- Livraison rapide : 90% des commandes livrées en moins de 24h
- Performance optimisée : Temps de réponse minimal pour une expérience utilisateur réactive

Cette approche technique permet de répondre aux besoins du marché tout en garantissant stabilité, rapidité et simplicité d'utilisation.

### 1.3. Etude de l'existant

Cette section représente une étude des solutions existantes sur le marché pour comprendre le domaine .

Sur le plan international, plusieurs initiatives similaires à celle de Kamioun marketplace ont vu le jour et rencontrent un succès notable dans des marchés émergents à forte croissance, en particulier au Moyen-Orient et en Asie du Sud.

"Qawafel est une marketplace B2B qui simplifie le commerce entre les magasins et les producteurs de produits alimentaires périssables à travers le Royaume d'Arabie Saoudite."<sup>[1]</sup> Cette plateforme se spécialise dans la logistique alimentaire et optimise les flux entre petits commerces et producteurs, tout en répondant aux spécificités du marché saoudien. La figure 1.1 présente l'application web de "Qawafel".



FIGURE 1.1 – Application QUAWFEL

"Sary est la plateforme B2B leader du commerce et des services dans le Golfe, et ShopUp est la plus grande plateforme de commerce interentreprises au Bangladesh. Ces deux entités ont fusionné pour former le groupe SILQ, la plus grande plateforme B2B reliant le Golfe et l'Asie émergente, au service des marchés de consommation les plus dynamiques au monde." [2]

La figure 1.2 présente l'application web de "Sary".

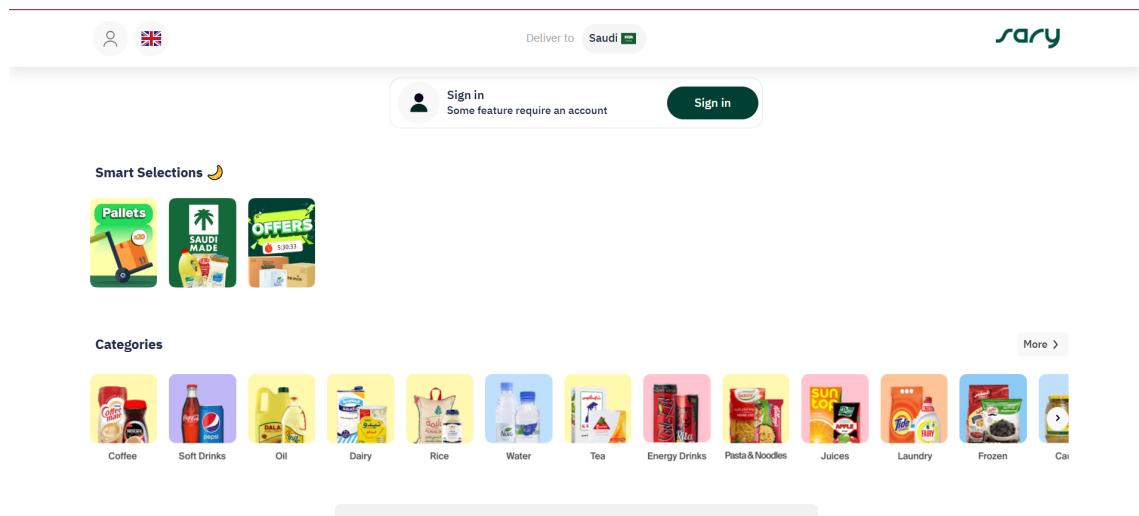


FIGURE 1.2 – Application Sary

Le tableau ci-dessous présente une comparaison des principales marketplaces afin de mieux comprendre leurs différences et de mettre en évidence la proposition de valeur unique de Kamioun sur le marché tunisien.

TABLE 1.1 – Comparaison des plateformes B2B pour détaillants

Critère	Sary (Moyen-Orient)	Qawafel (Arabie Saoudite)
Logistique	Livraison externalisée (zones rurales mal couvertes)	Livraison réfrigérée mais limitée aux produits frais
Catalogue	Large catalogue mais fragmenté	Catalogue restreint (produits frais uniquement)
Expérience UX	Processus de commande lent	Interface complexe
Modèle économique	Dépendance au crédit (risque d'impayés)	Marge limitée aux produits frais

## 1.4. Chronologie

Les différentes étapes de réalisation de ce projet pendant la durée du stage de 5 mois qui s'étend du 1er Mars jusqu'au 31 Juillet sont représentées par le diagramme de Gantt de la figure suivante.

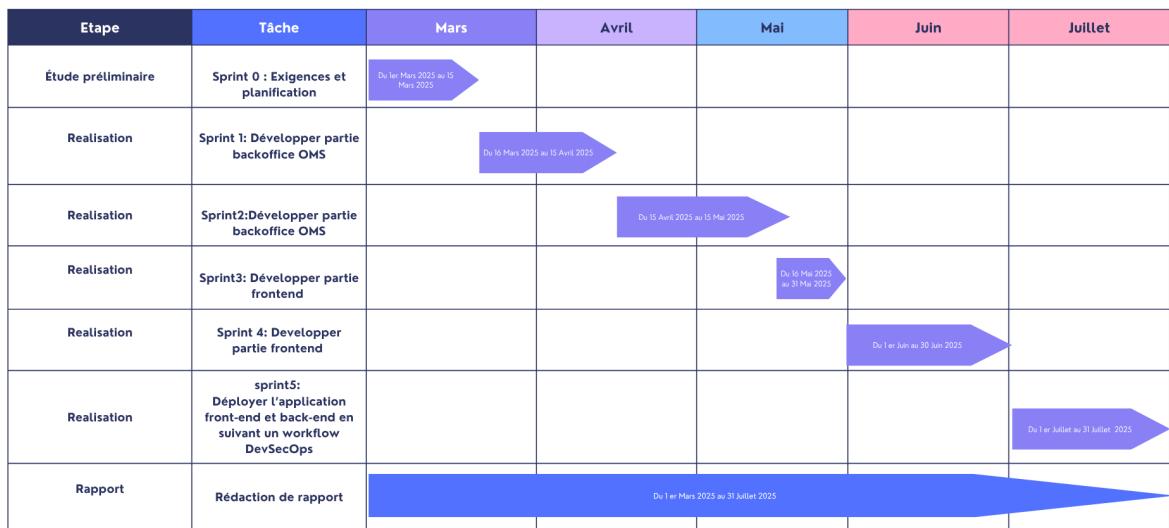


FIGURE 1.3 – Diagramme de Gantt représentant la planification du projet

## 1.5. Méthodologie adoptée

Le choix de la méthode de réalisation d'un projet est très important, car il influence le cycle de vie du projet qui peut avoir un impact sur la qualité du système logiciel. Chez Kamioun, nous avons adopté la méthodologie Scrum pour gérer nos projets de manière agile, ce qui nous permet de nous adapter rapidement aux évolutions du marché et aux retours des utilisateurs. Nous travaillons par sprints mensuels, ce qui offre à l'équipe l'occasion de se concentrer sur des objectifs précis et de livrer des résultats tangibles à la fin de chaque cycle.

Chaque jour, l'équipe se retrouve lors de courtes réunions (daily meetings) pour faire le point sur les avancées, identifier les obstacles éventuels et s'assurer que tout le monde est sur la même longueur d'onde concernant les priorités. À la fin de chaque mois, nous organisons une journée de démonstration pour mettre en lumière les progrès réalisés. Cela permet à l'équipe et aux parties prenantes de découvrir les résultats concrets du travail accompli.

Cette méthode permet à Kamioun de rester réactif, transparent et toujours en quête d'amélioration continue dans ses projets.

La figure 1.4 présente la méthodologie scrum.

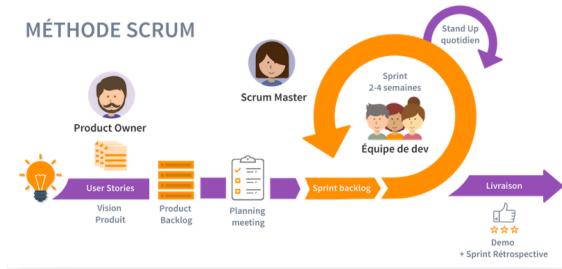


FIGURE 1.4 – Methodologie scrum

## Conclusion

A travers ce chapitre, nous avons présenté l'organisme d'accueil Kriga technologies. Nous avons également exposé la problématique et la solution proposée. Une étude de l'existant est aussi incluse dans ce chapitre. Nous avons enfin défini le processus de développement du logiciel adopté. Dans le prochain chapitre nous décrivons les fonctionnalités que notre logiciel doit fournir pour répondre aux besoins du client.

# **Chapitre 2**

## **Sprint 0**

### **Introduction**

Ce chapitre présente le sprint zéro, qui constitue la phase de départ de notre application à développer. Tout d'abord, nous identifions les acteurs de notre application. Ensuite, nous listons les besoins fonctionnels et non fonctionnels du système, puis nous détaillons le travail selon la méthodologie choisie dans le chapitre précédent. Enfin, nous donnons un bref aperçu du matériel de base ainsi que des technologies utilisées pour la mise en place de l'environnement de travail.

## 2.1. Capture des besoins

### 2.1.1. Identification des acteurs

TABLE 2.1 – Les acteurs de notre système

Acteur	Rôle
Détaillant	<ul style="list-style-type: none"> <li>- Authentification</li> <li>- Navigation générale</li> <li>- Gestion du compte</li> <li>- Découverte des produits</li> <li>- Gestion des favoris</li> <li>- Interaction produits</li> <li>- Passation de commande</li> <li>- Consultation historique</li> </ul>
Administrateur	<ul style="list-style-type: none"> <li>— Gestion des commandes (validation/modification)</li> <li>— Suivi des activités (notifications)</li> <li>— Administration système (crud bannières etc ..)</li> <li>— tableau de bord d'analyse</li> <li>— Gestion des stocks (commandes fournisseurs)</li> </ul>

### 2.1.2. Identification des besoins

Notre application doit répondre aux attentes de l'ensemble des utilisateurs. Nous présentons ci-après les besoins fonctionnels propres à chaque acteur, ainsi que les besoins non fonctionnels partagés par tous.

#### a. Besoins fonctionnels

1. **Authentification** : Chaque utilisateur doit pouvoir se connecter à l'application via un identifiant et un mot de passe sécurisés.
2. **Gestion du compte** : Les utilisateurs doivent pouvoir consulter et modifier leurs informations personnelles.
3. **Navigation produit** : Le détaillant doit pouvoir parcourir les catégories de produits, consulter les détails de chaque article, et visualiser les produits proposés par un partenaire ou une marque spécifique.
4. **Gestion des favoris** : Le détaillant peut ajouter ou retirer des produits ainsi que des partenaires de sa liste de favoris.
5. **Passation de commandes** : Le détaillant peut ajouter des articles au panier et confirmer une commande.

6. **Suivi des commandes** : Chaque utilisateur peut consulter l'historique et le statut de ses commandes.
7. **Pages informatives** : Les utilisateurs doivent pouvoir consulter une page "À propos" présentant l'équipe et la vision de Kamioun, ainsi qu'une page "Contact" pour entrer en relation avec l'équipe support.
8. **Administration du système** : L'administrateur doit pouvoir gérer les commandes, réservations, les bannières, les comptes des utilisateurs, les états et les status des commandes.
9. **Gestion des stocks** : L'administrateur peut consulter et passer des commandes depuis plusieurs fournisseurs.
10. **Analyse des performances** : L'administrateur peut visualiser les statistiques de vente et suivre les performances des partenaires.

**b. Besoins non fonctionnels** Notre système doit respecter les exigences qualitatives et contraintes techniques suivantes :

1. **Ergonomie** :

- Interfaces utilisateur intuitives et conformes aux standards UX/UI
- Navigation fluide avec parcours utilisateur optimisé
- Design responsive adapté aux supports mobiles et desktop

2. **Performance** :

- L'application doit offrir un temps de réponse minimal. Elle doit se charger rapidement, tant au niveau des interfaces que des données.

3. **Maintenabilité & Évolutivité** :

- Notre projet doit respecter les normes architecturales afin de faciliter la maintenance et garantir l'évolution du système lors de l'ajout ou de la modification des besoins.

4. **Sécurité** :

- Double authentification (JWT + Sessions chiffrées)
- Protection contre le BrutForce.
- Stockage sécurisé dans Supabase.
- Hashage bcrypt et complexité de mot de passes.
- Sécurité des conteneurs docker.

## 2.2. Pilotage du projet avec scrum

### 2.2.1. Identification de l'équipe scrum

L'équipe joue un rôle essentiel dans la méthode Scrum, car elle contribue à maximiser la productivité et la flexibilité. Dans le cadre de notre projet, l'équipe se compose de :

TABLE 2.2 – Composition de l'équipe Scrum

Rôle Scrum	Nom	Responsabilités principales
CEO	Fares Belghith	<ul style="list-style-type: none"> <li>— Vision stratégique du produit</li> <li>— Allocation des ressources</li> <li>— Décisions exécutives</li> </ul>
Product Owner	Sabri Zouari	<ul style="list-style-type: none"> <li>— Définition du Product Backlog</li> <li>— Priorisation des fonctionnalités</li> <li>— Validation des livrables</li> </ul>
Scrum Master	Abderrahman Jalled	<ul style="list-style-type: none"> <li>— Facilitation des cérémonies Scrum</li> <li>— Résolution des blocages</li> <li>— Garant de la méthodologie</li> </ul>
Développeur Mobile	Sami Ayachi	<ul style="list-style-type: none"> <li>— Développement applications mobiles</li> <li>— Intégration API</li> <li>— Optimisation performances</li> </ul>
Ingénieur Devops	Rania Brahmi	<ul style="list-style-type: none"> <li>— Gestion infrastructure cloud</li> <li>— Pipelines CI/CD</li> <li>— Monitoring et déploiements</li> </ul>
Développeur Backend	Yassine Medhioub	<ul style="list-style-type: none"> <li>— Développement API</li> <li>— Gestion base de données</li> <li>— Sécurité backend</li> </ul>
Data Analyst	Ahmed Elloumi	<ul style="list-style-type: none"> <li>— Analyse données métiers</li> <li>— Tableaux de bord</li> </ul>
Développeur Full Stack	Ameni Gharbi	<ul style="list-style-type: none"> <li>— Développement frontend/backend</li> <li>— Intégration systèmes</li> </ul>
UI/UX Designer	Chaima Arfaoui	<ul style="list-style-type: none"> <li>— Conception interfaces</li> <li>— Prototypage</li> </ul>

### 2.2.2. Le Backlog du produit

Le Product Backlog représente l'ensemble des fonctionnalités essentielles et des exigences techniques du produit, organisées sous forme de User Stories ou de Technical Stories. Il est construit de manière évolutive, en affinant progressivement les besoins identifiés.

Chaque User Story est structurée selon le format suivant :

*En tant que <rôle>, je veux <fonctionnalité ou action>, afin de <objectif ou valeur ajoutée>.*

Pour chaque User Story, plusieurs éléments sont pris en compte, notamment : le rang (priorité), l'estimation de l'effort requis, l'importance pour le produit, ainsi qu'une description claire.

Le tableau suivant illustre le Product Backlog de notre application et détaille les différentes User Stories identifiées. Les priorités y sont notées selon la légende suivante : ++ : priorité moyenne +++ : priorité élevée

TABLE 2.3 – Backlog produit de l'application Kamioun Marketplace

User Story	Je veux...	Pour que...	Priorité
En tant que détaillant	M'authentifier avec un identifiant et un mot de passe	Mes données soient sécurisées et accessibles uniquement par moi	+++
En tant que détaillant	Consulter et modifier mes informations personnelles	Mon compte reflète toujours mes données actuelles	++
En tant que détaillant	Parcourir les produits, consulter les détails et filtrer par marque ou partenaire	Trouver rapidement ce dont j'ai besoin	+++
En tant que détaillant	Ajouter ou retirer des produits et partenaires à mes favoris	Y accéder facilement plus tard	++
En tant que détaillant	Ajouter des produits au panier et passer commande	Finaliser mes achats efficacement	+++
En tant que détaillant	Consulter l'historique et le statut de mes commandes	Suivre leur avancement et les recevoir à temps	++
En tant que détaillant	Accéder à la page "About"	En savoir plus sur l'équipe Kamioun et sa vision	++
En tant que détaillant	Accéder à la page "Contact"	Poser mes questions ou signaler un problème à l'équipe support	++
En tant qu'admin	Gérer les commandes, réservations, comptes, états commandes, status commandes et bannières	Administrer efficacement le système	+++
En tant qu'admin	Consulter les stocks et passer des commandes auprès des fournisseurs	Assurer la disponibilité des produits	++
En tant qu'admin	Visualiser les statistiques de vente et les performances des partenaires	Analyser et améliorer les performances commerciales	++

### 2.2.3. Diagramme des cas d'utilisation global :

Dans ce qui suit, nous présentons le diagramme de cas d'utilisation global de notre application. Ce diagramme représente une vue d'ensemble du système, en modélisant ses principales fonctionnalités et les interactions entre les acteurs et le système.



FIGURE 2.1 – Diagramme de cas d'utilisation global

### 2.2.4. Planification des sprints

Une fois le backlog du produit finalisé, nous avons organisé la réunion de planification du sprint. Cette réunion avait pour objectif principal de construire le backlog de sprint à partir du backlog produit élaboré par le Product Owner. À l'issue de cette réunion, nous avons pu identifier les durées prévisionnelles pour chaque sprint ainsi que la répartition du travail sur plusieurs itérations.

Dans le cadre de notre projet, nous avons divisé le développement en trois releases, réparties comme suit :

Release 1 :

Sprint 1 : 1 mois

Sprint 2 : 1 mois

Release 2 :

Sprint 3 : 1 mois

Sprint 4 : 1 mois

Release 3 :

Sprint 5 : 1 mois

Le tableau ci-dessous présente la planification détaillée des sprints de notre projet

TABLE 2.4 – Planning détaillé des Releases et Sprints

<b>Release</b>	<b>Sprint</b>	<b>Contenu du sprint</b>	<b>Période</b>
Release 1	Sprint 1	— Gestion réservations clients — Commandes clients — Bannières — États commandes — Comptes utilisateurs — Statuts commandes	16 Mars – 16 Avril 2025
	Sprint 2	— Commandes fournisseurs — Dashboard d'analyse	17 Avril – 15 Mai 2025
Release 2	Sprint 3	— Authentification — Page d'accueil — About / Contact — Favoris produits / partenaires — Profil détaillant — Produits par marque — Détail bannières	16 Mai – 15 Juin 2025
	Sprint 4	— Page Shop — Détail produit — Panier — Confirmation commande — Historique commandes	16 Juin – 30 Juin 2025
Release 3	Sprint 5	— Déploiement Vercel — Automatisation — Monitoring — Qualité du code	1 Juillet – 31 Juillet 2025

## 2.3. Environnement de travail

### 2.3.1. Environnement Matériel :

Lors de la réalisation de ce projet, nous avons utilisés un ordinateur dont leur configuration sont présentées dans le tableau suivant :

TABLE 2.5 – Caractéristiques de l'ordinateur

Composant	Description
Disque dur	256 Go SSD
Processeur	Intel Core i7
Système d'exploitation	Windows 10

### 2.3.2. Environnement logiciel :

Dans cette partie, nous spécifions les logiciels utilisés, l'environnement de conception, l'environnement de développement, ainsi que l'environnement de la base de données utilisés lors de la réalisation du notre projet.

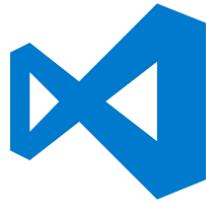
#### 1. Environnement de conception UML :

Le langage de modélisation unifié (UML) est un langage de modélisation standardisé utilisé pour visualiser la conception d'un système. StarUML prend en charge la plupart des diagrammes UML 2, notamment les diagrammes de classes, de packages, de structures composites, d'objets, de composants, de déploiement, de cas d'utilisation, de séquences, de communication, de synchronisation, de synthèse des interactions, d'états, d'activités, de flux d'informations et de profils.[3]



#### 2. Environnement de développement :

Visual Studio Code est un éditeur de code source léger, mais puissant, qui s'exécute sur votre bureau et est disponible pour Windows, macOS et Linux. Il est livré avec la prise en charge intégrée pour JavaScript, TypeScript et Node.js et dispose d'un riche écosystème d'extensions pour d'autres langages et environnements d'exécution.[4]

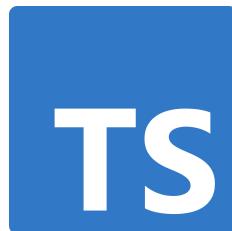


### 2.3.3. Technologies :

Dans cette partie, nous spécifions les technologies utilisées lors de la réalisation du notre projet.

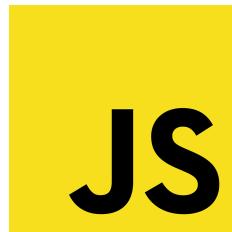
#### 1. TypeScript :

TypeScript est un langage de programmation fortement typé qui s'appuie sur JavaScript.[5]



#### 2. JavaScript :

JavaScript (souvent abrégé en « JS ») est un langage de script léger, orienté objet, principalement connu comme le langage de script des pages web.[6]



#### 3. MongoDB :

MongoDB a été créée en 2007 par deux développeurs qui souhaitaient suivre un nombre considérable – d'où son nom – de petites transactions dans le secteur de la diffusion publicitaire. La nouvelle base de données, initialement baptisée 10gen, stockait les données dans un simple « seau » de fichiers JSON, et était très rapidement

évolutive. Elle ne nécessitait pas de modèle de données complexe ni de simultanéité de transactions rigoureuse, car elle se contentait de compter les impressions publicitaires, et les enjeux étaient faibles.[7]



### 2.3.4. Frameworks/Libraries :

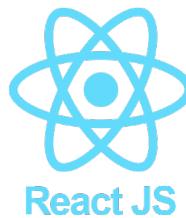
#### 1. NextJs :

Next.js est un framework React flexible qui vous fournit des éléments de base pour créer des applications Web rapides et complètes.[8]



#### 2. React :

React est une bibliothèque. Il vous permet d'assembler des composants, mais n'est pas prescriptif en ce qui concerne le routage ou le chargement de données.[9]



#### 3. Axios :

Axios est un client HTTP basé sur les promesses compatible avec node.js et les navigateurs. Il est isomorphe (c'est à dire qu'il peut opérer dans le navigateur et dans node.js avec le même code). Côté serveur, il utilise le module natif http de node.js, et côté client (navigateur) il utilise les XMLHttpRequest.[10]



#### 4. Tailwind :

Tailwind CSS est un framework CSS utilitaire entièrement compatible avec Next.js.[11]



#### 5. Prisma :

Prisma est un ORM qui simplifie l'utilisation des bases de données pour les applications Node.js et TypeScript.[12]



##### 2.3.5. Services/Outils :

###### 1. Supabase (Backend as a service) :

Supabase est un fournisseur de services backend sans serveur et une alternative open source à Firebase.[13]



###### 2. Vercel (Platform as a service) :

Créez et déployez sur le Cloud IA. Vercel fournit les outils de développement et l'infrastructure cloud nécessaires pour créer, faire évoluer et sécuriser un Web plus rapide et plus personnalisé.[14]



### 3. Sonarqube :

SonarQube Server automatise les examens de qualité et de sécurité du code et fournit des informations exploitables sur le code afin que les développeurs puissent se concentrer sur une création meilleure et plus rapide.[15]



### 4. Gitlab Ci :

GitLab CI/CD est un outil de développement logiciel qui permet aux organisations de mettre en œuvre des méthodologies « continues », notamment l'intégration continue (CI), la livraison continue (CD) et le déploiement continu (également abrégé en CD).[16]



### 5. Trivy :

Trivy est utilisé pour trouver des vulnérabilités (CVE) et des erreurs de configuration (IaC) dans les référentiels de code, les artefacts binaires, les images de conteneurs, les clusters Kubernetes, etc.[17]



### 6. Prometheus :

Un système de surveillance open source avec un modèle de données dimensionnel, un langage de requête flexible, une base de données de séries chronologiques efficace et une approche d'alerte moderne. [18]



Prometheus

### 7. Ansible :

Ansible automatise la gestion des systèmes distants et contrôle leur état souhaité.  
[19]



### 8. Grafana :

Grafana est une plateforme de visualisation de données interactive Open Source développée par Grafana Labs, qui permet aux utilisateurs de consulter leurs données via des graphiques unifiés dans un ou plusieurs tableaux de bord afin de mieux les interpréter et les comprendre. [20]



### 9. EmailJs :

EmailJS est un service qui vous permet d'envoyer des e-mails directement depuis JavaScript en connectant votre application côté client à des fournisseurs comme Gmail, Outlook ou Mailtrap. [21]



#### 10. Slack :

Slack est un espace qui optimise le travail pour ses millions d'utilisateurs.

Dans le monde entier, les entreprises à forte croissance optent pour Slack, une plateforme conversationnelle alimentée par l'IA, parce qu'elle rend le travail plus simple, plus agréable et plus productif.[22]



#### 11. Jira :

Jira est un outil de gestion de projets Agile qui prend en charge toute méthodologie Agile, qu'il s'agisse de Scrum, Kanban ou de votre propre création .[23]



### 2.4. Architecture générale de l'application :

Notre application mobile repose sur une **architecture trois tiers (3-Tier)** répartie comme suit :

#### Tier Présentation (Frontend)

- **Technologie** : React.js
- **Rôle** :
  - Interface utilisateur (UI)
  - Interaction avec l'utilisateur final
- **Communication** :

- Envoi de requêtes HTTP (REST/GraphQL)
- Vers le Backend (Next.js)

### Tier Application (Backend)

- **Technologie** : Next.js
- **Rôle** :
  - Réception des requêtes frontend
  - Traitement de la logique métier :
    - Authentification
    - Validation des données
    - Calculs métiers
  - Communication avec la base de données
  - Retour des réponses au frontend

### Tier Données (Persistence)

- **Technologie** : MongoDB (NoSQL)
- **Rôle** :
  - Stockage sécurisé des données
  - Gestion des opérations CRUD
  - Persistance de l'état de l'application

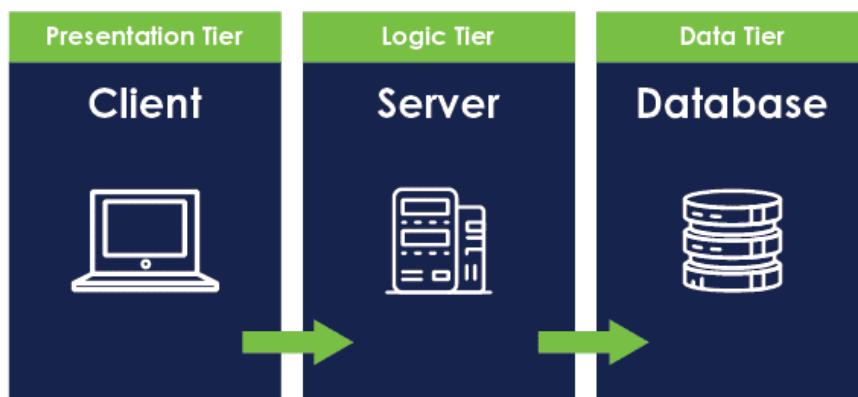


FIGURE 2.2 – Schéma d'architecture 3-Tier

## Conclusion :

Ce chapitre a structuré les fondements essentiels de notre projet à travers trois axes majeurs. Nous avons d'abord mené une analyse complète des besoins, identifiant les fonctionnalités clés et les contraintes techniques. Sur le plan méthodologique, nous avons constitué l'équipe projet, structuré le backlog priorisé et défini le découpage en sprints. L'étape finale a consisté à formaliser notre environnement technique et les outils logiciels. Ces travaux préparatoires établissent désormais le cadre opérationnel pour le développement effectif qui sera détaillé dans le chapitre suivant.



# Chapitre 3

## Sprint 1

### Introduction

Dans ce chapitre, nous présentons la réalisation du premier sprint, en organisant le travail sur trois phases principales qui sont l'analyse, la conception, et la réalisation.

#### 3.1. Backlog du sprint :

Le sprint est le cœur de Scrum. Il s'agit d'un bloc de temps durant lequel un incrément du produit sera réalisé. Tous les sprints d'une release ont une durée constante et ne se chevauchent jamais, c'est-à-dire qu'un sprint ne peut pas démarrer tant que le précédent n'est pas encore terminé. Avant de se lancer dans un sprint, l'équipe Scrum doit obligatoirement définir le but de ce dernier qui doit être un tableau descriptif qui précise la charge du travail pour chaque tâche en nombre de jours. Le tableau si dessous décrit les histoires de notre backlog du sprint :

TABLE 3.1 – Backlog de développement

ID	Histoires / User Stories	Estimation (jours)
US-01	Gestion des réservations clients	6
US-02	Gestion des commandes clients	6
US-03	Gestion des bannières promotionnelles	5
US-04	Gestion des états de commandes	5
US-05	Gestion des status de commandes	4
US-05	Gestion des comptes utilisateurs	4

## 3.2. Spécification fonctionnelle :

Dans cette partie nous présentons la phase d'analyse qui répond à la question « que fait le système ».

### 3.2.1. Diagramme de cas d'utilisation :

Les figures ci-dessous décrivent les diagrammes de cas d'utilisation du premier sprint.

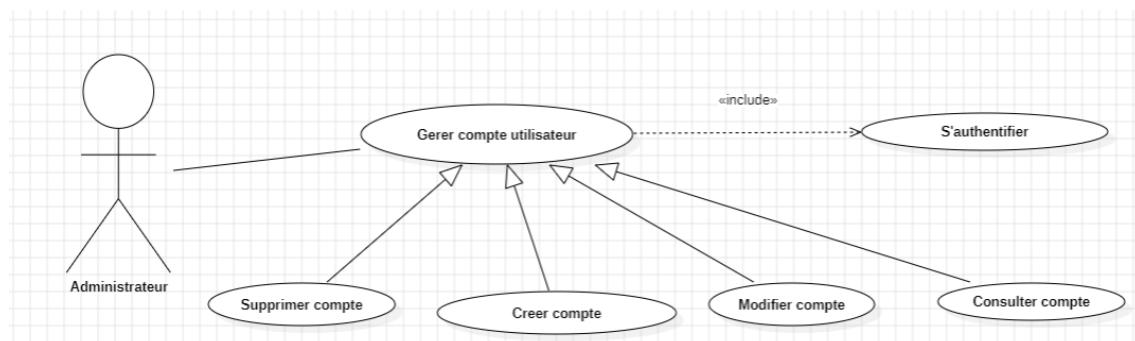


FIGURE 3.1 – Diagramme de cas d'utilisation "Gerer compte utilisateur"

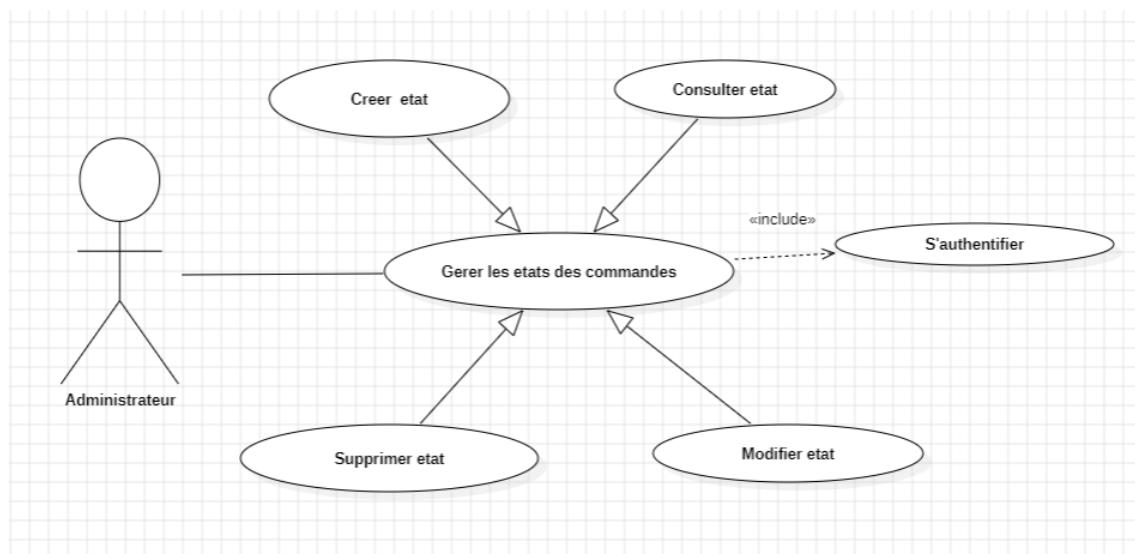


FIGURE 3.2 – Diagramme de cas d'utilisation "Gerer etat commande"

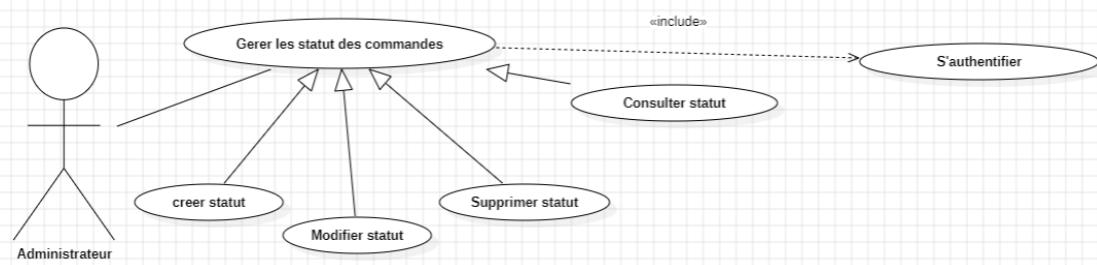


FIGURE 3.3 – Diagramme de cas d'utilisation "Gerer statut commande"

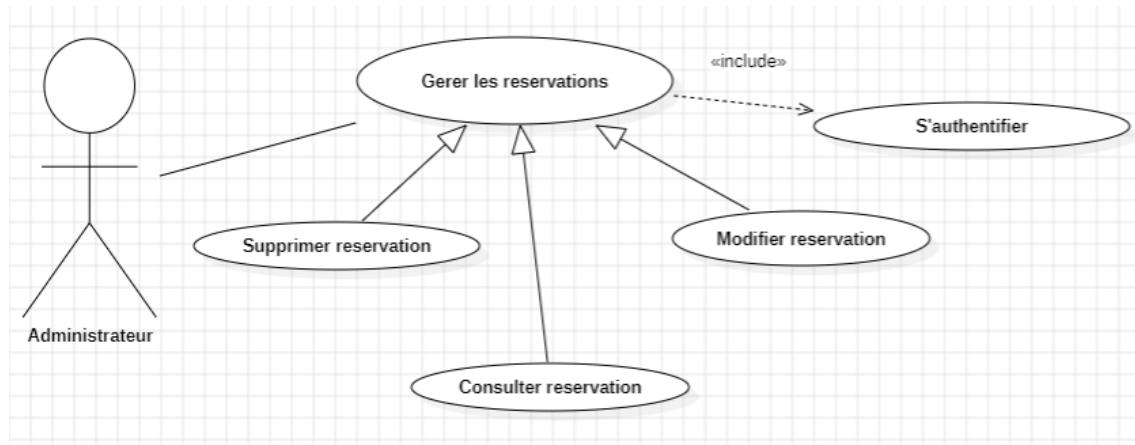


FIGURE 3.4 – Diagramme de cas d'utilisation "Gerer reservation"

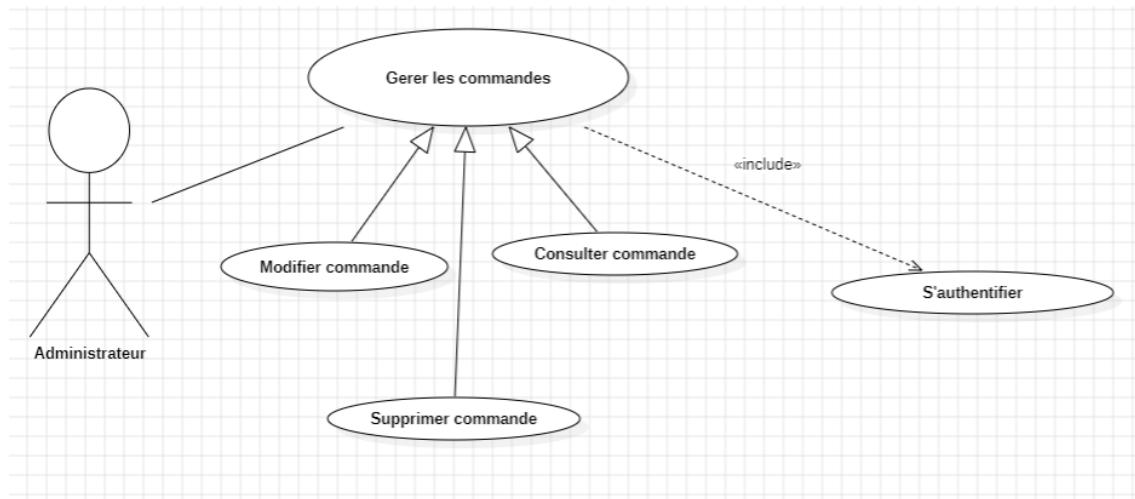


FIGURE 3.5 – Diagramme de cas d'utilisation "Gerer commande"

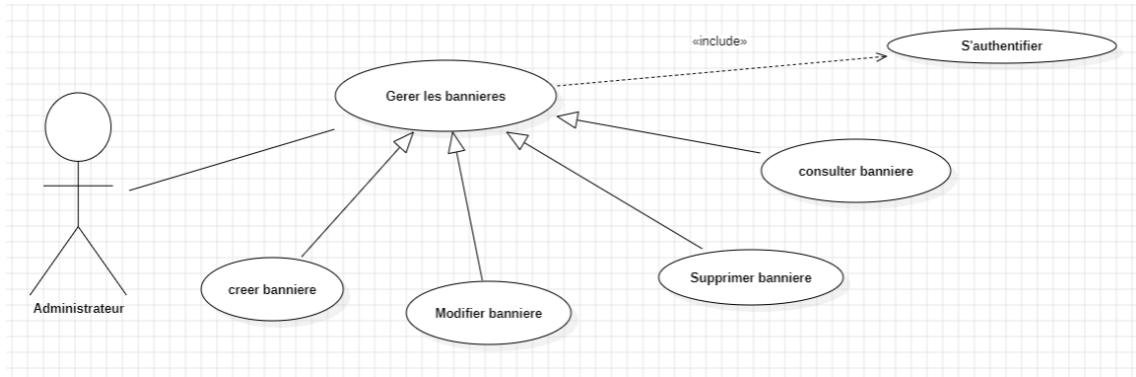


FIGURE 3.6 – Diagramme de cas d'utilisation "Gerer bannieres"

### 3.2.2. Description textuelle des cas d'utilisations :

#### 1- Description textuelle du cas d'utilisation "Creer compte détaillant" :

TABLE 3.2 – CU1.1 - Crédation de compte utilisateur

<b>SOMMAIRE D'IDENTIFICATION CU1.1 : Crédation de compte</b>	
<b>Acteur</b>	Administrateur
<b>Résumé</b>	Création d'un nouveau compte détaillant dans le système.
<b>Précondition</b>	Admin authentifié avec droits de création.
<b>Post-condition</b>	<ul style="list-style-type: none"> <li>— Nouveau compte créé dans la base</li> <li>— Documents sauvegardés dans Supabase</li> </ul>
<b>Scénario principal</b>	<ol style="list-style-type: none"> <li>1. Accès à la liste clients</li> <li>2. Click sur "Create New Customer"</li> <li>3. Remplissage du formulaire</li> <li>4. Upload CIN + Patente</li> <li>5. Validation → Sauvegarde données + images</li> </ol>
<b>Scénario d'exception</b>	<ul style="list-style-type: none"> <li>— Champs manquants → Erreur validation</li> <li>— Email/téléphone existant → Refus création</li> <li>— Échec upload → Annulation processus</li> </ul>

#### 2- Description textuelle du cas d'utilisation "Modifier compte détaillant" :

Le tableau ci-dessous décrit la description textuelle du cas d'utilisation "Modifier compte détaillant".

TABLE 3.3 – CU1.2 - Modification de compte utilisateur

<b>SOMMAIRE D'IDENTIFICATION CU1.2 : Modifier compte utilisateur</b>	
<b>Acteur</b>	Administrateur
<b>Résumé</b>	Mise à jour des informations d'un compte existant.
<b>Précondition</b>	<ul style="list-style-type: none"> <li>— Compte existant sélectionné</li> <li>— Admin avec droits de modification</li> </ul>
<b>Post-condition</b>	<ul style="list-style-type: none"> <li>— Données mises à jour</li> <li>— Nouveaux documents sauvegardés</li> </ul>
<b>Scénario principal</b>	<ol style="list-style-type: none"> <li>1. Sélection du client</li> <li>2. Click "Modifier"</li> <li>3. Mise à jour des champs</li> <li>4. Changement documents (optionnel)</li> <li>5. Validation → Update données</li> </ol>
<b>Scénario d'exception</b>	<ul style="list-style-type: none"> <li>— Champs obligatoires vides → Blocage</li> <li>— Nouveaux coordonnées déjà utilisées → Refus</li> <li>— Échec update Supabase → Rollback</li> </ul>

### 3- Description textuelle du cas d'utilisation "Creer banniere" :

Le tableau ci-dessous décrit la description textuelle du cas d'utilisation "Creer banniere".

TABLE 3.4 – CU2.1 - Création de bannière

<b>SOMMAIRE D'IDENTIFICATION CU2.1 : Creer de bannière</b>	
<b>Acteur</b>	Administrateur
<b>Résumé</b>	Ajout d'une nouvelle bannière publicitaire.
<b>Précondition</b>	Admin authentifié avec droits de création.
<b>Post-condition</b>	<ul style="list-style-type: none"> <li>— Nouvelle bannière créée</li> <li>— Image stockée dans Supabase</li> </ul>
<b>Scénario principal</b>	<ol style="list-style-type: none"> <li>1. Click sur "Ajouter bannière"</li> <li>2. Upload image (format valide)</li> <li>3. Saisie texte alternatif</li> <li>4. Ajout description optionnelle</li> <li>5. Envoi vers Supabase</li> <li>6. Confirmation création</li> </ol>
<b>Scénario d'exception</b>	<ul style="list-style-type: none"> <li>— Format image invalide → Blocage</li> <li>— Texte alternatif manquant → Avertissement</li> <li>— Échec upload → Annulation</li> </ul>

#### 4- Description textuelle du cas d'utilisation "Activer reservation" :

Le tableau ci-dessous décrit la description textuelle du cas d'utilisation "Activer reservation".

TABLE 3.5 – CU3.1 - Activation de réservation

<b>SOMMAIRE D'IDENTIFICATION CU3.1 : Activation</b>	
<b>Acteur</b>	Administrateur
<b>Résumé</b>	Transformation d'une réservation en commande.
<b>Précondition</b>	Réservation inactive sélectionnée.
<b>Post-condition</b>	<ul style="list-style-type: none"> <li>— Statut passé à "active"</li> <li>— Commande créée</li> </ul>
<b>Scénario principal</b>	<ol style="list-style-type: none"> <li>1. Sélection réservation</li> <li>2. Click bouton edit</li> <li>3. Click state et choisit "Active"</li> <li>4. Création automatique commande</li> <li>5. popup succès</li> </ol>
<b>Scénario d'exception</b>	<ul style="list-style-type: none"> <li>— Échec création commande → Rollback</li> <li>— Réservation déjà active → Erreur</li> </ul>

### 3- Description textuelle du cas d'utilisation "Supprimer reservation" :

Le tableau ci-dessous décrit la description textuelle du cas d'utilisation "Supprimer reservation".

TABLE 3.6 – CU3.2 - Suppression de réservation

<b>SOMMAIRE D'IDENTIFICATION CU3.2 : Suppression</b>	
<b>Acteur</b>	Administrateur
<b>Résumé</b>	Suppression définitive d'une réservation inactive.
<b>Précondition</b>	Réservation inactive sélectionnée.
<b>Post-condition</b>	Réservation supprimée de la base.
<b>Scénario principal</b>	<ol style="list-style-type: none"> <li>1. Sélection réservation inactive</li> <li>2. Click "Supprimer"</li> <li>3. Confirmation popup</li> <li>4. Suppression définitive</li> </ol>
<b>Scénario d'exception</b>	<ul style="list-style-type: none"> <li>— Réservation active → "erreur suppression"</li> </ul>

### 3- Description textuelle du cas d'utilisation "Creer état commandes" :

Le tableau ci-dessous décrit la description textuelle du cas d'utilisation "Creer état commandes".

TABLE 3.7 – CU4 - Creer état commandes

<b>SOMMAIRE D'IDENTIFICATION CU4 : Creer etat commandes</b>	
<b>Acteur</b>	Administrateur
<b>Résumé</b>	Création etat des commandes.
<b>Précondition</b>	Admin authentifié
<b>Post-condition</b>	<ul style="list-style-type: none"> <li>— Nouvel état créé dans le système</li> <li>— Liste des états mise à jour</li> </ul>
<b>Scénario principal</b>	<ol style="list-style-type: none"> <li>1. Accès à l'interface des états</li> <li>2. Click "+ New State"</li> <li>3. Saisie du nom de l'état</li> <li>4. Validation de la création</li> <li>5. Affichage dans la liste</li> </ol>
<b>Scénario d'exception</b>	<ul style="list-style-type: none"> <li>— Nom vide → "State name is required"</li> <li>— Nom existant → "State already exists"</li> </ul>

### 6- Description textuelle du cas d'utilisation "Creer statut commande" :

Le tableau ci-dessous décrit la description textuelle du cas d'utilisation "Creer statut commande".

TABLE 3.8 – CU5 - Creer statut commandes

<b>SOMMAIRE D'IDENTIFICATION CU5 :Creer statut commandes</b>	
<b>Acteur</b>	Administrateur
<b>Résumé</b>	Création de statuts et association aux états existants.
<b>Précondition</b>	<ul style="list-style-type: none"> <li>— Admin authentifié</li> <li>— États existants créés</li> </ul>
<b>Post-condition</b>	<ul style="list-style-type: none"> <li>— Nouveau statut créé</li> <li>— Association à un état effectuée</li> </ul>
<b>Scénario principal</b>	<ol style="list-style-type: none"> <li>1. Accès à l'interface des statuts</li> <li>2. Click "+ New Status"</li> <li>3. Saisie du nom du statut</li> <li>4. Sélection d'un état associé</li> <li>5. Validation de la création</li> </ol>
<b>Scénario d'exception</b>	<ul style="list-style-type: none"> <li>— Champ nom vide → "Status name is required"</li> <li>— État non sélectionné → "Please select a state"</li> <li>— Nom existant → "Status already exists"</li> </ul>

## 7- Description textuelle du cas d'utilisation "Modifier commande" :

Le tableau ci-dessous décrit la description textuelle du cas d'utilisation "Modifier commande".

TABLE 3.9 – CU7 - Modification des commandes

MODIFICATION DE COMMANDE	
<b>Acteur</b>	Administrateur
<b>Résumé</b>	Mettre à jour les informations d'une commande avec feedback immédiat
<b>Préconditions</b>	<ul style="list-style-type: none"> <li>— Commande existante sélectionnée</li> <li>— Connexion WebSocket active</li> <li>— Session admin valide</li> </ul>
<b>Scénario principal</b>	<ol style="list-style-type: none"> <li>1. Effectuer des modifications (statut/agent/quantités)</li> <li>2. <b>Système :</b> <ul style="list-style-type: none"> <li>— Vérifie la cohérence des quantités</li> <li>— Met à jour les stocks (sealable/réel)</li> <li>— Recalcule les montants</li> </ul> </li> <li>3. <b>Notification :</b> <ul style="list-style-type: none"> <li>— Envoi via WebSocket</li> </ul> </li> <li>4. <b>Feedback utilisateur :</b> <ul style="list-style-type: none"> <li>— Popup de confirmation (succès/erreur)</li> <li>— Durée : 5s (disparaît automatiquement)</li> </ul> </li> </ol>
<b>Post-conditions</b>	<ul style="list-style-type: none"> <li>— Données commande mises à jour en BDD</li> <li>— Stocks ajustés selon règles métier</li> <li>— Notification WebSocket générée</li> <li>— Historique des modifications enregistré</li> <li>— Interface utilisateur actualisée</li> </ul>
<b>Exceptions</b>	<ul style="list-style-type: none"> <li>— <b>Échec WebSocket</b></li> <li>— <b>Problème de stock :</b> <ul style="list-style-type: none"> <li>— Popup d'erreur spécifique :</li> <li>— "Stock insuffisant"</li> </ul> </li> </ul>

### 3.3. La conception :

Dans cette section, nous présentons les différents diagrammes de séquence détaillés pour ce sprint.

#### 3.3.1. Diagrammes de séquences détaillés :

Un diagramme de séquence détaillée permet une représentation détaillée des interactions entre les objets métiers de notre système selon un ordre chronologique. Nous présentons dans ce qui suit les diagrammes des séquences détaillés des histoires du premier sprint.

### 1. Diagramme de séquence de l'opération « Creer compte détaillant » :

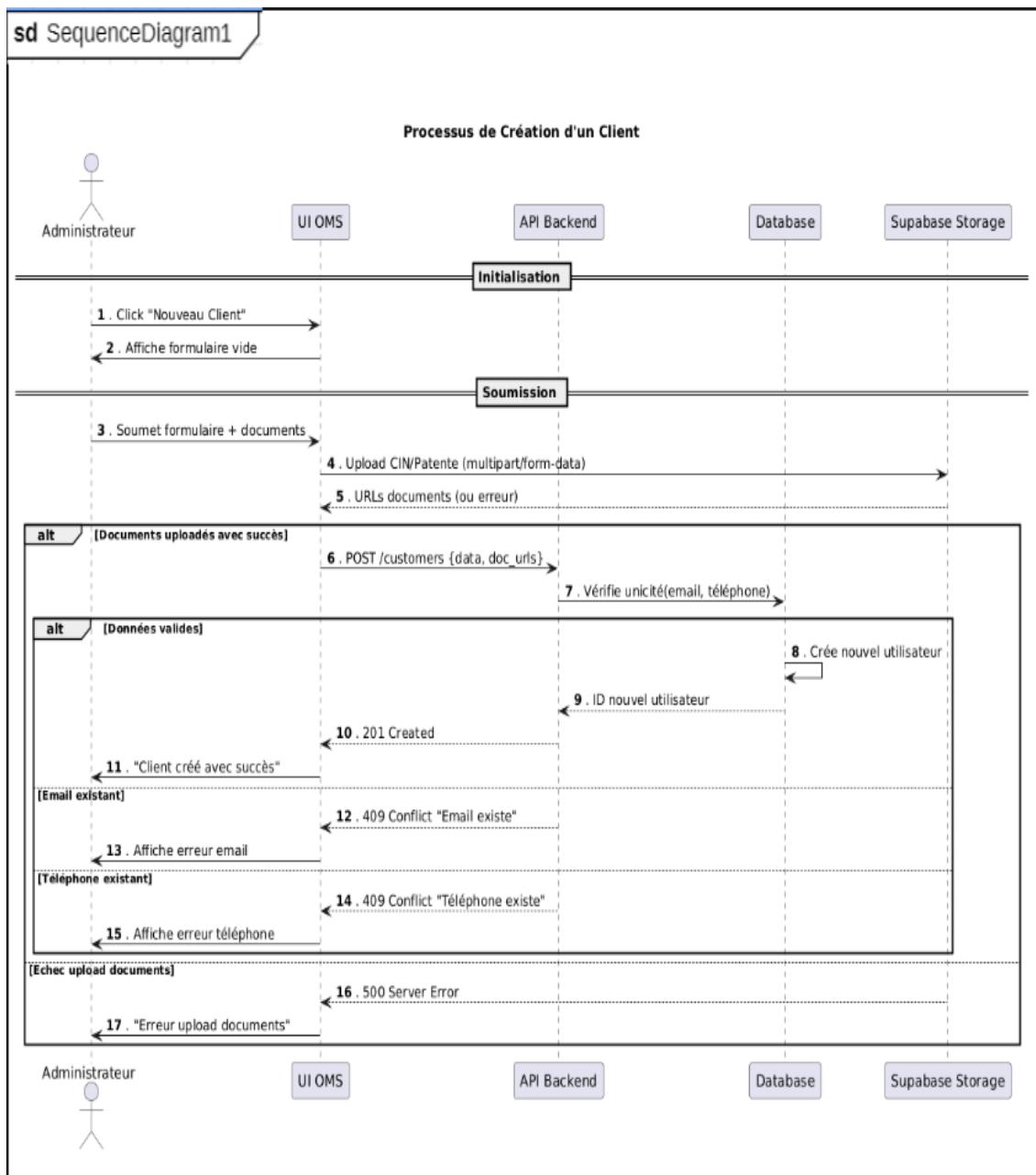


FIGURE 3.7 – Diagramme de sequence de l'opération « Creer compte détaillant »

#### Description :

Le processus débute lorsque l'administrateur clique sur « Nouveau Client » (flèche 1), ce qui affiche un formulaire vide (flèche 2). L'administrateur soumet ensuite le formulaire accompagné des documents nécessaires (flèche 3), déclenchant l'upload des fichiers (CIN/Patente) vers Supabase Storage (flèche 4). Si l'upload

réussit, les URLs des documents sont retournées (flèche 5), sinon une erreur est générée (flèche 16).

En cas de succès de l'upload, le système envoie une requête POST à l'API (flèche 6) pour créer le client, en vérifiant l'unicité de l'email et du téléphone (flèche 7). Si les données sont valides (flèche 8), un nouvel utilisateur est créé en base de données (flèche 9), et un ID unique est retourné (flèche 10), confirmant la création avec un statut 201 (flèche 11). Si l'email ou le téléphone existe déjà, le système renvoie une erreur 409 (flèches 12 et 14) et affiche un message spécifique à l'administrateur (flèches 13 et 15).

En cas d'échec de l'upload des documents (flèche 16), un message d'erreur est immédiatement notifié à l'administrateur (flèche 17). Les étiquettes « alt » matérialisent les alternatives possibles (succès/échec) à chaque étape critique, comme la validation des données ou l'upload des documents. Ce flux garantit une gestion robuste des erreurs et une expérience utilisateur claire, avec des retours immédiats en cas de problème.

## 2. Diagramme de séquence de l'opération «Modifier compte détaillant» :

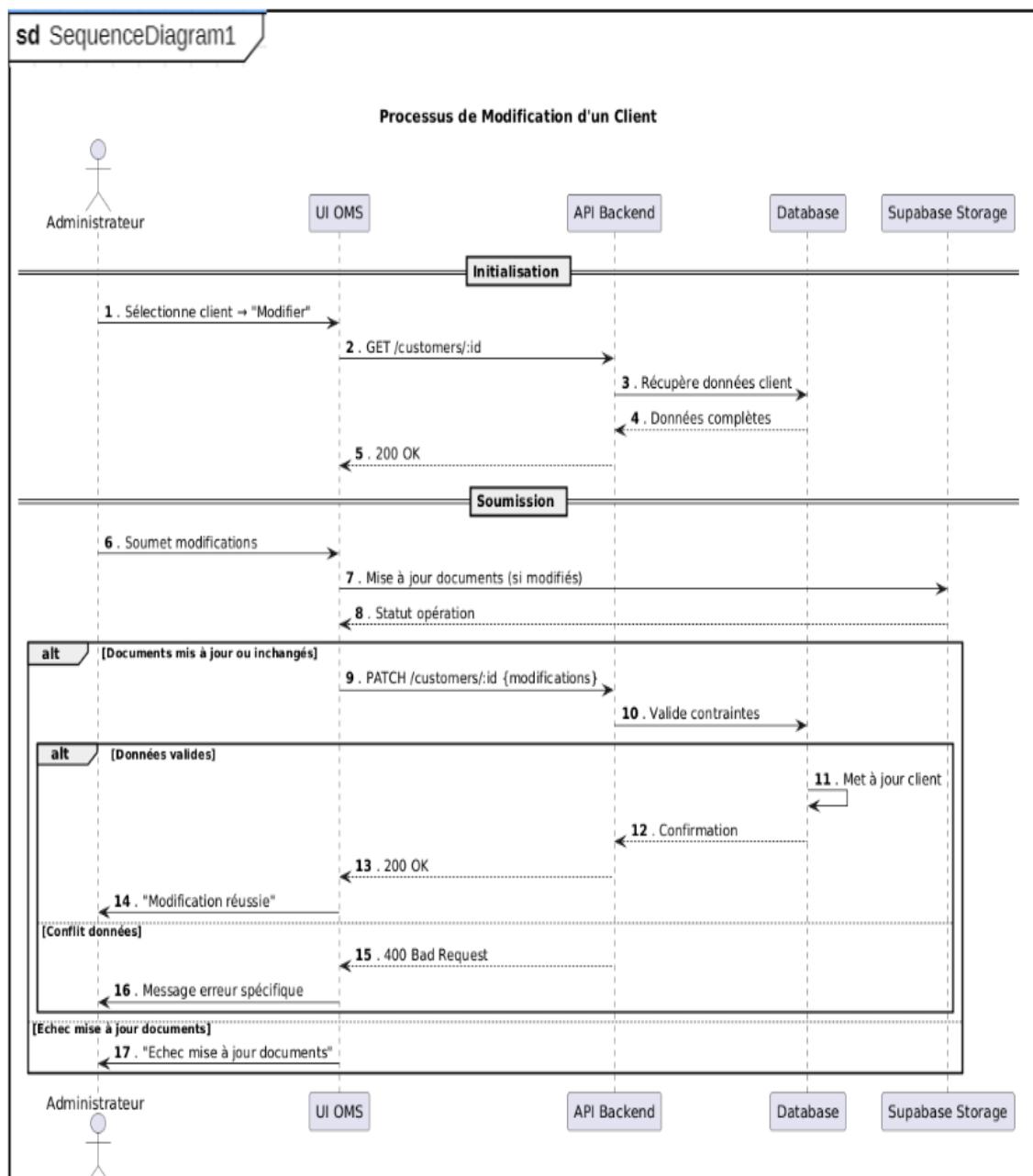


FIGURE 3.8 – Diagramme de sequence de l'opération «Modifier compte détaillant»

**Description :**

Le processus commence lorsque l'administrateur sélectionne un client et clique sur « Modifier » (flèche 1). L'interface utilisateur (UI OMS) envoie une requête GET à l'API Backend pour récupérer les données du client (flèche 2). L'API interroge la base de données (flèche 3), qui renvoie les informations complètes du client (flèche 4). L'API transmet ensuite ces données à l'interface avec un statut 200 OK (flèche 5), qui les affiche à l'administrateur.

Lorsque l'administrateur soumet les modifications (flèche 6), le système vérifie si des documents doivent être mis à jour dans Supabase Storage (flèche 7). Le statut de cette opération est retourné (flèche 8). Si les documents sont mis à jour avec succès (ou inchangés), une requête PATCH est envoyée à l'API pour valider les modifications (flèche 9). L'API vérifie les contraintes métier (flèche 10).

Si les données sont valides (flèche 11), la base de données met à jour le client (flèche 12) et confirme la modification avec un statut 200 OK (flèche 13). L'administrateur reçoit une notification de succès (flèche 14).

En cas de conflit de données (flèche 15), l'API renvoie une erreur 400 Bad Request (flèche 16), et l'interface affiche un message d'erreur spécifique.

Si la mise à jour des documents échoue (flèche 17), l'administrateur est immédiatement notifié de l'échec.

Les étiquettes « alt » illustrent les scénarios alternatifs (succès/échec), garantissant une gestion transparente des erreurs et une expérience utilisateur cohérente. Ce flux assure que les modifications sont sécurisées, validées et reflétées en temps réel dans le système.

### 3. Diagramme de séquence de l'opération « Creer banniere » :

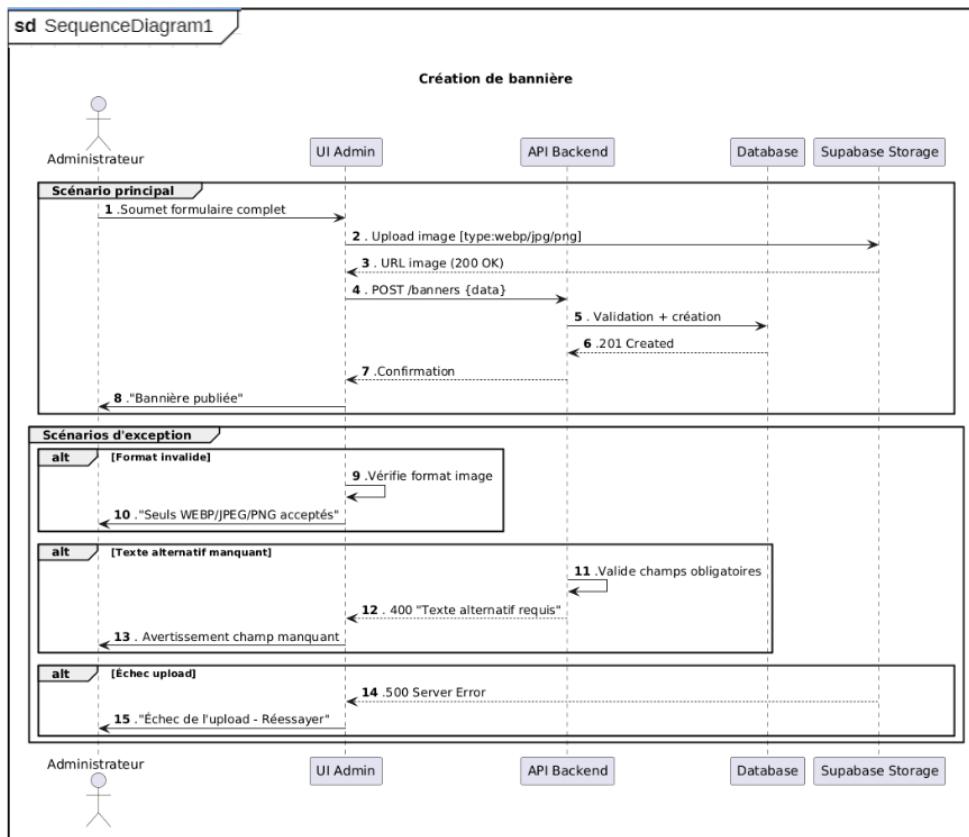


FIGURE 3.9 – Diagramme de sequence de l'opération «Creer banniere»

#### Description :

Le processus débute lorsque l'administrateur soumet un formulaire complet contenant les informations de la bannière (flèche 1). Le système initie alors l'upload de l'image vers Supabase Storage, en vérifiant que le format est valide (webp, jpg ou png) (flèche 2). Si l'upload réussit, l'URL de l'image est renvoyée avec un statut 200 OK (flèche 3).

Les données de la bannière sont ensuite envoyées à l'API Backend via une requête POST (flèche 4). L'API procède à la validation des données et à la création de la bannière dans la base de données (flèche 5). Si l'opération réussit, une confirmation 201 Created est renvoyée (flèche 6), et l'interface affiche un message de succès indiquant que la bannière a été publiée (flèches 7-8).

#### Gestion des exceptions :

Format d'image invalide : Le système vérifie le format de l'image (flèche 9) et affiche un message d'erreur spécifique si le format n'est pas supporté (flèche 10).

Texte alternatif manquant : L'API valide les champs obligatoires (flèche 11) et

retourne une erreur 400 Bad Request si le texte alternatif est absent (flèche 12), ce qui déclenche un avertissement dans l'interface (flèche 13).

Échec de l'upload : En cas d'erreur serveur (500) lors de l'upload (flèche 14), l'interface invite l'administrateur à réessayer l'opération (flèche 15).

Les étiquettes « alt » mettent en évidence les différents scénarios possibles, permettant une gestion robuste des erreurs et une expérience utilisateur optimale. Ce processus garantit que la création de bannière s'effectue de manière sécurisée, avec des validations à chaque étape et des retours d'information clairs pour l'administrateur. Les documents sont d'abord stockés dans Supabase Storage avant que les métadonnées ne soient enregistrées en base de données, assurant ainsi l'intégrité des données.

#### 4. Diagramme de séquence de l'opération « Activer reservation » :

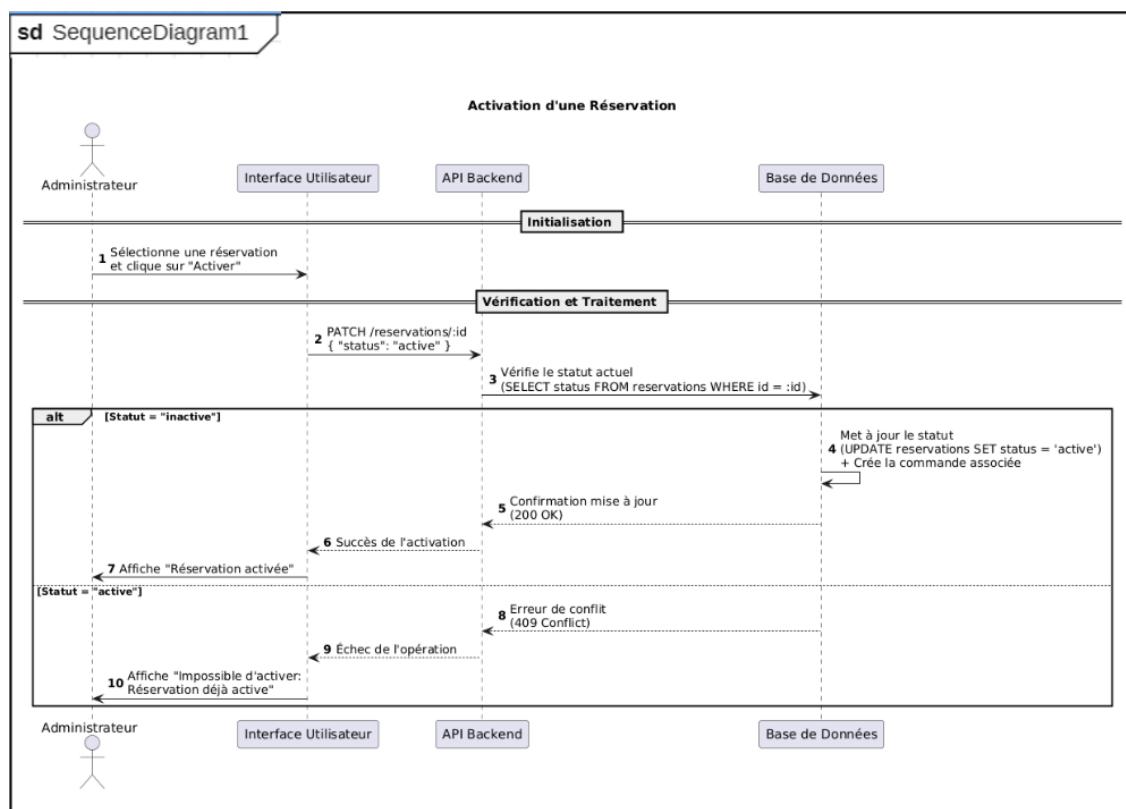


FIGURE 3.10 – Diagramme de sequence de l'opération « Activer reservation »

#### Description :

Le processus débute lorsque l'administrateur sélectionne une réservation inactive et clique sur "Activer" (flèche 1). L'interface utilisateur envoie alors une requête

PATCH à l'API Backend pour modifier le statut en "active"(flèche 2). L'API interroge la base de données pour vérifier le statut actuel de la réservation(flèche 3).

Si la réservation est bien inactive (flèche 4), le système :

-Met à jour le statut à "active" dans la base de données (flèche 5)

-Crée automatiquement une commande associée (flèche 6)

-Retourne une confirmation 200 OK (flèche 7)

-Affiche un message de succès à l'administrateur (flèche 8)

Gestion des exceptions :

Réservation déjà active : Si le statut est déjà "active"(flèche 9), le système retourne une erreur 409 Conflict(flèche 10) et affiche un message explicite(flèche 11)

Échec de la mise à jour : En cas d'erreur lors de l'opération (flèche 12), l'interface notifie l'administrateur(flèche 13)

Points clés :

-Vérification systématique du statut avant modification

-Création automatique d'une commande associée

-Messages d'erreur contextuels et informatifs

-Gestion robuste des conflits

Les étiquettes "alt" permettent de visualiser clairement les différents scénarios. Ce processus garantit une activation fiable tout en maintenant l'intégrité des données, avec des retours d'information précis à chaque étape. La réservation n'est activée que si toutes les validations sont passées avec succès, assurant ainsi la cohérence du système.

## 5. Diagramme de séquence de l'opération «Créer état commandes » :

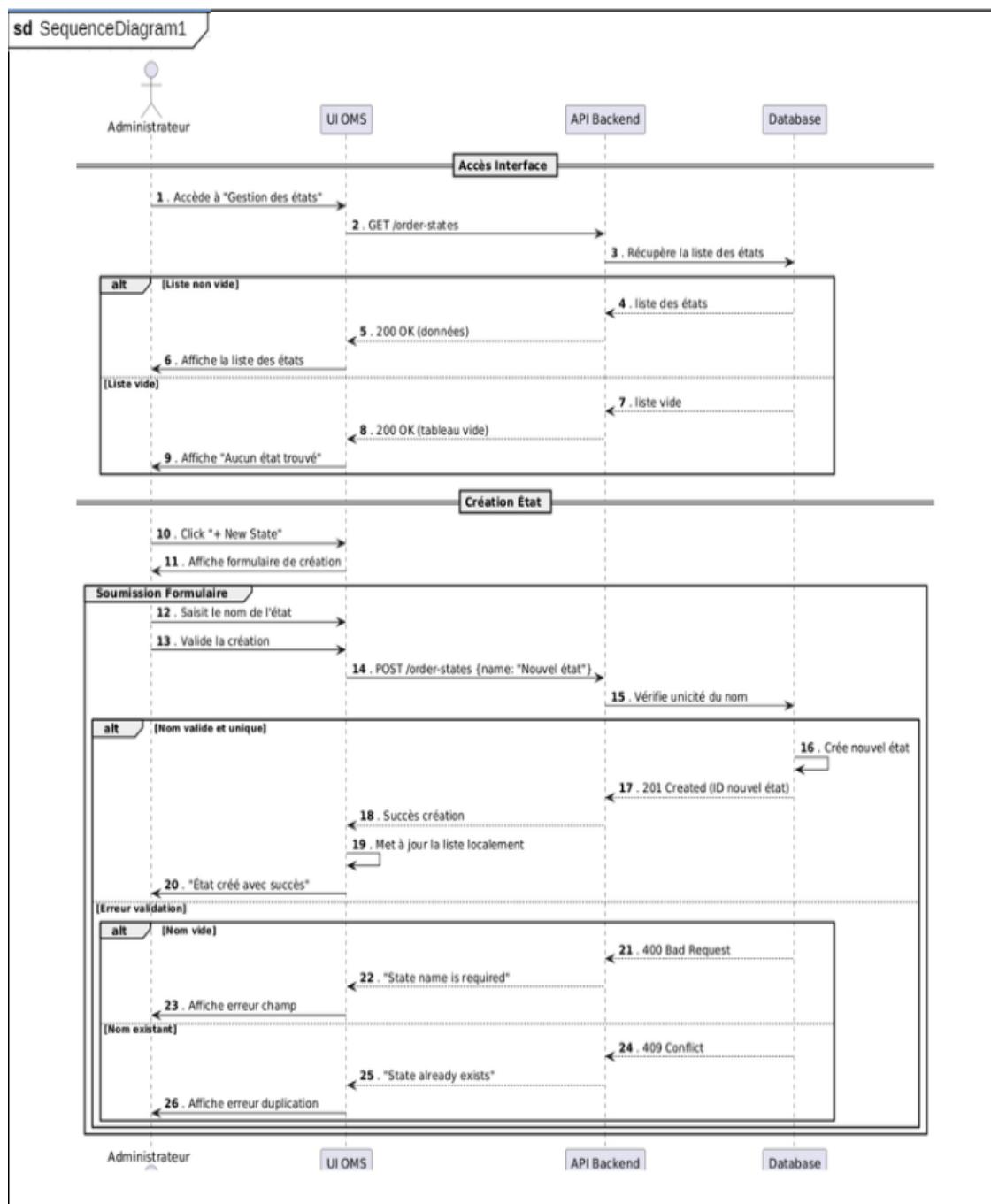


FIGURE 3.11 – Diagramme de sequence de l'opération «Créer état commande »

### Description :

Le processus débute lorsque l'administrateur accède à l'interface des états des commandes (flèche 1), déclenchant une requête GET vers l'API pour récupérer la liste des états existants (flèche 2). Le système interroge alors la base de données

(flèche 3) et deux scénarios sont possibles :

Si des états existent (alt liste non vide), ils sont retournés avec un statut 200 OK (flèche 5) et affichés dans l'interface (flèche 6). Si aucun état n'existe (alt liste vide), un tableau vide est retourné (flèche 8) et le message "Aucun état trouvé" s'affiche (flèche 9).

L'administrateur peut alors cliquer sur "+ New State" (flèche 10) pour afficher le formulaire de création (flèche 11). Après avoir saisi le nom du nouvel état (flèche 12) et validé le formulaire (flèche 13), une requête POST est envoyée à l'API (flèche 14) qui vérifie l'unicité du nom (flèche 15).

En cas de succès (nom valide et unique), le nouvel état est créé en base de données (flèche 16), un statut 201 Created est retourné (flèche 17), la liste est mise à jour localement (flèche 19) et un message de confirmation s'affiche (flèche 20).

En cas d'erreur, deux scénarios sont possibles :

Si le nom est vide : un statut 400 Bad Request est retourné (flèche 21) avec le message "State name is required" (flèche 22)

Si le nom existe déjà : un statut 409 Conflict est retourné (flèche 24) avec le message "State already exists" (flèche 25) Dans les deux cas, l'interface affiche le message d'erreur correspondant (flèche 23).

Les étiquettes "alt" matérialisent les différents chemins possibles à chaque étape critique du processus. Ce flux garantit une gestion robuste des états de commande avec des retours clairs à l'administrateur, que l'opération réussisse ou échoue.

## 6. Diagramme de séquence de l'opération «Creer statut commande » :

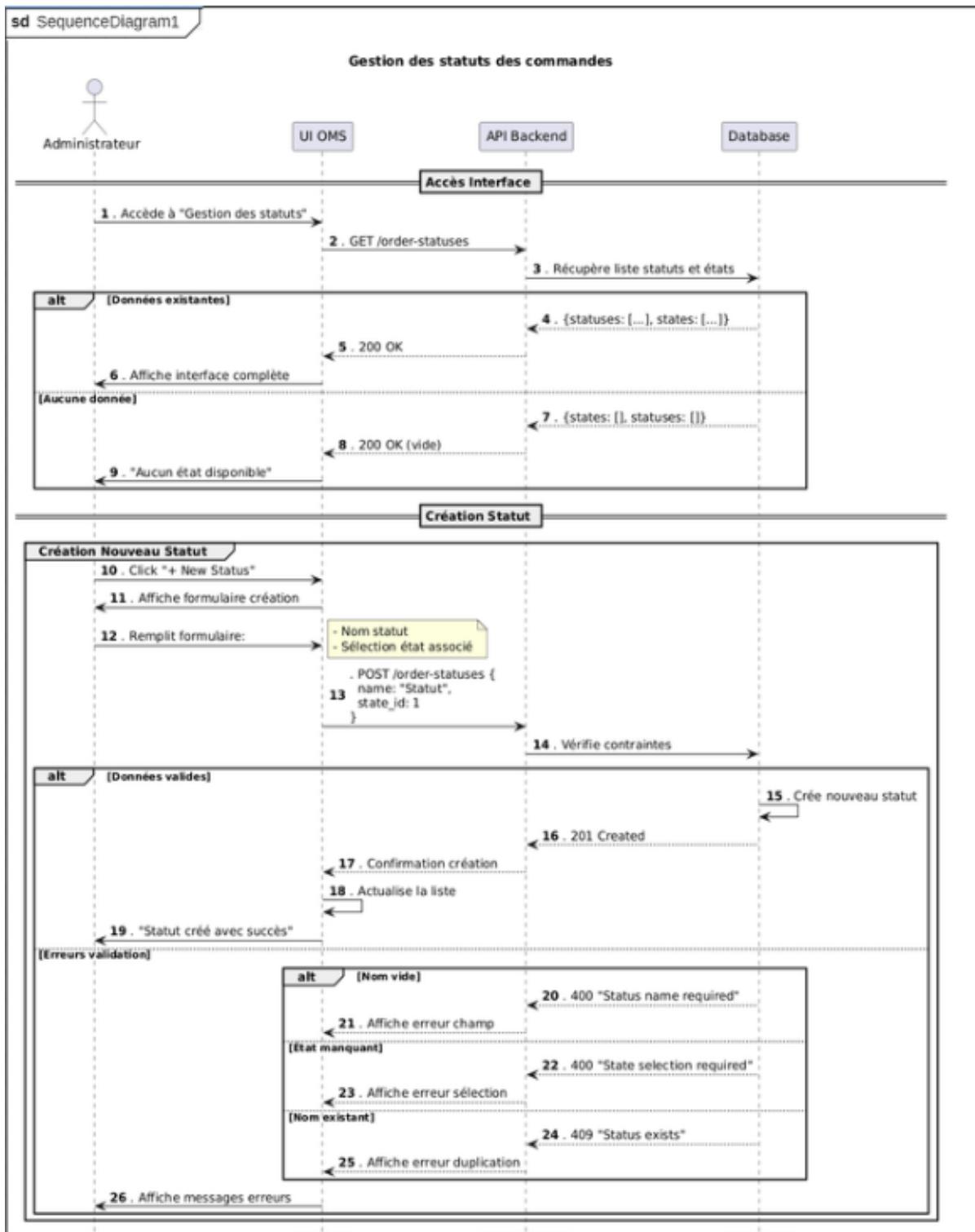


FIGURE 3.12 – Diagramme de sequence de l'opération «Creer statut commande»

**Description :**

Le processus débute lorsque l'administrateur accède à l'interface de gestion des statuts des commandes (flèche 1). L'UI OMS envoie ensuite une requête GET vers l'API Backend pour récupérer la liste des statuts (flèche 2), qui interroge la base de données (flèche 3). L'API renvoie la liste trouvée (flèche 4) ou une liste vide (flèche 7). Si des données sont disponibles, l'UI affiche l'interface complète (flèche 6), sinon un message « Aucun état disponible » apparaît (flèche 9).

Pour la création d'un nouveau statut, l'administrateur clique sur « + New Status » (flèche 10), ce qui affiche le formulaire (flèche 11). Il remplit les champs requis, comme le nom et l'état associé (flèche 12), puis l'UI envoie une requête POST avec ces informations (flèche 13). L'API vérifie les contraintes (flèche 14) et crée le statut dans la base de données si tout est valide (flèche 15). Une réponse de confirmation est renvoyée (flèche 16), l'UI confirme la création (flèche 17) et actualise la liste (flèche 18) avant d'afficher « Statut créé avec succès » (flèche 19).

En cas d'erreur, le processus suit les flèches 20 à 26 : message « Status name required » si le champ nom est vide, « State selection required » si l'état n'est pas sélectionné, ou « Status exists » si le statut est déjà présent. Ces erreurs sont ensuite affichées à l'utilisateur (flèche 26).

## 7. Diagramme de séquence de l'opération «Modifier commande» :

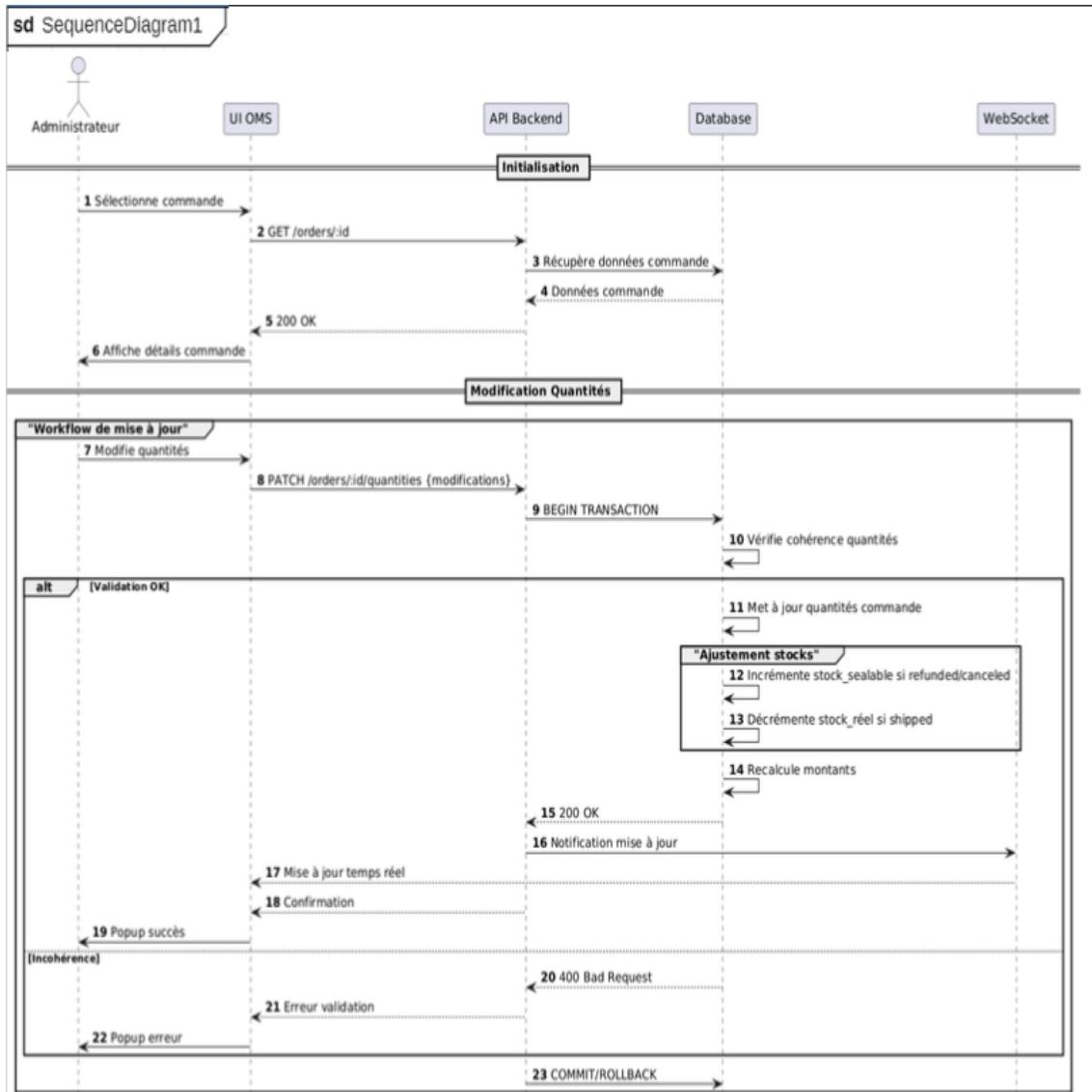


FIGURE 3.13 – Diagramme de sequence de l'opération «Modifier commande»

### Description :

Ce diagramme de séquence illustre le processus complet de mise à jour des quantités des produits d'une commande, depuis la consultation initiale jusqu'à la synchronisation en temps réel. Le flux débute lorsque l'administrateur sélectionne une commande (flèche 1), déclenchant une requête GET /order/s/{id} (flèche 2) vers l'API. Le backend récupère alors les données depuis la base de données (flèche 3) et les retourne (flèche 4) avec un statut 200 OK (flèche 5), permettant à l'interface d'afficher les détails de la commande (flèche 6).

Lorsque l'administrateur modifie les quantités (flèche 7), le système envoie une

requête PATCH /order/s/id/quantities (flèche 8) et initie une transaction (flèche 9). Le backend vérifie la cohérence des nouvelles quantités (flèche 10), conduisant à deux scénarios possibles :

En cas de validation réussie, le système :

Met à jour les quantités des produits (flèche 11)

Ajuste les stocks selon le statut : incrémente le stock "sealable" pour les articles remboursés/annulés (flèche 12) et décrémente le stock reel pour les articles expédiés (flèche 13)

Recalcule les montants (flèche 14)

Retourne un statut 200 OK (flèche 15)

Notifie les changements via WebSocket (flèche 16)

Affiche une popup de confirmation (flèche 19)

En cas d'échec de validation, le système :

Retourne un statut 400 Bad Request (flèche 20)

Affiche une popup d'erreur (flèche 22)

Le mécanisme WebSocket assure la synchronisation en temps réel (flèche 17) et la confirmation des opérations (flèche 18). Ce processus met en œuvre quatre composants clés : l'interface UI OMS pour les interactions administrateur, l'API Backend pour le traitement des requêtes, la Database pour le stockage des données, et le service WebSocket pour les mises à jour instantanées. Chaque étape est matérialisée par des flèches explicites qui tracent le flux des données et des validations, garantissant ainsi une gestion fiable et transparente des modifications de commande

### 3.3.2. Diagramme de classe :

## 3.4. Réalisation :

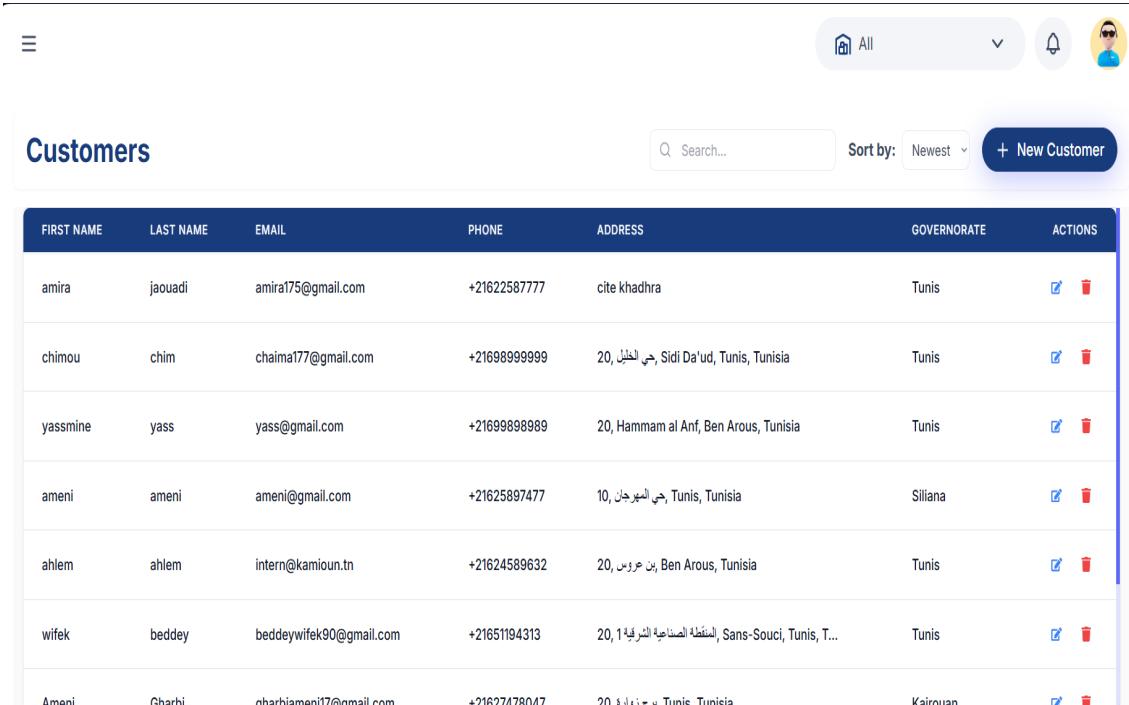
Cette partie est consacrée à l'exposition du travail achevé à travers des captures d'écrans de différentes interfaces développées pendant ce sprint.

### 3.4.1. Les interfaces de gestion comptes détaillants :

#### 1. Interface de gestion des clients :

La figure 3.14 affiche la liste des clients existants affichant leurs informations principales (nom, prénom, email, téléphone, adresse.) ainsi que des actions possibles pour chaque entrée. Les données sont présentées sous forme de tableau . Le tableau intègre un champ de recherche permettant de filtrer la liste par client en fonction du

nom, du prénom ,adresse etc.. Il offre également une fonctionnalité de tri par date de création (sort by created at), permettant d'afficher les clients du plus récent au plus ancien, ou inversement.



The screenshot shows a web-based application for managing customers. At the top, there is a header with a search bar, a dropdown menu set to 'All', a notification bell icon, and a user profile icon. Below the header, the title 'Customers' is displayed, followed by a search bar and a sorting dropdown set to 'Newest'. A blue button labeled '+ New Customer' is also present. The main content area is a table with the following columns: FIRST NAME, LAST NAME, EMAIL, PHONE, ADDRESS, GOVERNORATE, and ACTIONS. The table contains seven rows of customer data:

FIRST NAME	LAST NAME	EMAIL	PHONE	ADDRESS	GOVERNORATE	ACTIONS
amira	jaouadi	amira175@gmail.com	+21622587777	cite khadra	Tunis	 
chimou	chim	chaima177@gmail.com	+21698999999	20, حي الخليل, Sidi Da'ud, Tunis, Tunisia	Tunis	 
yassmine	yass	yass@gmail.com	+21699898989	20, Hammam al Anf, Ben Arous, Tunisia	Tunis	 
ameni	ameni	ameni@gmail.com	+21625897477	10, حي المبرجان, Tunis, Tunisia	Siliana	 
ahlem	ahlem	intern@kamioun.tn	+21624589632	20, بن عروس, Ben Arous, Tunisia	Tunis	 
wifek	beddey	beddeywifek90@gmail.com	+21651194313	20, المنقطة الصناعية الشرقية, Sans-Souci, Tunis, T...	Tunis	 
Ameni	Gharbi	qharbiameni17@gmail.com	+21627478047	20, زارزور, Tunis, Tunisia	Kairouan	 

FIGURE 3.14 – Interface de gestion des clients

## 2. Forumulaire de creation d'un nouveau détaillant (1/2) :

La figure 3.15 présente la formulaire de création d'un nouveau détaillant comprenant deux sections principales : les informations personnelles (nom, prénom, email, téléphone, mot de passe) et les informations d'adresse. Les champs marqués d'un astérisque sont obligatoires

The screenshot shows a modal dialog titled "Create New Customer". The first section, "Personal Information", contains six input fields: "First Name \*", "Last Name \*", "Email \*", "Phone \*", "Password \*", and "Confirm Password \*". The second section, "Address Information", contains two input fields: "Address \*" and "Governorate \*". At the bottom right of the modal are two buttons: "Cancel" and a blue "Create Customer" button.

FIGURE 3.15 – Formulaire de creation d'un nouveau détaillant (1/2)

### 3. Formulaire de creation d'un nouveau détaillant (2/2) :

La figure 3.16 présente la formulaire de création d'un nouveau détaillant (suite) qui présente les champs spécifiques au entreprise (nom social, identifiant fiscal, type d'entreprise, activité principale, activité secondaire, type de patente) et des documents requis (CIN, patente fiscale).

The screenshot shows a 'Create New Customer' form. At the top, it says 'Create New Customer'. The first section contains fields for 'Business Type \*' (dropdown: Superette - سوبر ماركت), 'Primary Activity \*' (dropdown: fruits secs), 'Secondary Activity' (dropdown: alimentation générale), and 'Patent Type' (dropdown: Forfaitaire). The second section is titled 'Required Documents \*' and contains fields for 'CIN (National ID) \*' (button: Upload CIN, file: cin.jpg) and 'Patente Fiscale \*' (button: Upload Patente, file: patente.jpeg). At the bottom right are 'Cancel' and 'Create Customer' buttons.

FIGURE 3.16 – Formulaire de creation d'un nouveau détaillant (2/2)

#### 4. Formulaire de modification d'un client existant :

La figure 3.17 présente la formulaire de modification des informations d'un client existant, permettant de mettre à jour les données personnelles, les données d'entreprise etc ..

The screenshot shows a modal dialog titled "Edit Customer". It is divided into two main sections: "Personal Information" and "Address Information".

- Personal Information:**
  - First Name \*: amira
  - Last Name \*: jaouadi
  - Email \*: amira175@gmail.com
  - Phone \*: +21622587777
  - New Password (optional): New Password
- Address Information:**
  - Address \*: cite khadra
  - Governorate \*: Tunis

At the bottom right of the dialog are two buttons: "Cancel" and "Update Customer".

FIGURE 3.17 – Formulaire de modification d'un client existant

## 5. Suppression d'un client existant :

La figure 3.18 présente la boîte de dialogue de confirmation pour la suppression d'un client. Cette interface demande une validation explicite de l'utilisateur avant la suppression définitive, avec les options **Annuler** ou **Supprimer**.

Suite à cette action, une notification contextuelle s'affiche pour indiquer le statut de l'opération :

- **Succès** : Confirmation de la suppression du client.
- **Échec** : Message d'erreur en cas de problème.

L'administrateur conserve la possibilité d'annuler l'opération avant validation finale via le bouton **Annuler**.

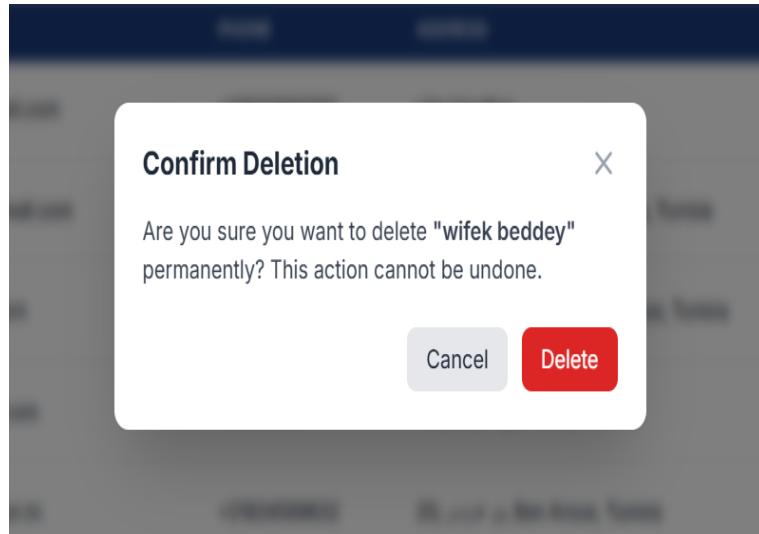


FIGURE 3.18 – la boîte de dialogue de confirmation pour la suppression d'un client

## 6. Interface de gestion du stockage dans Supabase :

La figure 3.19 présente l'interface de gestion du stockage dans Supabase, montrant différentes sections pour les buckets, fichiers et documents clients, y compris les options de configuration et de gestion des fichiers.

FIGURE 3.19 – Interface de gestion du stockage dans Supabase

### 3.4.2. Les interfaces de gestion des bannières :

#### 1. Interface de gestion des bannières :

La figure 3.20 présente l'interface de gestion des bannières. L'écran affiche la liste des bannières existantes avec leurs informations principales (image, texte alternatif, description, dates de création et mise à jour) ainsi que des actions disponibles pour chaque élément. Les données sont organisées sous forme de tableau interactif.

IMAGE	ALT TEXT	DESCRIPTION	CREATED AT	UPDATED AT	ACTIONS
	BOOM BOOM	Our new energy drink is crafted to fuel yo...	25 juin 2025, 17:27	25 juin 2025, 17:29	
	Agritable fruits and vegetables	Discover Fresh, Sustainable Delights fro...	25 juin 2025, 17:27	25 juin 2025, 17:29	
	Taralli	Taste the Difference – Where Quality Mee...	9 mai 2025, 15:24	25 juin 2025, 17:30	

FIGURE 3.20 – Interface de gestion des bannières

#### 2. Formulaire de création d'une nouvelle bannière :

La figure 3.21 présente la formulaire de création des bannières. Les images téléchargées sont automatiquement enregistrées dans le service **Supabase Storage**, avec un bucket dédié pour garantir une gestion sécurisée et évolutive des médias.

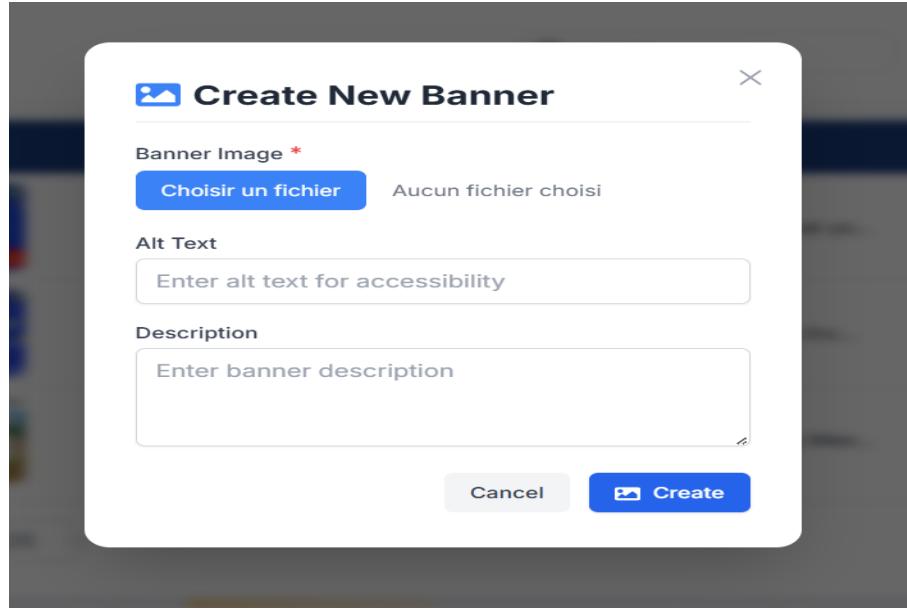


FIGURE 3.21 – Formulaire de creation d'une nouvelle banniere

### 3. Stockage des bannieres dans supabase storage :

La figure 3.22 montre le stockage des bannières dans Supabase, avec un bucket dédié **banners**. Chaque bannière est enregistrée avec :

- Son nom de fichier unique (ex : `banner-1754735836214-imgBanner1.PNG`)
- Ses métadonnées (type, taille, dates)
- Des options de gestion (téléchargement, URL, suppression)

Ce système centralisé permet un accès rapide et sécurisé aux bannières depuis l'application.

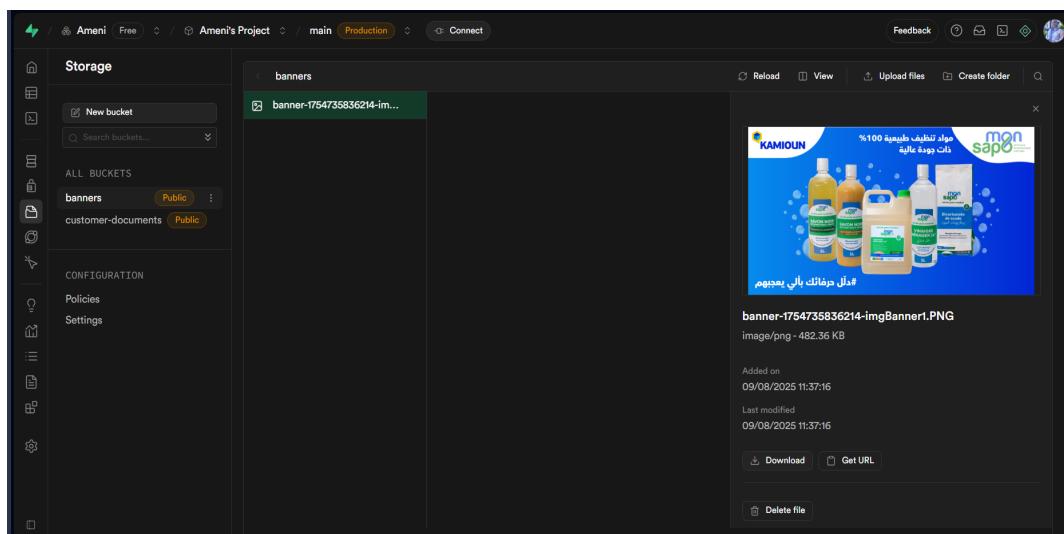


FIGURE 3.22 – Stockage des bannieres dans supabase storage

### 3.4.3. Interfaces de gestion des réservations :

#### 1. Interface de gestion des réservations :

La figure 3.23 présente le tableau de gestion des réservations, listant les commandes avec leurs détails principaux

The screenshot shows a web application interface for managing reservations. At the top right, there are buttons for 'All' (with a dropdown arrow), a bell icon, and a user profile icon. Below the header, the title 'Reservations' is displayed. A search bar with the placeholder 'Search Reservations by Id...' is present. The main area features a table with the following columns:

	AMOUNT TTC	AMOUNT ORDERED	SHIPPING METHOD	SHIPPING AMOUNT	STATE	FROM MOBILE	WEIGHT	CUSTOMER	PAYMENT METHOD	CREATED AT	UPDATED AT	RESERVATION ITEMS	COMMENT	ACTIONS
i73636262c09ff	24 DT	25.24 DT	delivery	0 DT	Active	No	10.2	⋮ Ameni Gharbi	cash	16/07/2025	16/07/2025	<a href="#">View Items (1)</a>	no comment	<span>edit</span> <span>trash</span> <span>down</span>
168d5d1962374	103.7 DT	114.14 DT	delivery	8.4 DT	Active	No	54.4	⋮ Ameni Gharbi	cash	16/07/2025	16/07/2025	<a href="#">View Items (3)</a>	no comment	<span>edit</span> <span>trash</span> <span>down</span>
i705454fc83397	16.8 DT	17.8 DT	delivery	0 DT	Active	No	120	⋮ Ameni Gharbi	cash	15/07/2025	15/07/2025	<a href="#">View Items (1)</a>	no comment	<span>edit</span> <span>trash</span> <span>down</span>
a76f743dd6c19	21.6 DT	23.82 DT	delivery	1DT	Active	No	7.2	⋮ Ameni Gharbi	cash	15/07/2025	15/07/2025	<a href="#">View Items (1)</a>	no comment	<span>edit</span> <span>trash</span> <span>down</span>
i7a12d2b438769	22.5 DT	24.73 DT	delivery	1DT	Active	No	7.2	⋮ Ameni Gharbi	cash	14/07/2025	14/07/2025	<a href="#">View Items (1)</a>	no comment	<span>edit</span> <span>trash</span> <span>down</span>
4cab59f22c10	20.8 DT	21.8 DT	delivery	0 DT	Inactive	No	120	⋮ Ameni Gharbi	cash	14/07/2025	14/07/2025	<a href="#">View Items (1)</a>	no comment	<span>edit</span> <span>trash</span> <span>down</span>
i341b7255f7d87	400 DT	400 DT	pickup	0 DT	Inactive	No	4	⋮ Yassin Merhineh	cash	12/07/2025	12/07/2025	<a href="#">View Items (2)</a>	no	<span>edit</span> <span>trash</span> <span>down</span>

At the bottom left, there is a 'Items per page:' dropdown set to 25. On the bottom right, there are navigation icons for page numbers 1 through 4.

FIGURE 3.23 – Interface de gestion des réservations

#### 2. Interface détails des articles réservés :

La figure 3.24 présente le détail des articles réservés, organisés par partenaires commerciaux. L’interface montre que chaque partenaire peut disposer de plusieurs sources d’approvisionnement distinctes, avec pour chaque item :

- Les informations du partenaire et sa source
- La date de livraison prévue
- Les caractéristiques produits (nom, référence SKU, prix)
- Les quantités et poids totaux

Cette vue granulaire permet non seulement le suivi des commandes mais aussi une traçabilité complète de la chaîne d’approvisionnement, avec la possibilité d’identifier précisément l’origine de chaque produit.

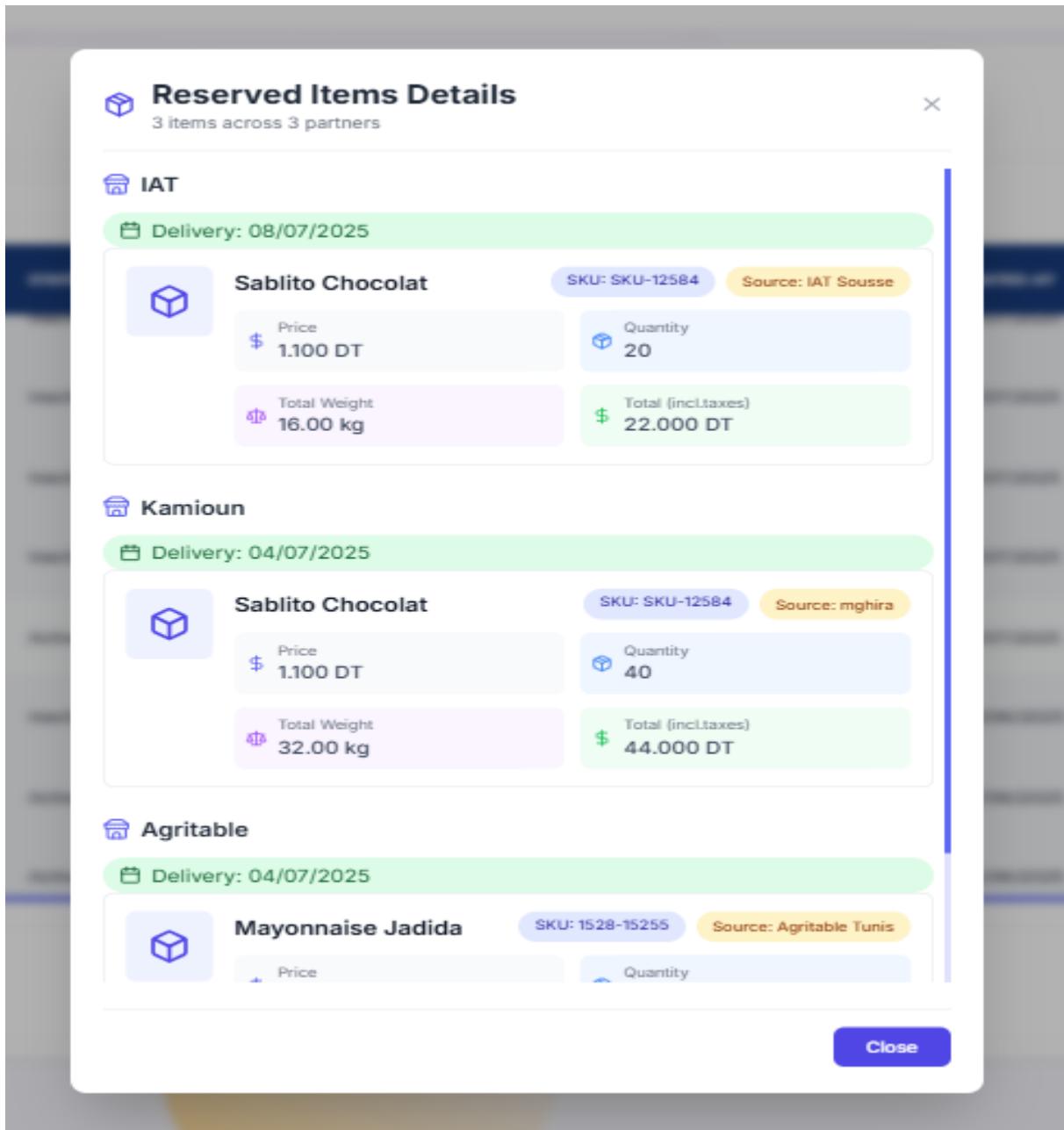


FIGURE 3.24 – Interface détails des articles réservés

### 3. Interface de modification reservation :

La figure 3.25 présente l'interface de modification d'une réservation. Cette vue permet exclusivement de changer le statut de la réservation de **Inactive** à **Active** pour générer une commande associée, sans possibilité de modifier les autres champs (montants, méthode de livraison, informations client). Les données existantes (poids, montants, etc.) restent verrouillées pour garantir l'intégrité de la transaction.

The screenshot shows a modal window titled "Edit Reservation" with the ID "#6863bbcb6c8909dc4ba959ac". The window is divided into three main sections: "Financial Information", "Status Information", and "Additional Information".

- Financial Information:** Contains fields for "Amount TTC (DT)" (13.2), "Amount Ordered (DT)" (17.6), and "Shipping Amount (DT)" (4.4).
- Status Information:** Contains fields for "State" (set to "Inactive"), "From Mobile" (set to "No"), "Shipping Method" (set to "delivery"), "Loyalty Points Value" (0), and "Weight" (8.8).
- Additional Information:** Contains fields for "Customer" (Ameni Gharbi) and "Created At" (01/07/2025).

At the bottom right are "Cancel" and "Save Changes" buttons.

FIGURE 3.25 – Interface de modification reservation

#### 3.4.4. Interfaces de gestion des commandes :

##### 1. Interface de gestion des commandes :

La figure 3.26 présente l'interface de gestion des commandes de détaillant.

### Order Management

The screenshot displays a table titled "Order Management" with the following data:

ID	AMOUNT TTC	AMOUNT REFUNDED	AMOUNT CANCELED	AMOUNT ORDERED	AMOUNT SHIPPED	SHIPPING METHOD	SHIPPING AMOUNT	IS ACTIVE	FROM MOBILE	WEIGHT	CREATED AT	UPDATED AT	STATUS	STATE	CUSTOMER
3l	52	0	0	52	52	Delivery	0.00	Yes	No	300	16/07/2025 12:22	16/07/2025 12:23	open	new	America
3b	36	0	0	36	36	Delivery	12.00	Yes	No	24	16/07/2025 12:16	16/07/2025 12:19	closed	closed	America
i03	24	0	0	24	24	Delivery	0.00	Yes	No	19.2	16/07/2025 12:06	16/07/2025 12:06	open	new	America
7	103.7	0	0	114.14	52.5	Delivery	8.40	Yes	No	54.4	16/07/2025 11:33	16/07/2025 12:21	open	new	America
3e	16.8	8.4	0	16.8	8.4	Delivery	0.00	Yes	No	120	15/07/2025 15:34	16/07/2025 11:52	closed	closed	America
lb0	21.6	9.6	9.6	21.6	2.4	Delivery	1.00	Yes	No	7.2	15/07/2025 15:15	15/07/2025 15:31	open	new	America
8a	22.5	0	0	24.73	0	Delivery	1.00	Yes	No	7.2	14/07/2025 14:27	14/07/2025 14:27	open	new	America

Items per page: 25 1 2 >

FIGURE 3.26 – Interface de gestion des commandes

## 2. Interface de modification des articles commandés(1/2) :

La figure 3.27 présente l'interface de modification des articles commandés. Cette interface impose une contrainte de validation importante : lors de la modification de la quantité livrée (`shipped`), la somme des quantités `refunded` (retournée) et `canceled` (annulée) doit obligatoirement être égale à la différence entre la quantité initialement commandée et la quantité livrée.

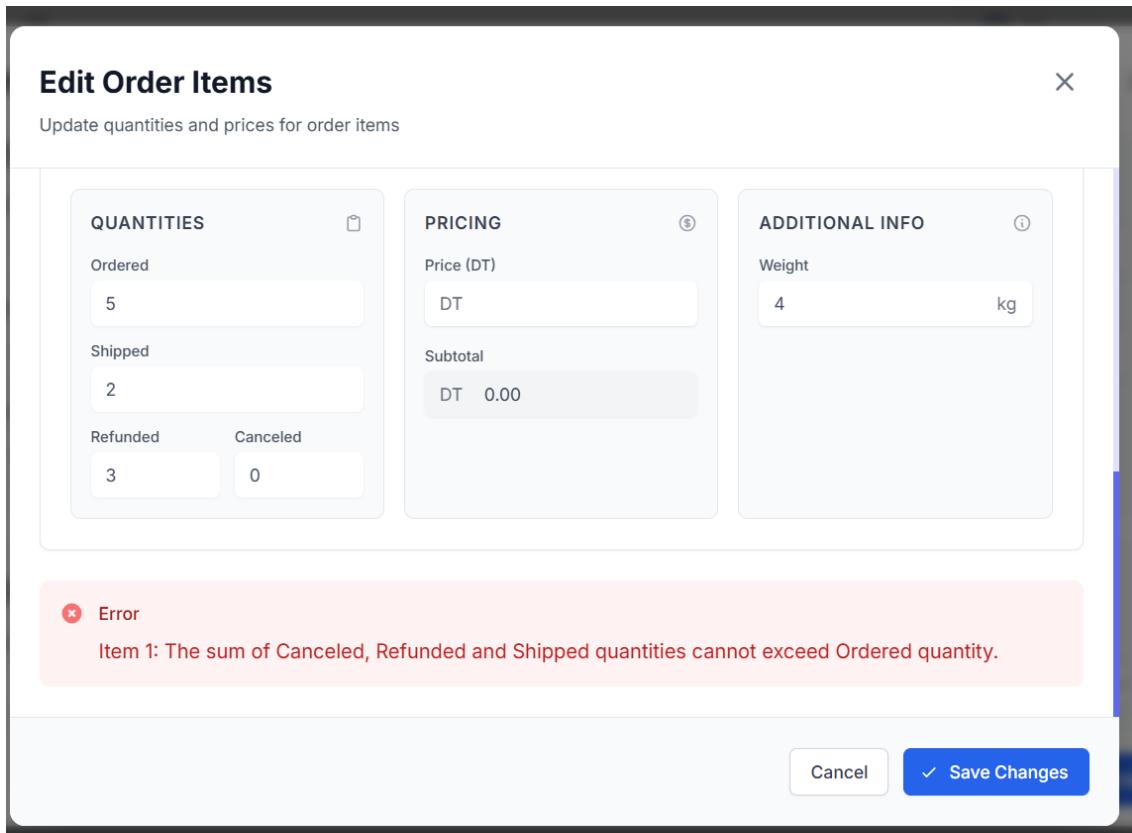


FIGURE 3.27 – Interface de modification des articles commandés

### 3. Interface de modification des articles commandés(2/2) :

La figure 3.28 présente l'interface de modification des montants des articles commandés. Après validation réussie des modifications (quantités livrées, retournées ou annulées), l'interface affiche en temps réel les montants recalculés automatiquement

Ce recalculation instantané permet une visualisation immédiate de l'impact financier des modifications tout en garantissant l'exactitude des données comptables. Les valeurs sont mises à jour selon les règles métiers prédéfinies sans nécessiter d'actualisation manuelle.

The screenshot shows a modal window titled "Edit Order Details". It contains three main sections: "Amounts", "Order Details", and "Associations".

- Amounts:** A grid of six input fields with placeholder icons and values:
  - Amount TTC: 96,3
  - Amount Refunded: 19,8
  - Amount Ordered: 96,3
  - Amount Shipped: 76,5
  - Amount Canceled: 0
  - Shipping Amount: 38,5
- Order Details:** A section with two dropdown menus:
  - State: closed
  - Status: (empty)
- Associations:** A section with two dropdown menus:
  - Customer: Ameni Gharbi
  - Agent: (empty)

At the bottom right are "Cancel" and "Update Order" buttons.

FIGURE 3.28 – Interface de modification des articles commandés avec recalcule auto des montants après modification des quantités

#### 4. Interface de modification de commande :

La figure 3.29 présente l’interface de gestion des commandes offrant deux fonctionnalités clés :

- Assignation d’agents : Possibilité d’affecter un agent de livraison à une commande immédiatement après sa création
- Gestion des états :
  - Modification du `state` (état global) et `status` (statut détaillé)
  - Transition contrôlée lors des événements clés (livraison, annulation)
  - Couplage strict entre chaque état et ses statuts autorisés

Cette double logique permet :

- Une traçabilité complète du cycle de vie des commandes
- Une coordination optimale entre les équipes commerciales et logistiques
- Le respect des workflows métiers prédéfinis

The screenshot shows a modal window titled 'Edit Order Details'. It is divided into two main sections: 'Order Details' on the left and 'Associations' on the right.

**Order Details:**

- State: complete
- Status: delivered
- Shipping Method: delivery

**Associations:**

- Customer: Ameni Gharbi
- Agent: Yassine Medhioub

FIGURE 3.29 – Interface de modification commande

## 5. Notification de modification commande :

La figure 3.30 montre les notifications de mise à jour des commandes, incluant systématiquement :

- L'administrateur ayant effectué la modification
- Le numéro de commande concerné
- La date et heure précise de l'opération

Ces alertes permettent un suivi en temps réel des modifications apportées aux commandes.

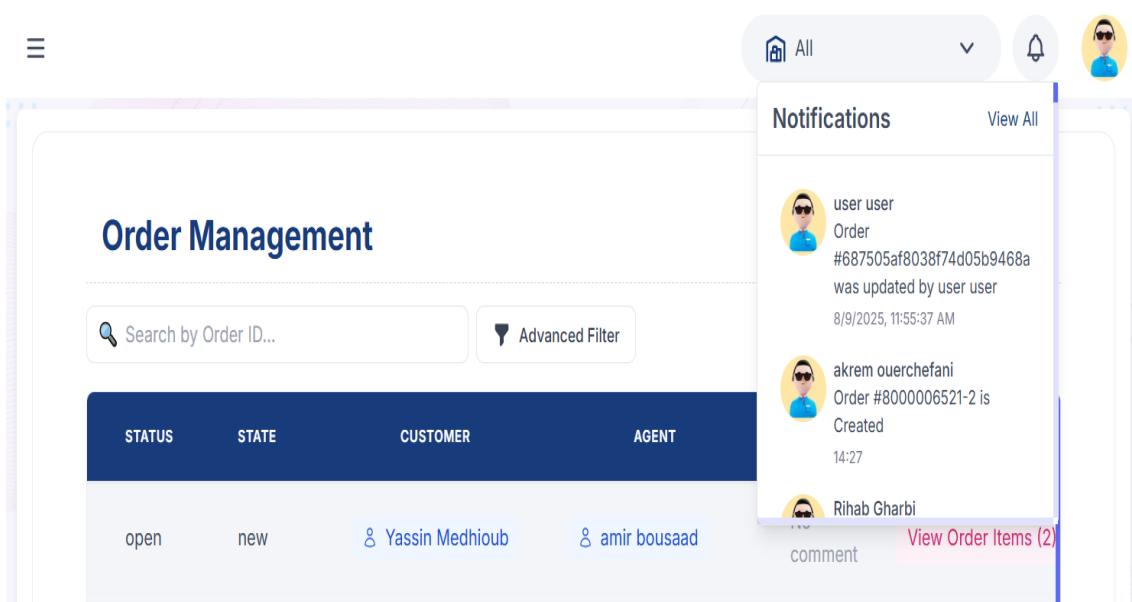


FIGURE 3.30 – Notification de modification commande

### 3.4.5. Interfaces de gestion des états des commandes :

#### 1. Interface de gestion des états des commandes :

La figure 3.31 présente l'interface de gestion des états des commandes, permettant de :

- Rechercher et trier les états
- Modifier les états existants
- Créer de nouveaux états

The screenshot shows the KAMIOUN application's navigation menu on the left, with 'Order' selected. The main area is titled 'State' and contains a table of command states:

ID	NAME	ACTIONS
682c5267e7838328c7d61c5b	canceled	
682b01f94a83b7597968ac28	closed	
67f3dd0f9308b66367b1c9cc	complete	
67d954d4e92933dc381ee237	new	

FIGURE 3.31 – Interface de gestion des états des commandes

## 2. Formulaire de creation d'un nouveau etat

La figure 3.32 présente le formulaire de création d'un nouvel état pour les commandes.

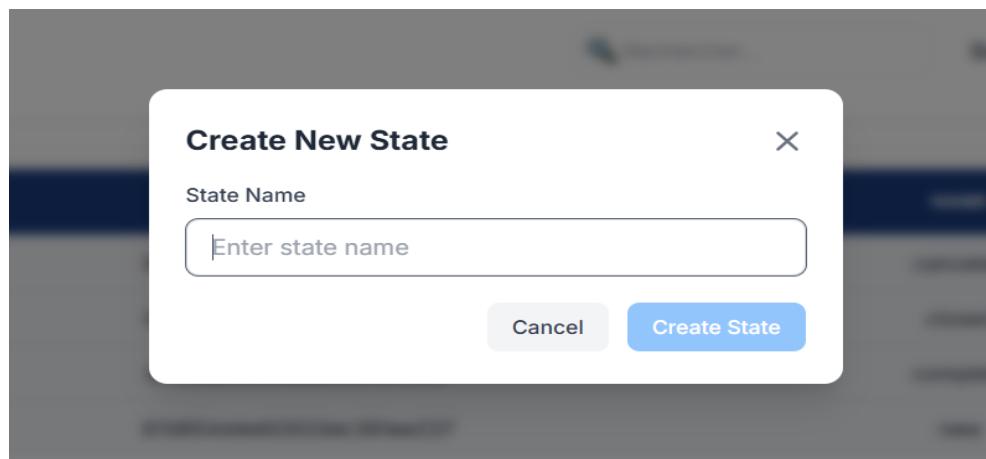


FIGURE 3.32 – Formulaire de creation d'un nouvel etat

## 3. Cas d'erreur lors de creation d'un nouvel etat

La figure 3.33 illustre un cas d'erreur lors de la création d'un état. Le système détecte et empêche la duplication d'états existants (comme "canceled", "closed", etc.) en affichant la liste complète des états déjà configurés. Cette validation garantit l'unicité des états dans le workflow.

ID	NAME	ACTIONS
682c526767838328c7d61c5b	canceled	
682b01f94a83b7597968ac28	closed	
67f3dd0f9308b66367b1c9cc	complete	
67d954d4e92933dc381ee237	new	

FIGURE 3.33 – Cas d'erreur lors de creation d'un nouvel etat

### 3.4.6. Interfaces de gestion des statuts des commandes :

#### 1. Interface de gestion des statuts des commandes :

La figure 3.34 présente l'interface de gestion des statuts des commandes, permettant de :

- Rechercher et trier les statuts des commandes
- Modifier les statuts existants
- Créeer de nouveaux statuts

ID	NAME	STATE NAME	ACTIONS
687829ad62a3d490d20da14f	delivered	closed	<input checked="" type="checkbox"/>
683d837c2cdcfdd843dd5d1	closed	closed	<input checked="" type="checkbox"/>
683d834a2cdcfdd843dd5d9b	unpaid	complete	<input checked="" type="checkbox"/>
683d826d2cdcfdd843dd5d99	valid	new	<input checked="" type="checkbox"/>
683d824e2cdcfdd843dd5d96	canceled	canceled	<input checked="" type="checkbox"/>
67f3e0679308b66367b1c9d9	open	new	<input checked="" type="checkbox"/>
67f3e0569308b66367b1c9d6	shipped	new	<input checked="" type="checkbox"/>

FIGURE 3.34 – Interface de gestion des statuts des commandes

#### 2. Formulaire de création d'un statut

La figure 3.35 présente le formulaire de création d'un nouveau statut :

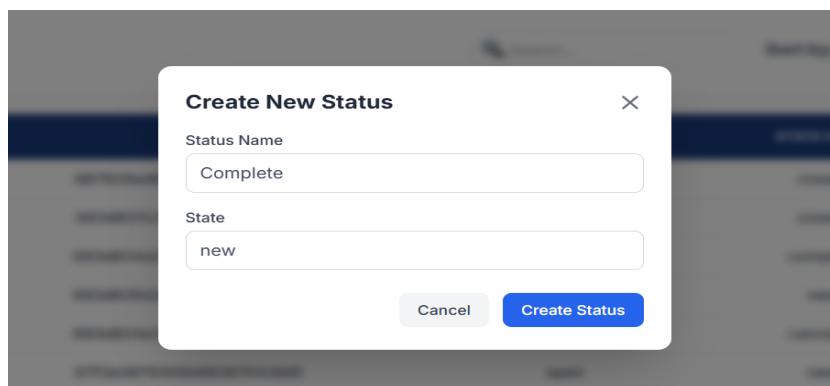


FIGURE 3.35 – Formulaire de création d'un statut

### 3.5. Test :

Le test d'un produit logiciel est un processus essentiel qui vise à vérifier le bon fonctionnement du système en comparant les comportements attendus aux résultats obtenus.

Avant la fin de chaque sprint, nous avons testé les fonctionnalités du module, puis les avons validées avec le Product Owner.

Pour ce faire, nous avons élaboré un ensemble de cas de test couvrant les scénarios du Sprint 1, présentés dans le tableau ci-dessous.

TABLE 3.10 – Résultats des tests - Sprint [1]

Fonctionnalité	Procédure de test	Résultat attendu	Statut
<b>Gerer des comptes</b>			
CU1.1 - Créer compte	<ul style="list-style-type: none"> <li>— Remplissage formulaire complet</li> <li>— Upload documents</li> <li>— Validation</li> </ul>	<ul style="list-style-type: none"> <li>— Compte créé (BDD + Supabase)</li> <li>— popup succès</li> </ul>	OK
CU1.2 - Modifier compte	<ul style="list-style-type: none"> <li>— Sélection compte</li> <li>— Mise à jour champs</li> <li>— Changement documents</li> </ul>	<ul style="list-style-type: none"> <li>— Données mises à jour</li> </ul>	OK
<b>Gerer des bannières</b>			
CU2.1 - Créeer bannière	<ul style="list-style-type: none"> <li>— Upload image (WebP/JPG/PNG)</li> <li>— Saisie metadata</li> </ul>	<ul style="list-style-type: none"> <li>— Bannière publiée</li> <li>— URL accessible</li> </ul>	OK
<b>Gerer des commandes</b>			
CU7 - Modifier commande	<ul style="list-style-type: none"> <li>— Changement statut</li> <li>— Modification quantités</li> <li>— Réassigntion agent de livraison</li> </ul>	<ul style="list-style-type: none"> <li>— Mise à jour en temps réel</li> <li>— Notification Web-Socket</li> <li>— Ajustement stocks</li> </ul>	OK
CU3.1 - Activer réservation	<ul style="list-style-type: none"> <li>— Sélection réservation</li> <li>— Basculer → "Active"</li> </ul>	<ul style="list-style-type: none"> <li>— Commande créée</li> <li>— Statut mis à jour</li> </ul>	OK
<b>Gerer états et statuts des commandes</b>			
CU4 - Gérer États commandes	<ul style="list-style-type: none"> <li>— Créeation nouvel état</li> <li>— Association statuts</li> </ul>	<ul style="list-style-type: none"> <li>— Disponible dans workflow</li> <li>— Cohérence maintenue</li> </ul>	OK
CU5 - Gérer Statuts commandes	<ul style="list-style-type: none"> <li>— Créeation statut</li> <li>— Lien à un état</li> </ul>	<ul style="list-style-type: none"> <li>— Transitions possibles</li> </ul>	OK