

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



## DEPARTEMENT INFORMATIQUE

## RAPPORT DE PROJET DE FIN D'ÉTUDES

*En vue de l'obtention du :*  
*Diplôme National d'Ingénieur Informatique*  
*Option : Génie Logiciel-Architecture Logicielle*

---

# Conception et Développement d'une application mobile d'expédition et de suivie des colis

---

**Elaboré par :** Rihem FARJALLAH

Soutenu le 11/07/2023 devant le jury :

<i>Président :</i>	supervisor name	ISSAT Sousse
<i>Examinateur :</i>	supervisor name	ISSAT Sousse
<i>Encadrant :</i>	Dr. Nessrine ELLOUMI	ISSAT Sousse
<i>Encadrant Industriel :</i>	Ing. Ibtissem BOUCHNAK	SAHTICARE

Année Universitaire : 2022-2023

Code Sujet : FI-GL23-053

# Dédicace

# Remerciements

# Résumé

Ce travail a été élaboré dans le cadre du projet de fin d'études pour l'obtention du diplôme d'ingénieur en informatique, qui a été réalisé au sein de la société SAHTICARE.

Ce projet consiste à concevoir et développer une application qui comprend deux parties : une partie web et une partie mobile. Elle assure la connexion entre le voyageur qui propose son service et le client qui souhaite en bénéficier. En fait, l'utilisateur de notre application peut être les deux à la fois. Le voyageur propose son offre et les clients y postulent, puis le voyageur choisit les clients qui lui conviennent et peut même les contacter via une discussion instantanée.

**Mots clés :** Application Mobile, Flutter, Model-View-Controller, Dart.

---

## Abstract

This work was developed as part of a final project to obtain a degree in computer engineering, which was carried out within the company SAHTICARE.

The project aims to design and develop an application that consists of two parts : a web part and a mobile part. It facilitates the connection between the traveler who offers their service and the client who wants to benefit from it. In fact, the user of our application can be both the traveler and the client. The traveler presents their offer and clients can apply for it, after which the traveler selects suitable clients and can even contact them through an instant messaging feature.

**keywords :** Mobile application, Flutter, Model-View-Controller, Dart.

---

# Table des matières

<b>Table des figures</b>	vi
<b>Liste des tableaux</b>	viii
<b>Introduction générale</b>	1
<b>1 Présentation générale du projet</b>	3
Introduction . . . . .	3
1.1 Présentation de l'organisme d'accueil . . . . .	3
1.1.1 Les fondateurs de SAHTICARE . . . . .	3
1.1.2 Fiche technique . . . . .	4
1.1.3 Cadre général du projet . . . . .	4
1.1.4 Problématique . . . . .	4
1.2 Etude de l'existant . . . . .	4
1.2.1 Analyse de l'existant . . . . .	4
1.2.2 Critique de l'existant . . . . .	7
1.2.3 Solution envisagée . . . . .	8
1.3 Processus de développement . . . . .	8
1.3.1 Développement incrémental . . . . .	8
1.3.2 Incréments du logiciel . . . . .	8
1.3.3 planification anticipée des tâches . . . . .	9
Conclusion . . . . .	9
<b>2 Analyse et spécification des besoins</b>	10
Introduction . . . . .	10
2.1 Identification des acteurs . . . . .	10
2.1.1 Partie Web . . . . .	10
2.1.2 Partie Mobile . . . . .	10
2.2 Spécification des besoins . . . . .	11
2.2.1 Les besoins fonctionnels . . . . .	11
2.2.2 Besoins non Fonctionnels . . . . .	12
2.3 Diagrammes de cas d'utilisation . . . . .	13
2.3.1 Diagramme de cas d'utilisation général de la partie web . . . . .	13
2.3.2 Diagramme de cas d'utilisation général de la partie mobile . . . . .	18
2.4 Diagrammes de navigation système . . . . .	23
2.4.1 Diagramme de navigation système partie web . . . . .	23
2.4.2 Diagramme de navigation système partie mobile . . . . .	24

2.5	Diagrammes de séquence système . . . . .	26
2.5.1	diagramme de séquence système de cas d'utilisation «S'authentifier» . . . . .	26
2.5.2	Diagramme de séquence système de cas d'utilisation «Gérer ses favoris» . .	27
2.5.3	Diagramme de séquence système de cas d'utilisation «Gérer la liste des offres» . . . . .	28
2.5.4	Diagramme de séquence système de cas d'utilisation «Gérer son profil» . .	29
	Conclusion . . . . .	30
<b>3</b>	<b>Aperçu Conceptuel</b>	<b>31</b>
	Introduction . . . . .	31
3.1	Vue statique de l'application . . . . .	31
3.1.1	Les patrons d'architecture . . . . .	31
3.1.2	Les patrons de conception . . . . .	33
3.1.3	Diagrammes de classes des entités . . . . .	34
3.2	Vue dynamique de l'application . . . . .	36
3.2.1	Diagrammes de séquence détaillé . . . . .	36
3.2.2	Diagrammes d'activités . . . . .	41
3.2.3	Diagrammes états-transitions . . . . .	42
	Conclusion . . . . .	43
<b>4</b>	<b>Réalisation</b>	<b>44</b>
	Introduction . . . . .	44
4.1	Diagramme de déploiement . . . . .	44
4.2	Environnement de développement . . . . .	45
4.2.1	Environnement matériel . . . . .	45
4.2.2	Technologies de programmation . . . . .	46
4.2.3	Langages de programmation . . . . .	47
4.2.4	Base de données . . . . .	48
4.2.5	Outils . . . . .	48
4.3	Réalisation des incrément . . . . .	50
4.3.1	Réalisation de l'incrément "Partie Web" . . . . .	50
4.3.2	Réalisation de l'incrément "Partie Mobile" . . . . .	53
4.3.3	Réalisation de l'incrément "Module IA" . . . . .	61
	Conclusion . . . . .	66
	<b>Conclusion générale</b>	<b>67</b>
	<b>Références</b>	<b>69</b>

# Table des figures

1.1	Interfaces de l'application mobile GrabExpress . . . . .	5
1.2	Interfaces de l'application mobile Roadie . . . . .	6
1.3	Interfaces de l'application mobile Nimer . . . . .	7
1.4	Modèle incrémental . . . . .	8
1.5	Diagramme de Gantt . . . . .	9
2.1	Diagramme de cas d'utilisation global de la partie web . . . . .	14
2.2	Diagramme de cas d'utilisation «Gérer la liste des utilisateurs» . . . . .	15
2.3	Diagramme de cas d'utilisation «Gérer la liste des offres» . . . . .	17
2.4	Diagramme de cas d'utilisation «Consulter tableau de bord» . . . . .	18
2.5	Diagramme de cas d'utilisation global de la partie mobile . . . . .	19
2.6	Diagramme de cas d'utilisation «Gérer des offres» . . . . .	20
2.7	Diagramme de cas d'utilisation «Gérer ses propres offres» . . . . .	22
2.8	Diagramme de navigation système partie web . . . . .	24
2.9	Diagramme de navigation système partie mobile . . . . .	25
2.10	Diagramme de séquence système de cas d'utilisation «S'authentifier» . . . . .	27
2.11	Diagramme de séquence système de cas d'utilisation «Gérer ses favoris» . . . . .	28
2.12	Diagramme de séquence système de cas d'utilisation «Gérer la liste des offres» . . . . .	29
2.13	Diagramme de séquence système de cas d'utilisation «Gérer son profil» . . . . .	30
3.1	Patron d'architecture "3 tiers" . . . . .	32
3.2	Principe du patron d'architecture MVC . . . . .	33
3.3	Principe du patron de conception Factory . . . . .	33
3.4	Principe du patron de conception Observer . . . . .	34
3.5	Diagramme de classes des entités . . . . .	35
3.6	Diagramme de séquence «Créer un nouveau utilisateur» . . . . .	37
3.7	Diagramme de séquence «Consulter la liste des demandeurs» . . . . .	38
3.8	Diagramme de séquence «Postuler à une offre» . . . . .	39
3.9	Diagramme de séquence «Lancer une discussion» . . . . .	40
3.10	Diagramme de séquence «Consulter tableau de bord» . . . . .	41
3.11	Diagramme du cas d'utilisations «Ajouter une nouvelle offre» et «Gérer les offres non confirmées» . . . . .	42
3.12	Diagramme d'états-transitions «Gérer ses offres» . . . . .	43
3.13	Diagramme d'états-transitions «Gérer des offres» . . . . .	43
4.1	Diagramme de déploiement . . . . .	45
4.2	Interface «Authentification» . . . . .	51

4.3	Interface «Tableau de bord» . . . . .	51
4.4	Interface «Liste des utilisateurs mobile» . . . . .	52
4.5	Interface « Liste des offres non confirmées» . . . . .	52
4.6	Interface «Détails d'une offre» . . . . .	53
4.7	Interface «Ajout d'un nouveau rôle» . . . . .	53
4.8	Interfaces d'accueil . . . . .	54
4.9	Interfaces d'accueil . . . . .	55
4.10	Interfaces d'accueil . . . . .	56
4.11	Interfaces d'accueil . . . . .	57
4.12	Interfaces d'accueil . . . . .	58
4.13	Interfaces d'accueil . . . . .	59
4.14	Interfaces d'accueil . . . . .	60
4.15	Interfaces d'accueil . . . . .	61
4.16	Flux de travail . . . . .	62
4.17	Caractéristiques du jeu de données initiales . . . . .	62
4.18	Répartition du jeu de données . . . . .	63
4.19	Architecture d'une cellule LSTM . . . . .	64
4.20	Précision par époque . . . . .	65
4.21	taux de précision et perte . . . . .	65

# Liste des tableaux

1.1	Tableau comparatif des applications existantes . . . . .	8
2.1	Description textuelle du cas d'utilisation «Gérer la liste des utilisateurs» . . . . .	16
2.2	Description textuelle du cas d'utilisation «Gérer la liste des offres» . . . . .	17
2.3	Description textuelle du cas «Consulter tableau de bord» . . . . .	18
2.4	Description textuelle du cas «Gérer des offres» . . . . .	21
2.5	Description textuelle du cas d'utilisation «Gérer ses propres offres» . . . . .	23
3.1	Description des classes . . . . .	36
4.1	Environnement matériel . . . . .	45

# Introduction générale

Ces dernières années, le développement mobile a connu une croissance significative, en grande partie due à l'utilisation croissante des smartphones. Les investissements importants dans ce domaine ont permis aux développeurs de créer une multitude d'applications pratiques et divertissantes, ce qui suscite un intérêt croissant pour ce secteur.

Dans ce contexte dynamique, notre application mobile de transfert de colis capitalise sur cette tendance en exploitant la puissance des smartphones et en offrant une solution innovante pour le transport de colis. En utilisant une interface conviviale et des fonctionnalités avancées, nous permettons aux voyageurs de transformer leur capacité de transport inutilisée en opportunités de revenus, tout en offrant aux clients un moyen pratique et abordable d'envoyer leurs colis.

Cette synergie entre le développement mobile en plein essor et les besoins croissants dans le domaine de la livraison de colis crée une dynamique passionnante pour notre application. Nous avons saisi l'occasion de mettre à profit les avancées technologiques et l'engouement pour les applications mobiles pour développer une plateforme robuste qui facilite la mise en relation entre les voyageurs et les clients, tout en offrant une expérience utilisateur optimale.

Ce stage a été effectué dans le cadre de notre projet de fin d'études, marquant ainsi la conclusion de notre formation en Génie Logiciel à l'Institut Supérieur des Sciences Appliquées et de Technologie de Sousse pour l'année académique 2022-2023.

Notre projet a été effectué au sein de l'entreprise « SAHTICARE », dans une durée de quatre mois.

Le présent rapport qui documente le travail effectué dans le cadre de ce stage est organisé en quatre chapitres comme suit :

Le premier chapitre, intitulé "**Présentation générale du projet.**", constitue une introduction qui présente l'entreprise d'accueil, la problématique à résoudre, la solution proposée et les objectifs de notre projet. Il comprend également une étude de l'existant et une description du processus de développement de l'application.

Le deuxième chapitre, intitulé "**Spécification des besoins**", définit les acteurs de notre application et spécifie les besoins fonctionnels et non fonctionnels auxquels notre application doit répondre, tout en présentant ses principales fonctionnalités.

Le troisième chapitre, intitulé “**Aperçu conceptuel**”, décrit les schémas conceptuels et l’architecture adoptée pour la solution proposée, avec une explication du comportement dynamique de l’application.

Le quatrième chapitre, intitulé “**Réalisation**”, présente l’environnement de développement, les outils utilisés, ainsi que la visualisation des résultats obtenus à travers les principales interfaces du logiciel.

Enfin, notre projet se conclura par une synthèse dans laquelle nous mettrons en avant les différents atouts de l’application réalisée et les perspectives d’amélioration qui peuvent être envisagées.

# Chapitre 1

## Présentation générale du projet

### Introduction

Dans ce chapitre, nous allons commencer par présenter l'organisme d'accueil. Ensuite, nous allons examiner en détail la problématique de notre stage et la solution que nous proposons, qui vise à résoudre les limites des solutions existantes. Enfin, nous allons décrire le calendrier de notre stage à l'aide du diagramme de Gantt et expliquer le processus de développement que nous allons suivre.

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

“SAHTICARE”, est une jeune entreprise tunisienne de renom, fondée en 2021. Elle se spécialise dans la fourniture de solutions technologiques innovantes et personnalisées pour les secteurs de la santé, de la mobilité, des banques, des assurances et d'autres industries.

Grâce à une équipe hautement qualifiée et passionnée par l'innovation technologique, “SAHTICARE” développe des solutions logicielles sur mesure pour répondre aux besoins de ses clients. En tant qu'entreprise pionnière dans son domaine, “SAHTICARE” est résolue à rester à l'avant garde des tendances technologiques et à anticiper les besoins futurs de ses clients.



Logo de SAHTICARE

#### 1.1.1 Les fondateurs de SAHTICARE

Les fondateurs de «SAHTICARE» sont :

- Mohamed Taha Ben Mhenni : Chef de la direction.
- Iheb Sahloul : Directeur technique.
- Saifeddine Ben Mhenni : Chef des opérations.

### 1.1.2 Fiche technique

- Adresse : Pole Technologique & Industriel 5011 Monastir, Tunisia.
- Téléphone : (+216) 28 796 295.
- Site web : vitah.tech

Ce stage a été effectué au sein de SAHTICARE sous la direction de l'équipe de développement.

### 1.1.3 Cadre général du projet

Au cours des dernières années, nous avons assisté à une croissance exponentielle du nombre de voyageurs ainsi que du nombre de voyages effectués sans avoir une capacité de transport complètement utilisée. Cette tendance a ouvert la voie au développement d'une application mobile qui permet aux voyageurs d'offrir leurs services pour le transfert de colis.

### 1.1.4 Problématique

La tendance croissante du transfert international de colis a créé un besoin pressant pour les voyageurs de disposer d'une plateforme spécialement conçue pour faciliter les échanges de services entre eux et les clients intéressés par ces services. Actuellement, les réseaux sociaux existants ne répondent pas de manière adéquate à cette demande croissante. Ils ne fournissent pas les fonctionnalités nécessaires pour faciliter efficacement ces transactions.

D'une part, les voyageurs souhaitent pouvoir publier facilement leurs offres de services de transfert de colis, avec des détails tels que les destinations, les dates disponibles et les tarifs proposés. Cependant, sur les réseaux sociaux traditionnels, ces informations peuvent facilement se perdre dans le flux d'activités et ne pas atteindre leur public cible de manière optimale. D'autre part, les clients qui ont besoin d'envoyer des colis à l'étranger cherchent une plateforme centralisée où ils peuvent trouver des voyageurs fiables et intéressés par le transport de leurs colis. Malheureusement, les réseaux sociaux classiques ne permettent pas une recherche ciblée de voyageurs en fonction de critères spécifiques tels que la destination, la disponibilité et les évaluations des expériences passées.

## 1.2 Etude de l'existant

Cette étape revêt une importance capitale lors de la réalisation d'un projet. Pendant cette phase, nous avons effectué une collecte d'informations approfondie sur les solutions concurrentes proposées aux entreprises de distribution, offrant des services similaires à ceux que nous envisageons de mettre en place. Notre objectif était d'identifier et d'éviter leurs points faibles, tout en proposant un ensemble de fonctionnalités plus étendu et plus avancé. Nous avons donc entrepris une démarche de recherche et d'analyse afin de recueillir des informations essentielles sur ces solutions concurrentes, ce qui nous a permis d'élaborer une solution plus solide et plus compétitive.

### 1.2.1 Analyse de l'existant

Dans ce paragraphe, nous présentons les différentes solutions existantes étudiées dans le cadre de notre projet.

- **GrabExpress** : GrabExpress fait partie de Grab, une entreprise basée à Singapour. Grab a été fondée en 2012 et a lancé ses services de livraison de colis, y compris GrabExpress, dans les années suivantes.

### 1. Points forts :

- Interfaces intuitives et faciles à utiliser.
- Communication directe avec les livreurs : L'application Grab permet une communication directe entre les utilisateurs et les livreurs, ce qui facilite la coordination et la résolution rapide de tout problème ou changement éventuel.

### 2. Points faibles :

- Limitations géographiques : La disponibilité de GrabExpress est limitée, elle est disponible seulement dans l'Asie.
- Problèmes de localisation ou d'itinéraire : Comme GrabExpress utilise la géolocalisation et les itinéraires pour organiser les livraisons, il peut y avoir des problèmes potentiels liés à la précision de la localisation ou aux itinéraires suggérés, ce qui peut entraîner des retards ou des erreurs de livraison.

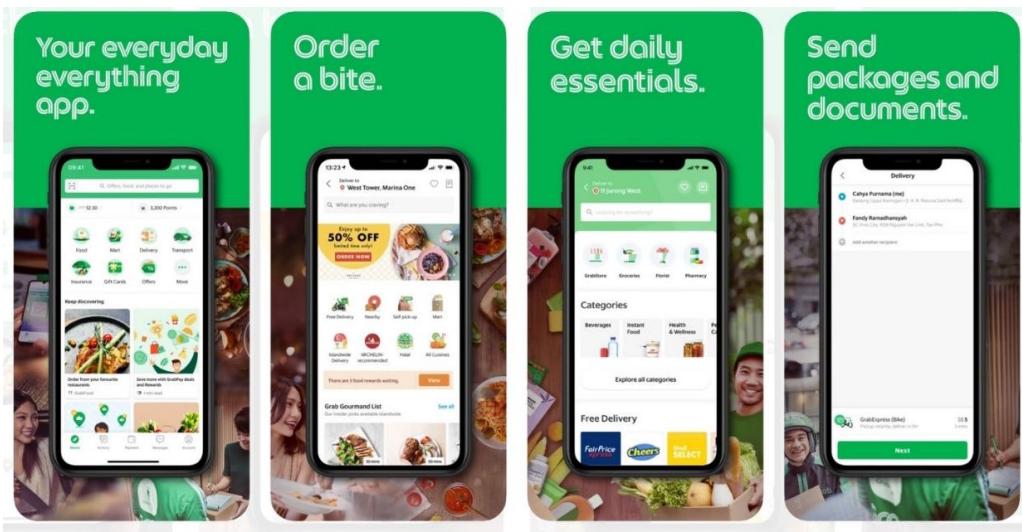


FIGURE 1.1 – Interfaces de l'application mobile GrabExpress

- **Roadie** : Roadie a été fondée en 2014 aux États-Unis. L'application a été créée pour offrir un service de livraison de colis basé sur la mise en relation de voyageurs effectuant des trajets spécifiques. L'application permet aux utilisateurs de comparer les tarifs d'expédition et de choisir la méthode de livraison la plus adaptée à leurs besoins. Elle offre également un suivi en temps réel des colis, ce qui permet aux utilisateurs de surveiller l'état de leurs envois.

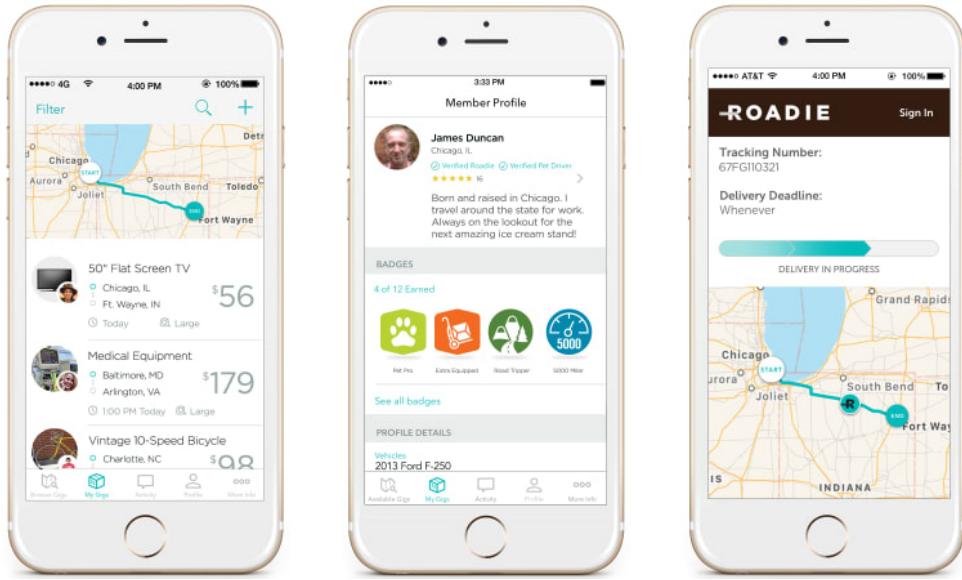
### 1. Points forts :

- Simplicité d'utilisation : Roadie propose une application conviviale avec une interface intuitive, ce qui facilite la création et la gestion des livraisons pour les utilisateurs.
- Flexibilité des horaires : Les utilisateurs de Roadie ont la possibilité de planifier les livraisons selon leurs propres horaires et disponibilités, offrant une plus grande flexibilité par rapport aux services de livraison traditionnels.

### 2. Points faibles :

- Limitations géographiques : La disponibilité de Roadie est limitée, elle est disponible seulement dans l'Amérique spécialement aux Etats-Unis.

- Gestion des problèmes et du service client : Certains utilisateurs ont exprimé des préoccupations concernant la gestion des problèmes et ont noté un manque de réactivité du service client de Roadie.



**FIGURE 1.2 – Interfaces de l'application mobile Roadie**

- **Nimber** : Nimber a été fondée en 2014 en Norvège. L'application a été créée pour permettre aux utilisateurs de trouver des voyageurs pour effectuer des livraisons de colis pendant leurs trajets.

### 1. Points forts :

- Diversité des envois possibles : Nimber permet de livrer une variété d'articles, allant des petits colis aux articles plus volumineux, offrant ainsi une certaine adaptabilité aux besoins des utilisateurs.
- Personnalisation des trajets : Les utilisateurs de Nimber ont la possibilité de personnaliser leurs trajets de livraison en définissant leurs propres itinéraires et horaires, offrant ainsi une expérience plus flexible et adaptée à leurs besoins.

### 2. Points faibles :

- Limitations géographiques : La disponibilité et la couverture géographique de Nimber peuvent être limitées dans certaines régions, en fait elle est disponible seulement dans les pays d'europe.
- Interface utilisateur complexe : Certaines personnes peuvent trouver l'interface utilisateur de Nimber complexe ou difficile à comprendre, ce qui peut rendre l'utilisation de l'application moins intuitive pour certains utilisateurs.

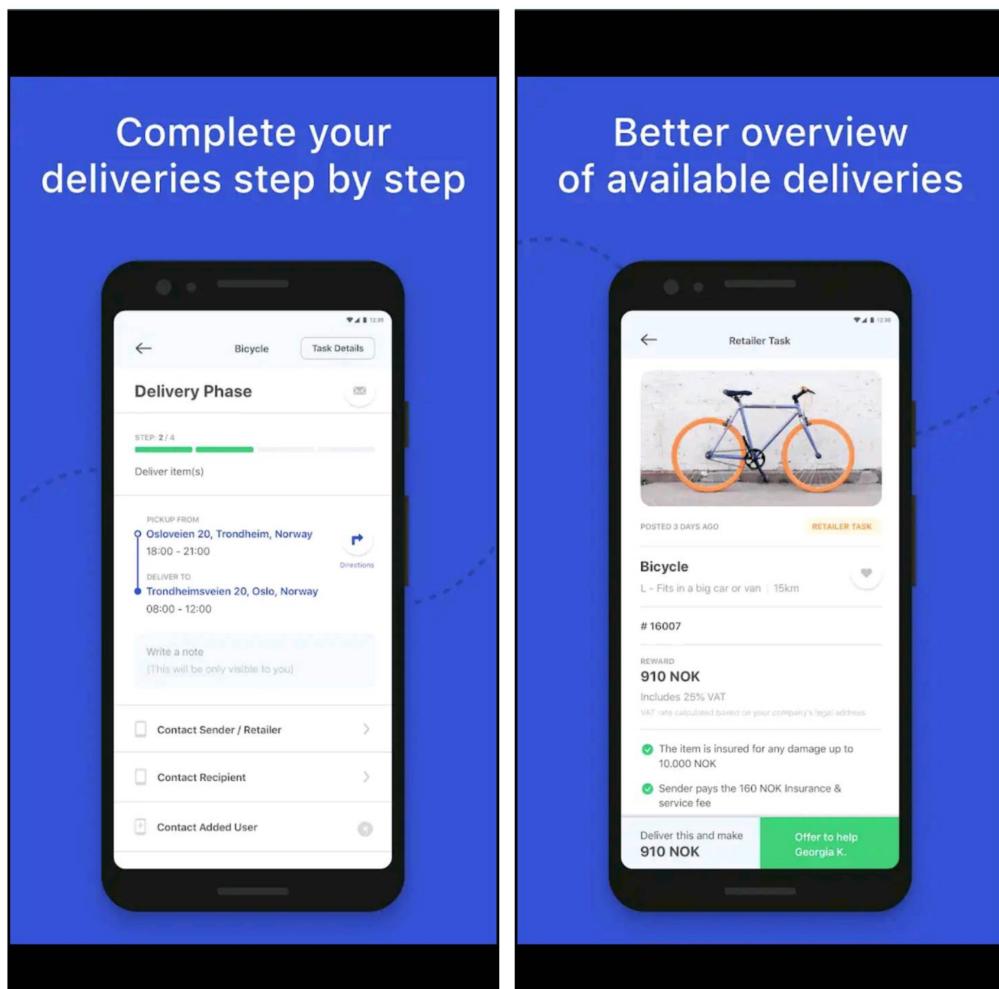


FIGURE 1.3 – Interfaces de l’application mobile Nimmer

### 1.2.2 Critique de l’existant

Malgré les solutions mentionnées précédemment qui répondent aux besoins des utilisateurs en transportant des colis à l’international, nous avons identifié certaines lacunes et points faibles. Ainsi, nous prévoyons d’apporter nos améliorations afin de corriger les incohérences et renforcer ces solutions existantes notamment :

- Manque de disponibilité de ces applications en Tunisie.
- Difficulté d’usage : Absence d’aide sur les fonctionnalités.
- Abscence de communication entre demandeur de service et fournisseur.

### 1.2.3 Solution envisagée

	GrabExpress	Roadie	Nimber	Notre solution
<b>Ergonomie</b>	oui	Oui	Non	Oui
<b>Rapidité d'exécution</b>	Oui	Oui	Non	Oui
<b>Disponibilité en tunisie</b>	Non	Non	Non	Oui
<b>Messagerie</b>	Oui	Non	Non	Oui

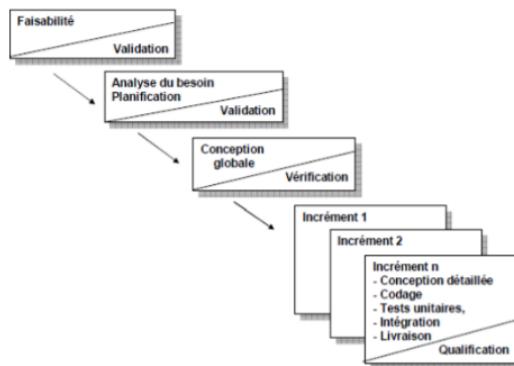
**TABLE 1.1** – Tableau comparatif des applications existantes

En développant cette application, nous répondrons au besoin croissant d'une plateforme dédiée pour le transfert de colis à l'international, en offrant aux voyageurs un moyen de monétiser leurs voyages et en permettant aux clients de bénéficier d'un service fiable et pratique pour l'envoi de leurs colis.

## 1.3 Processus de développement

### 1.3.1 Développement incrémental

Nous avons choisi comme processus de développement de notre logiciel le modèle incrémental. L'apport du modèle incrémental est le partitionnement du modèle du logiciel en un ensemble d'incréments individuellement cohérents, fonctionnels et livrables au client.



**FIGURE 1.4** – Modèle incrémental

### 1.3.2 Incréments du logiciel

Les différents incréments de notre logiciel sont :

- **Partie Mobile** : Elle permet aux utilisateurs de publier ses offres tout en s'authentifiant afin d'avoir une liste contenant d'autres utilisateurs qui ont postulé pour ses offres. Ces utilisateurs choisissent un ou plusieurs utilisateurs convenables pour transporter leurs colis à l'international.
- **Partie Web** : c'est la où les administrateurs peuvent gérer et consulter l'état de la partie mobile de l'application. Ils peuvent gérer les utilisateurs, confirmer et gérer les offres et

consulter le tableau de bord.

- **Module IA** : Le module IA a pour fonction de détecter les termes inappropriés lors de la confirmation de l'offre, permettant ainsi de prendre une décision quant à son acceptation ou son refus.

### 1.3.3 planification anticipée des tâches

Le diagramme de gantt est un outil graphique qui représente la gestion du projet dans le temps, ce qui facilite sa réalisation. En effet, la figure 1.5 représente l'avancement des activités et des tâches exécutées tout le long de notre projet.

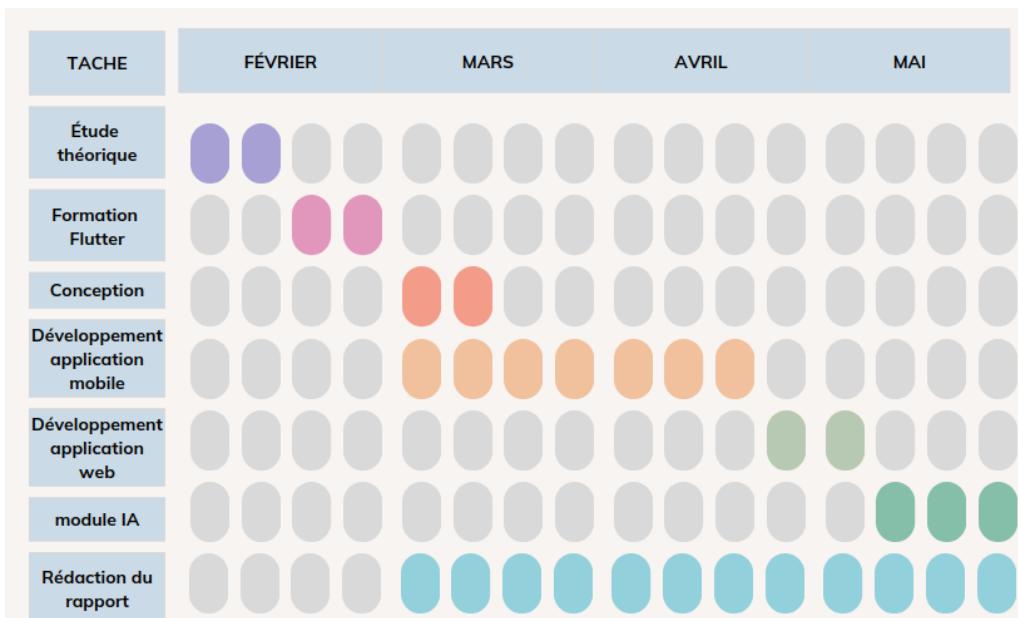


FIGURE 1.5 – Diagramme de Gantt

## Conclusion

Dans ce chapitre, nous avons présenté le contexte de notre projet en commençant par décrire l'organisme d'accueil. Ensuite, nous avons exposé la problématique que nous avons identifiée, avant de proposer notre solution et les objectifs que nous souhaitons atteindre. Nous avons également réalisé une analyse des applications existantes dans ce domaine.

En outre, nous avons suivi un processus de développement spécifique tout au long du projet et nous avons présenté l'avancement des activités dans ce cadre.

# Chapitre 2

## Analyse et spécification des besoins

### Introduction

L'analyse et la spécification des besoins constituent une étape cruciale dans le processus de développement d'un projet. Cette étape permet de passer de la vision abstraite du sujet à une définition concrète du produit, afin de mieux comprendre le travail requis.

Dans ce chapitre, nous allons commencer par identifier les différents acteurs impliqués et leurs responsabilités respectives. Ensuite, nous allons détailler les besoins fonctionnels et non fonctionnels que notre système doit satisfaire, afin de fournir une plateforme répondant aux attentes des utilisateurs.

### 2.1 Identification des acteurs

Nous prévoyons de développer deux applications distinctes pour notre solution : une application web (Back-office) et une application mobile disponible sur les plateformes iOS et Android. Les acteurs sont des utilisateurs humains ou des systèmes informatiques qui interagissent dans notre logiciel.

#### 2.1.1 Partie Web

Elle sera utilisée par :

— **Les acteurs primaires :**

- Administrateur : Cet acteur assume le rôle de responsable de notre système. Il a la capacité d'effectuer la majorité des opérations de gestion, telles que la gestion des utilisateurs, la gestion des offres et l'analyse des données.

— **Les acteurs secondaires :**

- Système de confirmation des offres : Notre objectif est de développer un système intelligent qui permettra de confirmer de manière efficace les offres créées par les utilisateurs de la partie mobile.

#### 2.1.2 Partie Mobile

Elle sera utilisée par :

- Utilisateur : Son rôle dans l'application est de gérer ses offres. Il peut effectuer les actions suivantes : rechercher des utilisateurs à proximité, consulter les offres disponibles, postuler à des offres, recevoir des candidatures d'autres utilisateurs, sélectionner parmi ces candidatures et même entrer en contact avec eux.

## 2.2 Spécification des besoins

Le but de cette section est de recueillir, analyser et définir les besoins de niveau supérieur ainsi que les caractéristiques du projet. L'accent est mis sur les exigences fonctionnelles et non fonctionnelles requises pour offrir des services conviviaux et faciles à utiliser.

### 2.2.1 Les besoins fonctionnels

Les besoins fonctionnels se réfèrent aux actions spécifiques que notre système doit effectuer et aux attentes des utilisateurs pour notre application en cours de développement.

Concernant notre **partie web (Back-office)**, les besoins fonctionnels se résument aux points suivants :

- **S'authentifier** : L'utilisateur back-office peut accéder à son espace privé en saisissant ses identifiants.
- **Gérer son profil** : L'utilisateur a la possibilité de modifier ses informations personnelles, notamment son nom, sa photo de profil et ses coordonnées de contact.
- **Gérer la liste des utilisateurs** :
  - **Gérer la liste des utilisateurs mobile** :
    - Consulter la liste des utilisateurs mobile.
    - Créer un nouveau compte utilisateur mobile.
    - Consulter les informations de l'utilisateur mobile.
    - Activer / Désactiver le compte utilisateur mobile.
  - **Gérer la liste des utilisateurs back-office** :
    - Consulter la liste des utilisateurs back-office.
    - Créer un nouveau utilisateur back-office.
    - Supprimer un utilisateur back-office.
    - Modifier le rôle d'utilisateur back-office.
- **Gérer la liste des offres** :
  - Consulter la liste des offres confirmées.
  - Consulter la liste des offres non confirmées.
  - Confirmer une offre.
  - Supprimer une offre.
  - Consulter une offre en détails.
- **Gérer les rôles des administrateurs** :
  - Consulter la liste des rôles des administrateurs.
  - Modifier un rôle d'utilisateur back-office.
  - Supprimer un rôle d'utilisateur back-office.
  - Ajouter un nouveau rôle.
- **Consulter tableau de bord** : cette fonctionnalité fournit une vue d'ensemble des statistiques de notre partie mobile ainsi que le nombre d'utilisateurs de notre partie web.

En ce qui concerne notre **partie mobile**, les besoins fonctionnels fondamentaux sont

récapitulés ci-dessous :

- **S'inscrire dans l'application** : L'utilisateur peut s'inscrire dans l'application mobile et bénéficier des fonctionnalités existantes en tant que membre inscrit.
- **S'authentifier** : L'utilisateur peut accéder à l'application en saisissant ses identifiants.
- **Gérer son profil** :
  - Consulter ses informations personnelles.
  - Consulter la liste de ses propres offres.
  - Modifier ses informations personnelles.
  - Changer son mot de passe.
  - Changer sa position actuelle.
- **Gérer des offres** :
  - Consulter la liste des offres valables.
  - Consulter une offre en détails.
  - appliquer à une offre.
  - Consulter l'état de ses demandes.
- **Gérer ses offres** :
  - Ajouter une nouvelle offre.
  - Supprimer une offre.
  - Modifier une offre.
  - Consulter la liste des demandeurs.
  - Accepter / Refuser un demandeur.
  - Consulter la liste des demandeurs acceptés.
- **Gérer ses favoris** :
  - Consulter la liste de ses offres favorites.
  - Ajouter une offre comme favori.
  - Supprimer une offre de sa liste de favoris.
  - Consulter la liste de ses utilisateurs favoris.
  - Ajouter un utilisateur comme favori.
  - Supprimer un utilisateur de sa liste de favoris.
- **Consulter les profils d'autres utilisateurs** : L'utilisateur a le droit de consulter les profils d'autres utilisateurs de notre application, que ce soit en jetant un œil sur leurs informations personnelles ou en examinant leurs propres offres.
- **Échanger des messages** :
  - Consulter ses anciens messages.
  - Lancer une nouvelle discussion.
  - Recevoir des messages de la part d'autres utilisateurs.
  - Envoyer des messages.
- **Chercher des utilisateurs à proximité** : L'utilisateur a la possibilité de modifier ses informations personnelles, telles que la latitude et la longitude, afin de les afficher sur la carte graphique et ainsi montrer les utilisateurs adjacents.

### 2.2.2 Besoins non Fonctionnels

Les besoins fonctionnels décrivent les fonctionnalités concrètes du produit, tandis que les besoins non fonctionnels sont des critères de qualité pour l'exécution de ces besoins fonctionnels. Par conséquent, notre application a certaines exigences spécifiques qui doivent être respectées.

- **Exigences de qualité** :

- nos interfaces doivent s'adapter aux différents appareils et résolutions d'écran pour garantir une expérience utilisateur optimale sur les tablettes et les smartphones.
- notre application doit répondre rapidement aux interactions de l'utilisateur, en évitant les délais de chargement excessifs et en fournissant un retour visuel approprié.
- Les éléments d'interface doivent être cohérents dans leur disposition, leur style et leur comportement afin de créer une expérience utilisateur fluide et prévisible.

— **Exigences de performance :**

Les futurs utilisateurs de notre système peuvent être des centaines, Le système doit donc supporter :

- La gestion de plus de 500 comptes.
- La connexion simultanée de plus de 50 utilisateurs.
- Les opérations de création simples telles que l'ajout des nouveaux utilisateurs et offres ne doivent pas prendre plus de 2 secondes.
- **Exigences de portabilité** : Nous allons choisir des technologies de développement qui offrent une prise en charge pour la création d'applications multiplateformes, telles que les frameworks hybrides.
- **Exigences d'extensibilité/maintenabilité** : Il est essentiel que notre code soit clair, bien commenté et correctement documenté afin de faciliter les futures évolutions et améliorations de notre application.
- **Contraintes de conception** : Pour garantir une gestion efficace des fichiers dans notre système, nous prévoyons d'adopter une approche d'enregistrement externe. Ainsi, les fichiers tels que les images seront stockés sur une source externe, tandis que leurs URL correspondantes seront référencées dans la base de données. Cela permettra d'optimiser l'utilisation de la base de données tout en assurant l'accès aux fichiers nécessaires via leurs liens URL associés.

## 2.3 Diagrammes de cas d'utilisation

Au niveau de cette partie, nous traduisons les besoins précédemment dégagés sous forme de diagrammes de cas d'utilisation. Les diagrammes de cas d'utilisation sont la première étape UML d'analyse d'un système et il faut la faire avec soin afin de produire un système conforme aux attentes des clients. Il permet de représenter les fonctions à effectuer par les différents acteurs sans s'inquiéter de la manière dont les fonctions vont être implémentés au niveau du système. En ce qui suit, nous présentons le diagramme de cas d'utilisation général de notre solution sous forme de packages puis nous procédons à le détailler et à expliquer les principaux cas d'utilisation.

### 2.3.1 Diagramme de cas d'utilisation général de la partie web

Le diagramme de cas d'utilisation général nous permet d'exposer une vision globale du comportement fonctionnel de notre application et son interaction avec les différents acteurs.

La figure 2.1 illustre le diagramme de cas d'utilisation général de notre partie web.

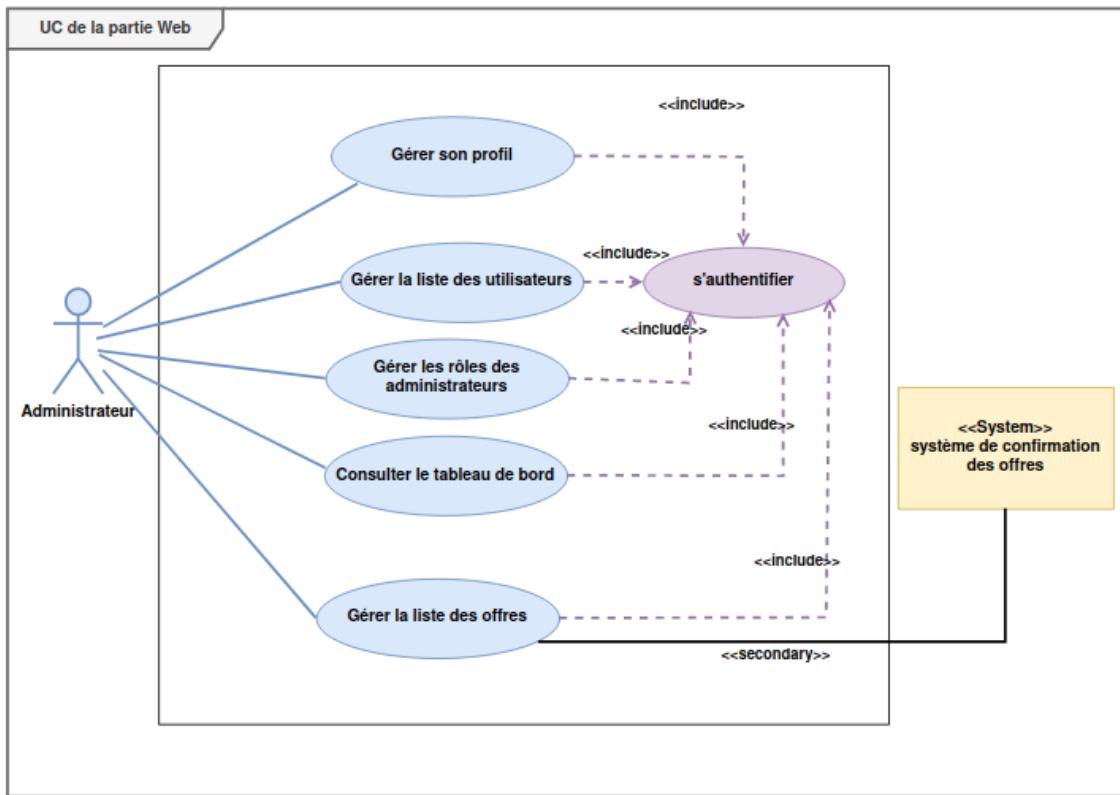
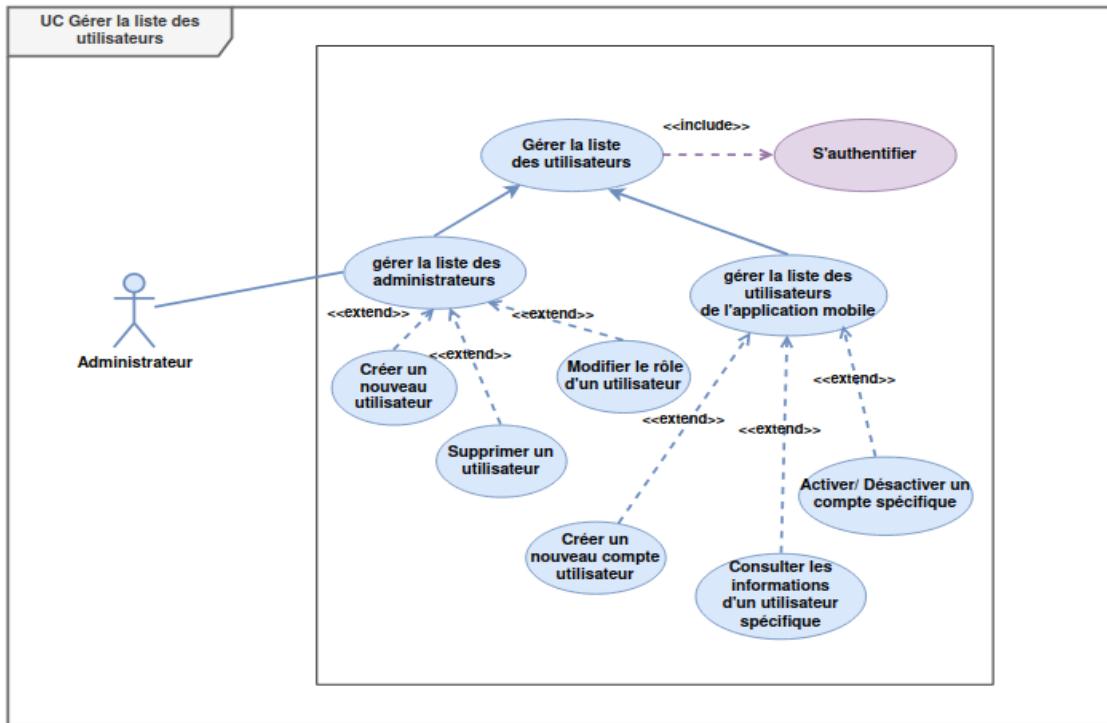


FIGURE 2.1 – Diagramme de cas d'utilisation global de la partie web

— **Raffinement de cas d'utilisation : «Gérer la liste des utilisateurs»**

Nous présentons, à travers la figure 2.2 le diagramme de cas d'utilisation relatif à «Gérer la liste des utilisateurs».



**FIGURE 2.2 – Diagramme de cas d'utilisation «Gérer la liste des utilisateurs»**

La description de scénario de cas d'utilisation : «Gérer la liste des utilisateurs» est donnée par le tableau 2.3 :

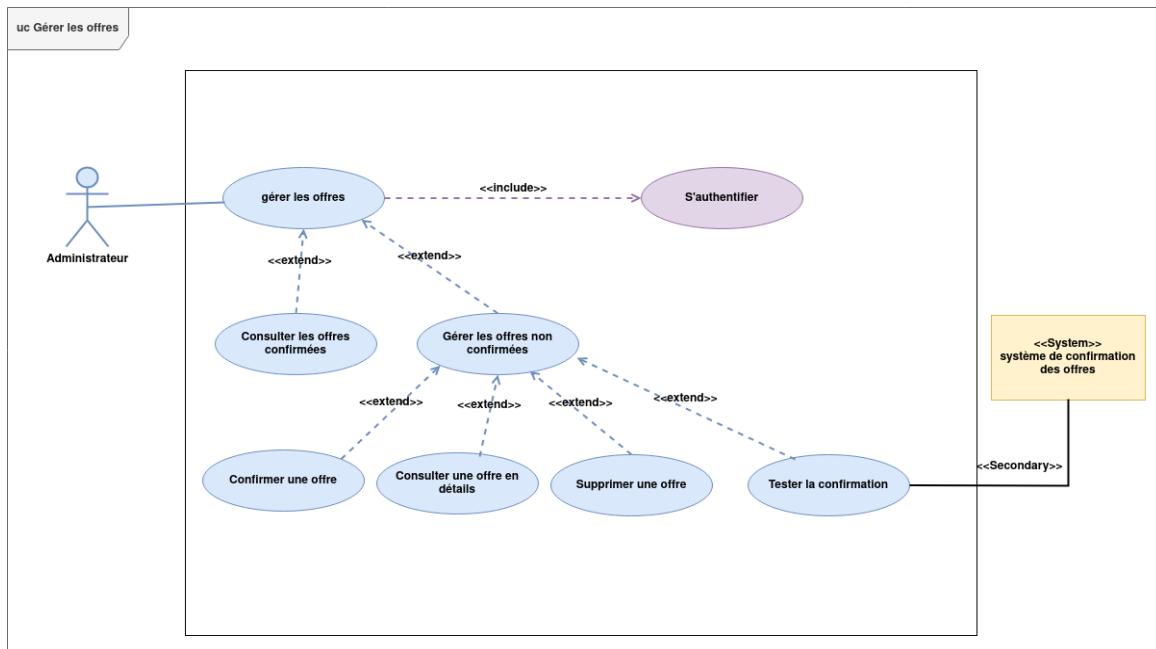
Titre	Gérer la liste des utilisateurs
Acteurs	Administrateurs.
Objectif	En accédant à une liste exhaustive des utilisateurs de l'application mobile, l'administrateur dispose de diverses possibilités de gestion. Il peut consulter les informations détaillées de chaque utilisateur, créer nouveau utilisateur et activer/désactiver un compte spécifique. De plus, en consultant la liste des administrateurs, l'administrateur a le droit d'ajouter un nouveau administrateur, supprimer un administrateur existant ou modifier son rôle.
Précondition	Administrateur authentifié.
Post-condition	utilisateur ajouté ou administrateur supprimé.

Scénario nominal	<p><b>[S1] : Ajouter un nouveau utilisateur</b></p> <ol style="list-style-type: none"> <li>1. L'administrateur demande l'interface de création d'un nouveau utilisateur.</li> <li>2. Le système affiche le formulaire.</li> <li>3. L'administrateur remplit le formulaire, puis valide en cliquant sur le bouton "Create".</li> <li>4. La création d'un nouveau utilisateur est faite avec succès et l'administrateur est orienté vers l'interface de la liste des utilisateurs.</li> </ol> <p><b>[S2] : Supprimer un administrateur</b></p> <ol style="list-style-type: none"> <li>1. L'administrateur demande la liste des administrateurs.</li> <li>2. Le système affiche la liste des administrateurs.</li> <li>3. L'administrateur choisit l'utilisateur à supprimer et clique sur le bouton "delete".</li> <li>4. Le système renvoie la liste des administrateurs synchronisée.</li> </ol>
Scénario alternatif	<p><b>[S1].b.A1 : Des champs sont vides</b></p> <ul style="list-style-type: none"> <li>— 1. Le système envoie un message d'erreur pour indiquer que ces champs doivent être remplis.</li> <li>— 2. Le scénario reprend à partir de l'étape b.</li> </ul> <p><b>[S1].b.A2 : Les valeurs entrées ne respectent pas les conditions de saisie</b></p> <ul style="list-style-type: none"> <li>— 1. Le système envoie un message d'erreur.</li> <li>— 2. Le scénario reprend à partir de l'étape b.</li> </ul> <p><b>[S1].b.A3 : L'adresse email saisie est déjà utilisée par un autre compte</b></p> <ul style="list-style-type: none"> <li>— 1. Le système informe l'administrateur du problème.</li> <li>— 2. Le scénario reprend à partir de l'étape b.</li> </ul> <p><b>[S2].b.A1 : La liste des administrateurs est vide.</b></p>
Exigences supplémentaires	<ul style="list-style-type: none"> <li>— Les mots de passe ne doivent pas être enregistrés en clair dans la base de données. Ils doivent être hachés.</li> <li>— L'interface doit être ergonomique.</li> <li>— Les messages d'erreur doivent être compréhensibles et clairs.</li> </ul>

**TABLE 2.1** – Description textuelle du cas d'utilisation «Gérer la liste des utilisateurs»

— **Raffinement de cas d'utilisation : «Gérer la liste des offres»**

Nous présentons, à travers la figure 2.3 le diagramme de cas d'utilisation relatif à «Gérer la liste des offres».



**FIGURE 2.3 – Diagramme de cas d'utilisation «Gérer la liste des offres»**

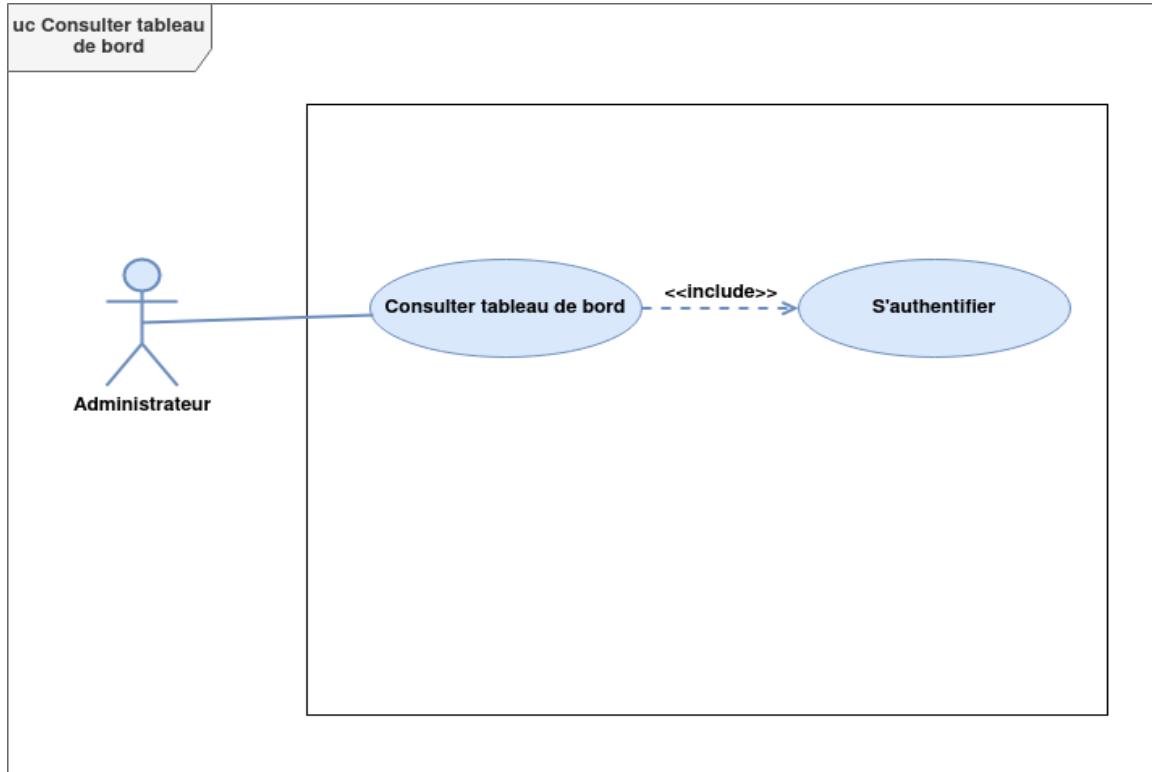
La description de scénario de cas d'utilisation : «Gérer la liste des offres» est donnée par le tableau 2.2 :

Titre	Gérer la liste des offres
Acteurs	Administrateurs.
Objectif	L'acteur doit avoir la possibilité d'afficher la liste des offres non confirmées, confirmer ou supprimer une entité de cette liste, consulter une en détails et tester sa confirmation avec le module intelligent. Il a également la possibilité de consulter la liste des offres confirmées.
Précondition	Administrateur authentifié.
Post-condition	liste des offres consultées ou offre confirmée ou offre supprimée.
Scénario nominal	<p>Si l'acteur :</p> <p><b>[S1] : Demande la liste des offres confirmées</b></p> <ol style="list-style-type: none"> <li>1. Le système affiche la liste des offres confirmées.</li> </ol> <p><b>[S2] : Demande la liste des offres non confirmées</b></p> <ol style="list-style-type: none"> <li>1. Le système affiche la liste des offres non confirmées.</li> <li>2. L'acteur a la possibilité de confirmer une offre, consulter une en détails, supprimer une ou tester sa confirmation.</li> </ol>
Scénario alternatif	<p><b>[S2-2a] : L'acteur supprime une offre</b></p> <ol style="list-style-type: none"> <li>1. Le système affiche un message de confirmation.</li> <li>2. L'acteur confirme la suppression.</li> <li>3. Le système supprime l'offre.</li> </ol>

**TABLE 2.2 – Description textuelle du cas d'utilisation «Gérer la liste des offres»**

— **Raffinement de cas d'utilisation : «Consulter tableau de bord»**

Nous présentons, à travers la figure 2.4 le diagramme de cas d'utilisation relatif à «Consulter tableau de bord».



**FIGURE 2.4** – Diagramme de cas d'utilisation «Consulter tableau de bord»

La description de scénario de cas d'utilisation : «Consulter tableau de bord» est donnée par le tableau 2.3 :

Titre	Gérer la liste des utilisateurs
Acteurs	Adimnistrateurs.
Objectif	L'acteur peut avoir un aperçu général sur la santé de l'application mobile grâce aux statistiques contenues dans le tableau de bord.
Précondition	Administrateur authentifié.
Post-condition	Tableau de bord consulté.
Scénario nominal	<ol style="list-style-type: none"> <li>1. L'acteur demande l'interface du tableau de bord.</li> <li>2. Le système affiche l'interface contenant les statistiques à propos de notre application.</li> </ol>

**TABLE 2.3** – Description textuelle du cas «Consulter tableau de bord»

### 2.3.2 Diagramme de cas d'utilisation général de la partie mobile

La figure 2.5 montre le diagramme de cas d'utilisation général de notre partie mobile.

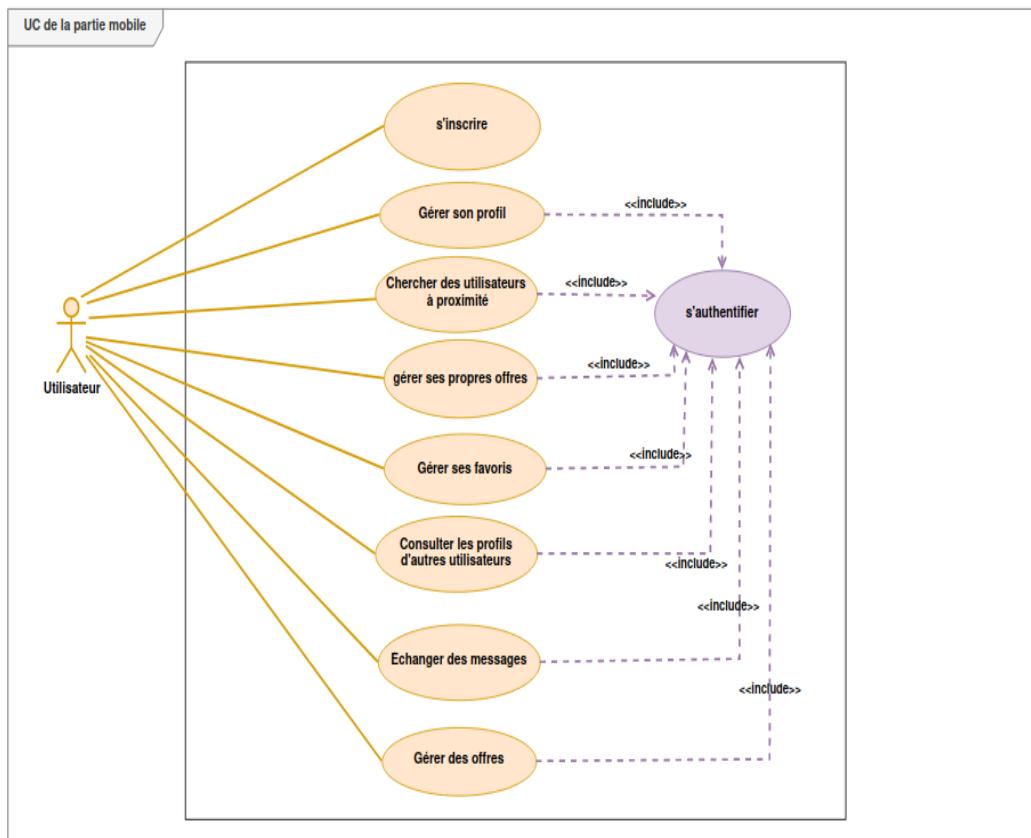
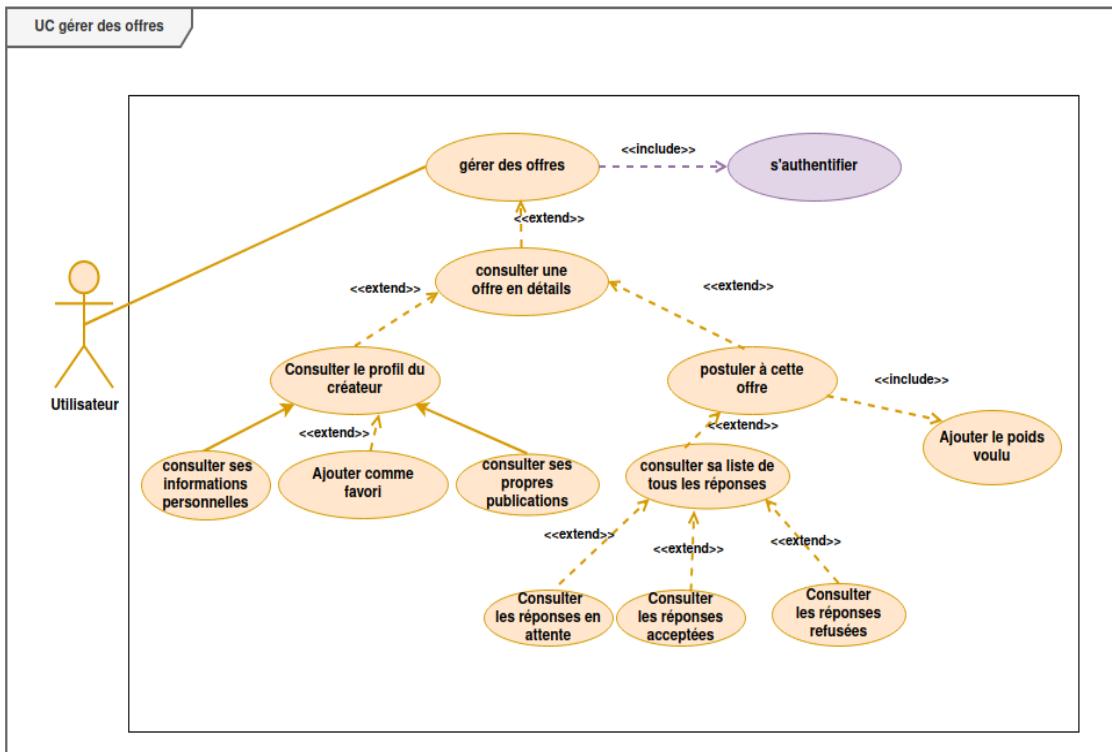


FIGURE 2.5 – Diagramme de cas d'utilisation global de la partie mobile

— **Raffinement de cas d'utilisation : «Gérer des offres»**

Nous présentons, à travers la figure 2.6, le diagramme de cas d'utilisations relatif à «Gérer des offres».



**FIGURE 2.6 – Diagramme de cas d'utilisation «Gérer des offres»**

La description de scénarios de cas d'utilisation : «Gérer des offres» est donnée par le tableau 2.4 :

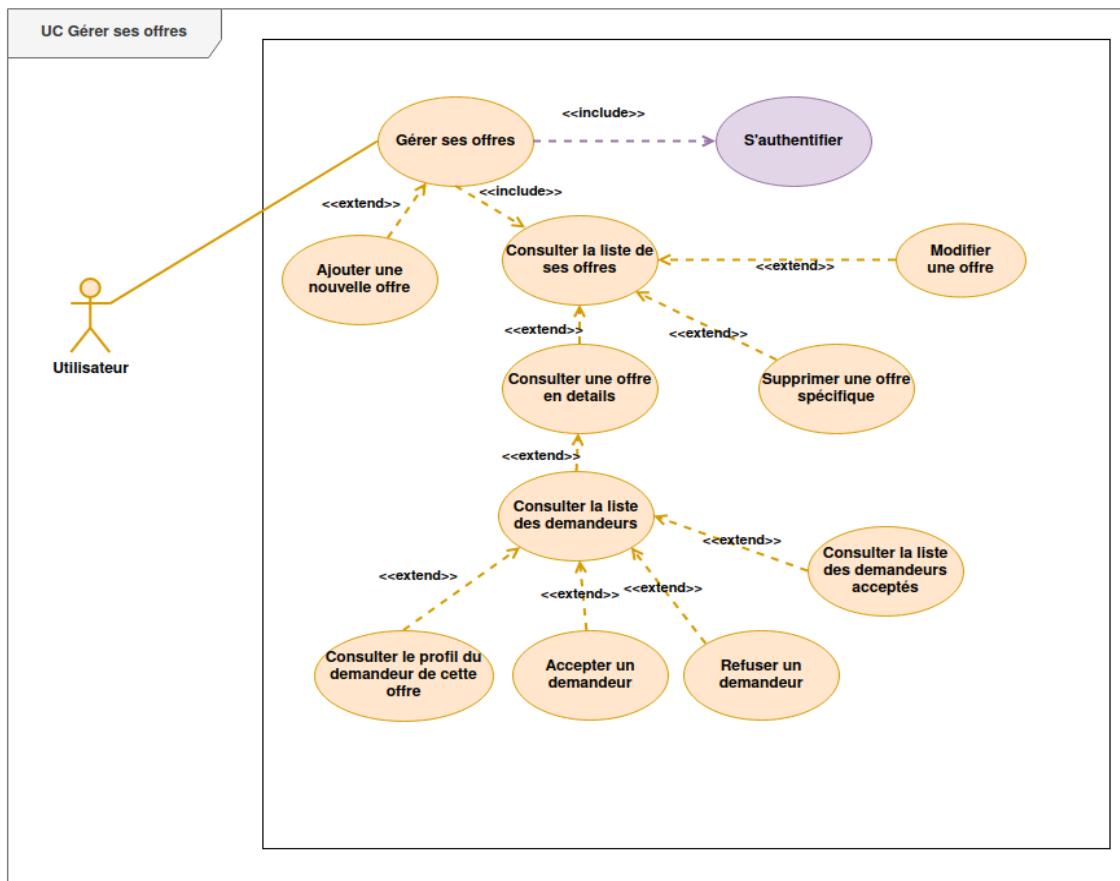
Titre	Gérer des offres.
Acteurs	Utilisateurs.
Objectif	En consultant les offres, l'utilisateur dispose de diverses possibilités de gestion. Il peut consulter une offre en détails, de même il a le droit de consulter le profil de son créateur, appliquer à cette offre en spécifiant le poids voulu, ajouter cette offre comme étant favori et même ajouter son créateur aussi.
Précondition	Utilisateur authentifié
Post-condition	offre ajoutée dans la lsite des applications ou offre ajoutée dans la lsite des favoris.

Scénario nominal	<p><b>[S1] : Appliquer à une offre</b></p> <ol style="list-style-type: none"> <li>1. L'utilisateur clique sur une offre spécifique.</li> <li>2. Le système affiche l'interface contenant les détails de cette offre.</li> <li>3. L'utilisateur clique sur le bouton "APPLY"</li> <li>4. Le système affiche une formulaire.</li> <li>5. L'utilisateur saisie le poids voulu et clique sur le bouton "SEND REQUEST".</li> <li>6. Le système accepte la demande et propose à l'utilisateur de visiter ses réponses.</li> </ol> <p><b>[S2] : Ajouter offre comme favori</b></p> <ol style="list-style-type: none"> <li>1. L'utilisateur choisit une offre.</li> <li>2. Le système affiche cette offre en détails.</li> <li>3. L'utilisateur clique sur l'icone du coeur.</li> <li>4. Le système ajoute l'offre à la liste des favoris.</li> </ol>
Scénario alternatif	<p><b>[S1].a.A1, [S2].a.A1 : La liste des offres est vide.</b></p> <p><b>[S1].b.A1 : bouton "APPLY" est désactivé</b></p> <ul style="list-style-type: none"> <li>— 1. Le scénario reprend à partir de l'étape b.</li> </ul> <p><b>[S1].e.A1 : Les valeurs entrées ne respectent pas les conditions de saisie</b></p> <ul style="list-style-type: none"> <li>— 1. Le système envoie un message d'erreur.</li> <li>— 2. Le scénario reprend à partir de l'étape b.</li> </ul> <p><b>[S2].b.A3 : L'offre déjà existe dans la liste des favoris</b></p> <ul style="list-style-type: none"> <li>— 1. L'icone du coeur ne s'affiche pas.</li> <li>— 2. Le scénario reprend à partir de l'étape b.</li> </ul>
Exigences supplémentaires	<p>Le bouton "APPLY" devient désactivé lorsque :</p> <ul style="list-style-type: none"> <li>— L'utilisateur a déjà appliqué à cette offre.</li> <li>— l'offre est expirée.</li> </ul>

**TABLE 2.4 – Description textuelle du cas «Gérer des offres»**

— **Raffinement de cas d'utilisation : «Gérer ses propres offres»**

Nous présentons, à travers la figure 2.7 le diagramme de cas d'utilisations relatif à «Gérer ses propres offres»



**FIGURE 2.7 – Diagramme de cas d'utilisation «Gérer ses propres offres»**

La description de scénario du cas d'utilisation : « Gérer ses propres offres » est donnée par le tableau 2.5 :

Titre	Gérer ses offres.
Acteurs	Utilisateurs.
Objectif	L'utilisateur peut gérer ses offres en accédant à une liste complète des offres créées par lui-même, où il peut consulter les détails de chacune, créer de nouvelles offres, les modifier et les supprimer. Il peut aussi accepter ou refuser un utilisateur et consulter la liste des utilisateurs acceptés.
Pré-condition	Utilisateur authentifié
Post-condition	utilisateur accepté ou offre modifiée

Scénario nominal	<p><b>[S1] : Accepter un utilisateur</b></p> <ol style="list-style-type: none"> <li>1. L'utilisateur choisit une offre parmi ses offres.</li> <li>2. Le système affiche cette offre en détails.</li> <li>3. L'utilisateur clique sur le bouton "Show my appliciers".</li> <li>4. le système affiche la liste des appliquants.</li> <li>5. L'utilisateur clique sur le bouton "Confirm".</li> <li>6. L'utilisateur accepté est enregistré dans une liste.</li> </ol> <p><b>[S2] : Modifier une offre</b></p> <ol style="list-style-type: none"> <li>1. L'utilisateur choisit une offre parmi ses offres et clique sur l'icône du modification.</li> <li>2. Le système affiche une formulaire déjà rempli par les informations de cette offre.</li> <li>3. L'utilisateur change les champs voulus et clique sur le bouton "Update".</li> <li>4. Le système affiche la liste des offres synchronisée.</li> </ol>
Scénario alternatif	<p><b>[S1].a.A1, [S2].a.A1 : La liste des offres est vide.</b></p> <p><b>[S1].d.A1 : la liste est vide.</b></p> <p><b>[S2].c.A1 : Les valeurs entrées ne respectent pas les conditions de saisie</b></p> <ul style="list-style-type: none"> <li>— 1.Le système envoie un message d'erreur.</li> <li>— 2. Le scénario reprend à partir de l'étape b.</li> </ul>
Exigences supplémentaires	<ul style="list-style-type: none"> <li>— Les interfaces doivent être ergonomique.</li> <li>— Les messages d'erreur doivent être compréhensibles et clairs.</li> </ul>

TABLE 2.5 – Description textuelle du cas d'utilisation «Gérer ses propres offres»

## 2.4 Diagrammes de navigation système

Les diagrammes de navigation permettent de visualiser et de planifier la manière dont les utilisateurs interagiront avec l'application ou le site. Ils aident à définir les différentes pages, les fonctionnalités et les flux de navigation disponibles pour les utilisateurs.

### 2.4.1 Diagramme de navigation système partie web

La figure 2.8 montre le diagramme de navigation du système concernant notre partie web. Elle nous montre les transitions entre ces écrans ainsi que les informations sur les actions disponibles à partir de chaque écran.

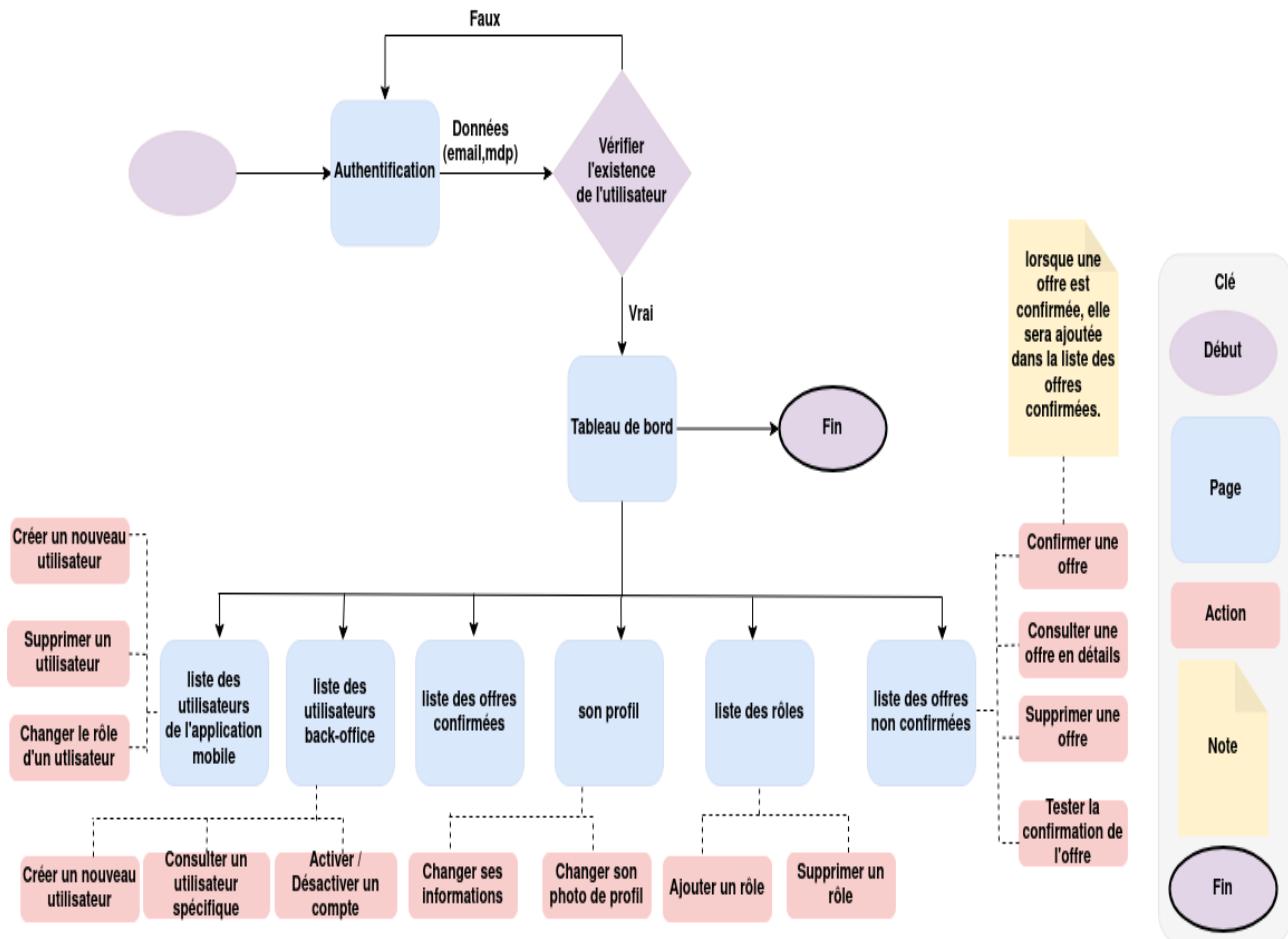


FIGURE 2.8 – Diagramme de navigation système partie web

## 2.4.2 Diagramme de navigation système partie mobile

La figure 2.9 montre le diagramme de navigation du système concernant notre partie mobile. Elle nous montre les transitions entre ces écrans ainsi que les informations sur les actions disponibles à partir de chaque écran.

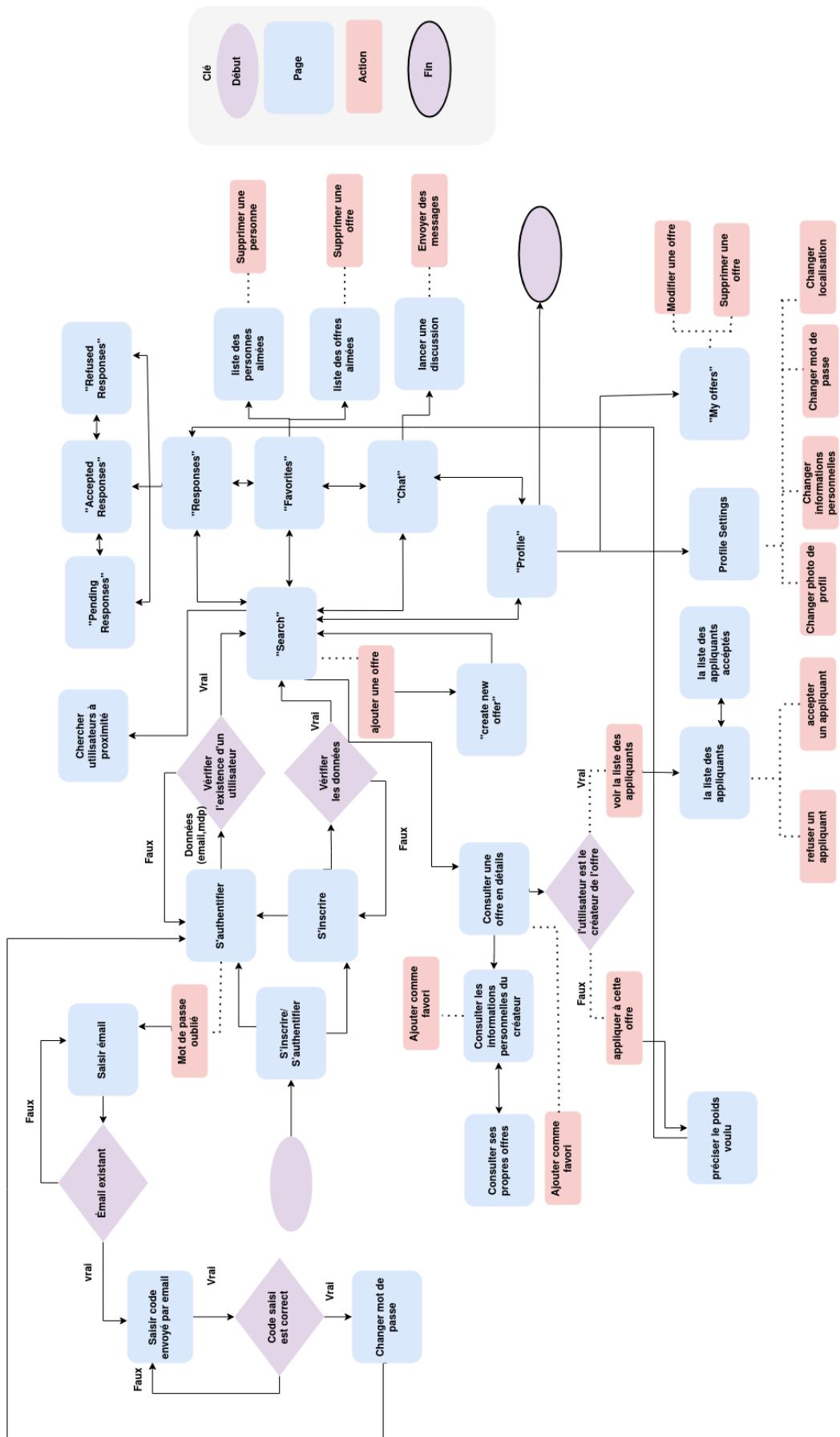


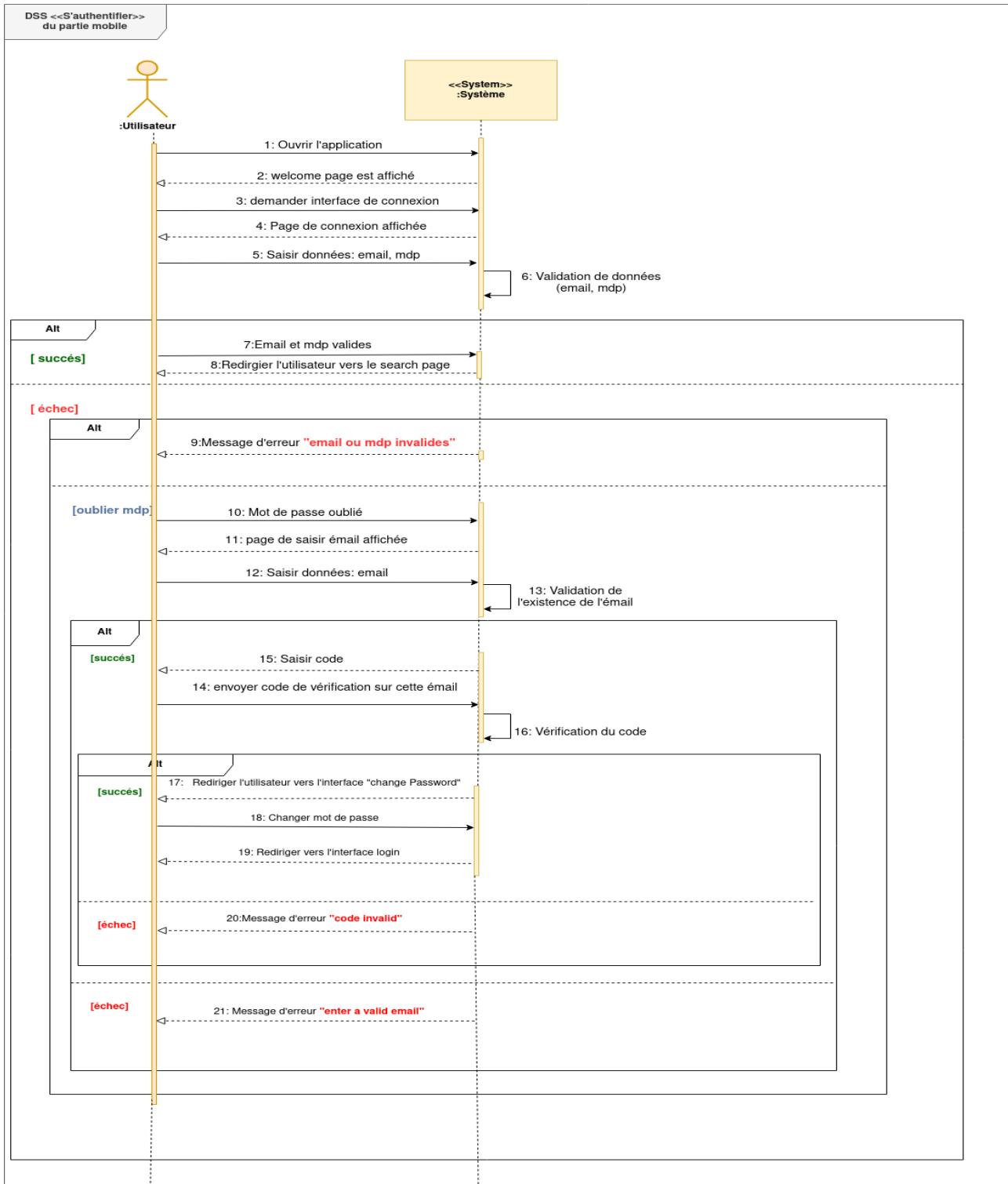
FIGURE 2.9 – Diagramme de navigation système partie mobile

## 2.5 Diagrammes de séquence système

Les diagrammes de séquence système nous aide à mettre l'accent sur les interactions entre notre système et ses différents acteurs. Ils sont utilisés pour modéliser le comportement dynamique d'un système du point de vue de ses interactions avec les acteurs externes.

### 2.5.1 diagramme de séquence système de cas d'utilisation «S'authentifier»

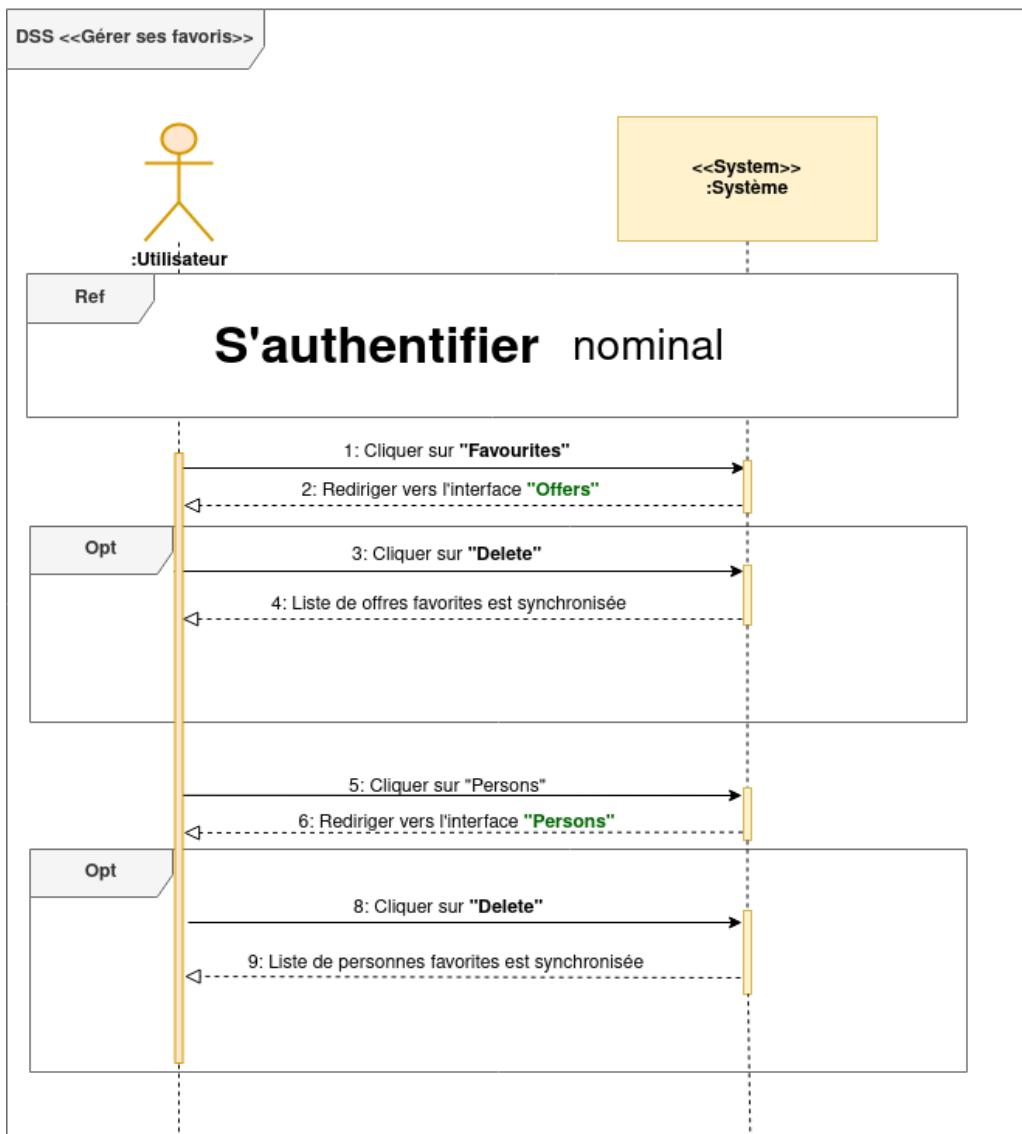
La figure 2.10 montre le diagramme de séquence système concernant le cas d'utilisation «S'authentifier». Il nous dépeint les différentes étapes que suit un utilisateur déjà inscrit pour s'authentifier. Lorsque les informations entrées dans le formulaire d'authentification sont jugées valides, une requête est envoyée au système pour vérifier si l'utilisateur existe déjà.



**FIGURE 2.10 – Diagramme de séquence système de cas d'utilisation «S'authentifier»**

### 2.5.2 Diagramme de séquence système de cas d'utilisation «Gérer ses favoris»

La figure 2.11 montre le diagramme de séquence système concernant le cas d'utilisation «Gérer ses favoris». Il nous montre les différentes étapes que suit un utilisateur déjà authentifié pour gérer la liste de ses favoris.



**FIGURE 2.11** – Diagramme de séquence système de cas d'utilisation «Gérer ses favoris»

### 2.5.3 Diagramme de séquence système de cas d'utilisation «Gérer la liste des offres»

La figure 2.12 montre le diagramme de séquence système concernant le cas d'utilisation «Gérer la liste des offres». il nous illustre les différentes étapes que suit un administrateur déjà authentifié pour gérer la liste des offres.

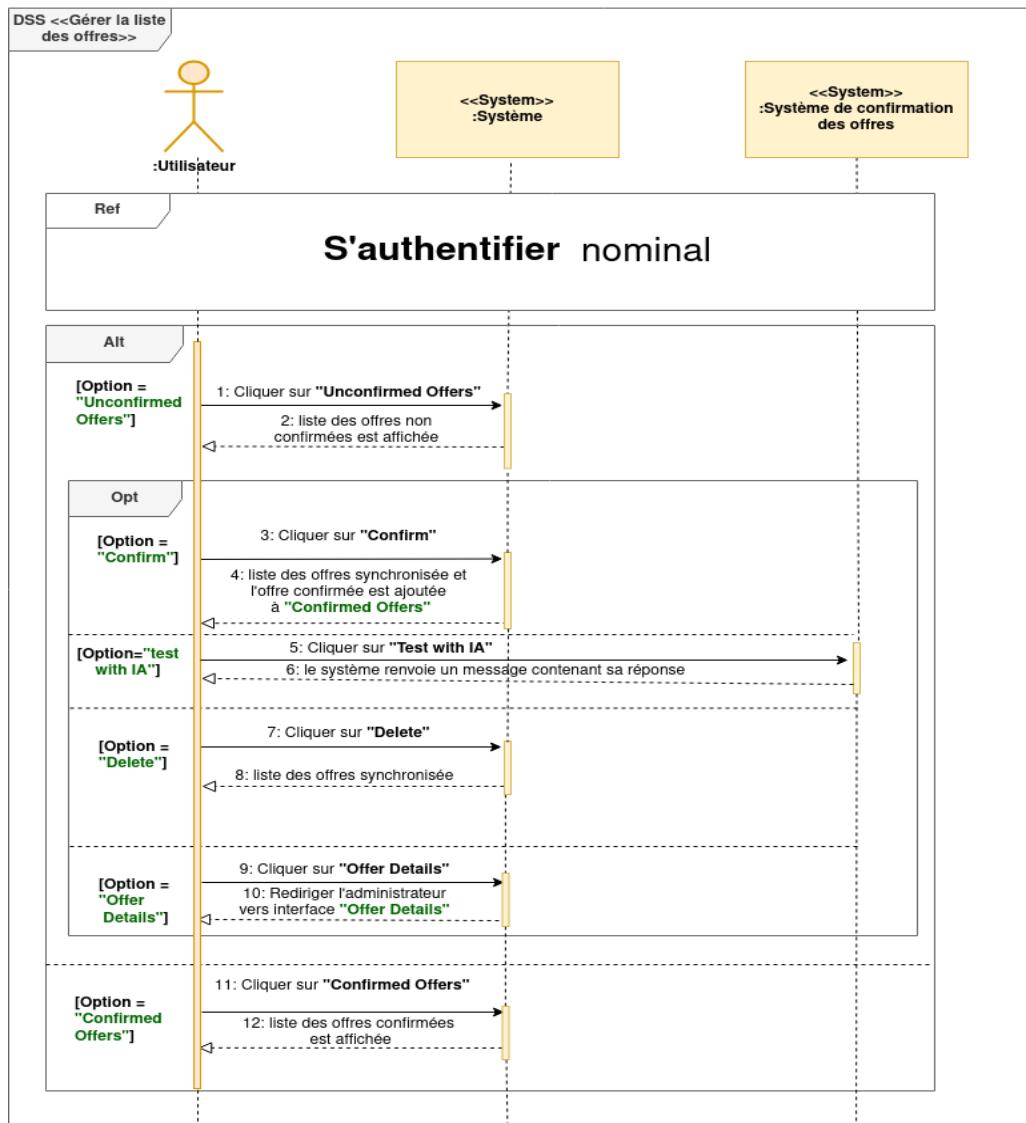


FIGURE 2.12 – Diagramme de séquence système de cas d'utilisation «Gérer la liste des offres»

## 2.5.4 Diagramme de séquence système de cas d'utilisation «Gérer son profil»

La figure 2.13 montre le diagramme de séquence système concernant le cas d'utilisation «Gérer son profil». il nous montre les différentes étapes que suit un utilisateur déjà authentifié pour gérer son profil.

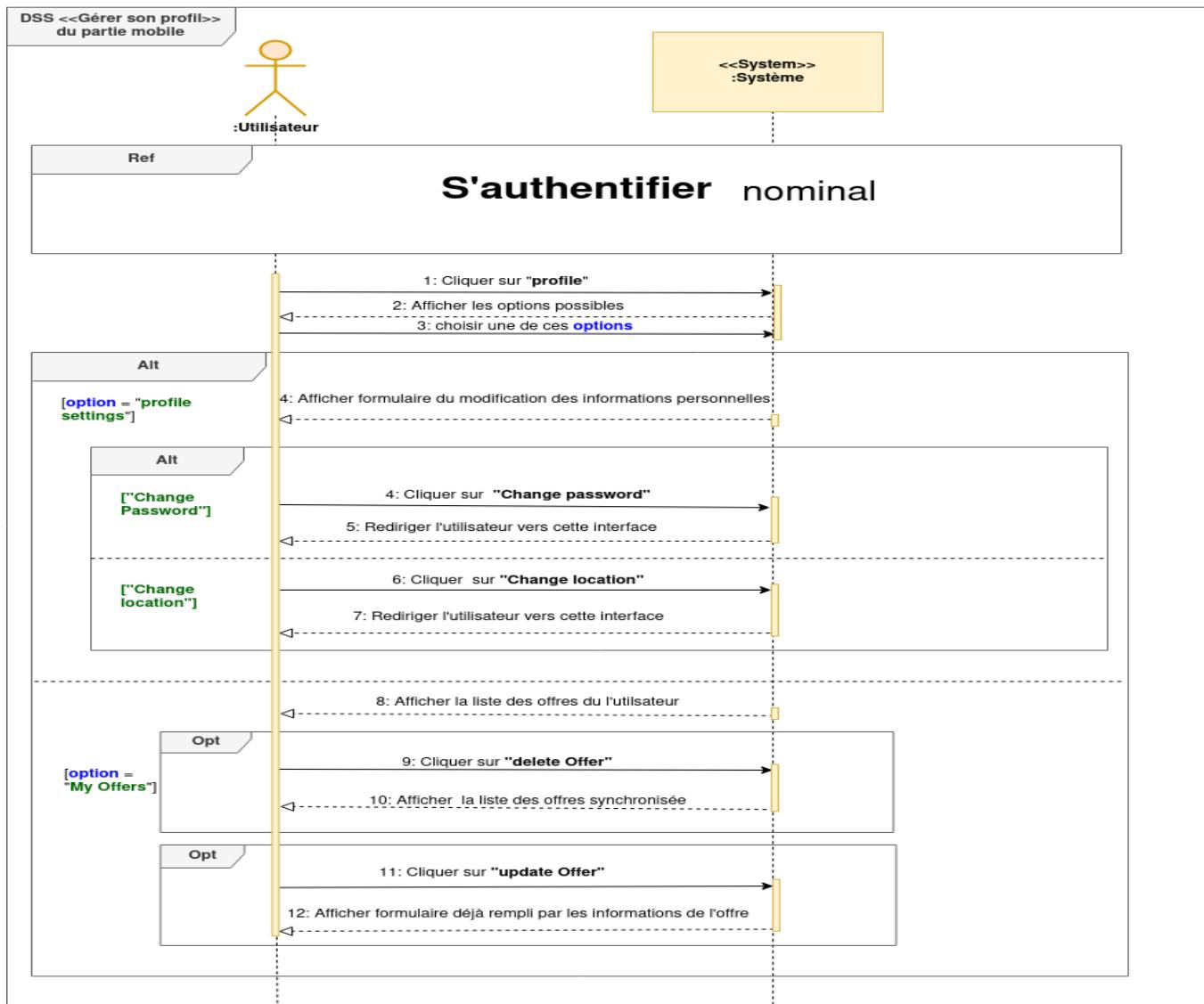


FIGURE 2.13 – Diagramme de séquence système de cas d'utilisation «Gérer son profil»

## Conclusion

Dans ce chapitre, nous avons décrit les phases de spécification des besoins de notre projet. Nous avons commencé par l'identification des différents acteurs. Ensuite, nous avons présenté les besoins fonctionnels et non fonctionnels du projet. Finalement, nous avons détaillé les diagrammes des différents cas d'utilisation, les diagrammes d'activité, les diagrammes de communication ainsi que les diagrammes de séquence système de quelques cas d'utilisation.

# Aperçu Conceptuel

## Introduction

Dans ce chapitre, nous allons poser les bases nécessaires à la mise en œuvre de notre projet. Pour ce faire, nous allons détailler la vision statique de l'architecture, suivie de la vision dynamique du comportement de notre logiciel, en utilisant des motifs d'architecture et des diagrammes UML.

### 3.1 Vue statique de l'application

#### 3.1.1 Les patrons d'architecture

Les patrons d'architecture offrent une approche structurée pour modéliser la structure globale d'un logiciel. Ils fournissent des directives et des principes permettant de concevoir des systèmes logiciels de manière organisée et cohérente, en séparant les préoccupations et en favorisant la réutilisation, la maintenabilité et l'évolutivité du code.

##### 1. Le patron de déploiement : 3-tiers

Ce patron est un modèle logique qui vise à clairement séparer trois couches logicielles au sein d'une même application. Voici une brève description des trois niveaux de cette architecture :

- **Couche présentation** : Ceci concerne la composante apparente de l'application qui permet à l'utilisateur de communiquer avec le système. Elle est responsable de la gestion de l'interface utilisateur, de l'interaction avec l'utilisateur et de l'affichage des données.
- **Couche applicative ou couche métier** : C'est la composante qui renferme la logique métier de l'application. Elle représente le noyau fonctionnel de l'application où les règles de gestion sont définies et où les opérations sur les données sont effectuées en réponse aux requêtes des utilisateurs provenant de la couche de présentation.
- **Couche accès aux données** : Il s'agit de la couche chargée de la gestion des données de l'application. Cette couche est responsable de récupérer les données et de les transmettre à la couche applicative pour être traitées, puis renvoyées à l'utilisateur.

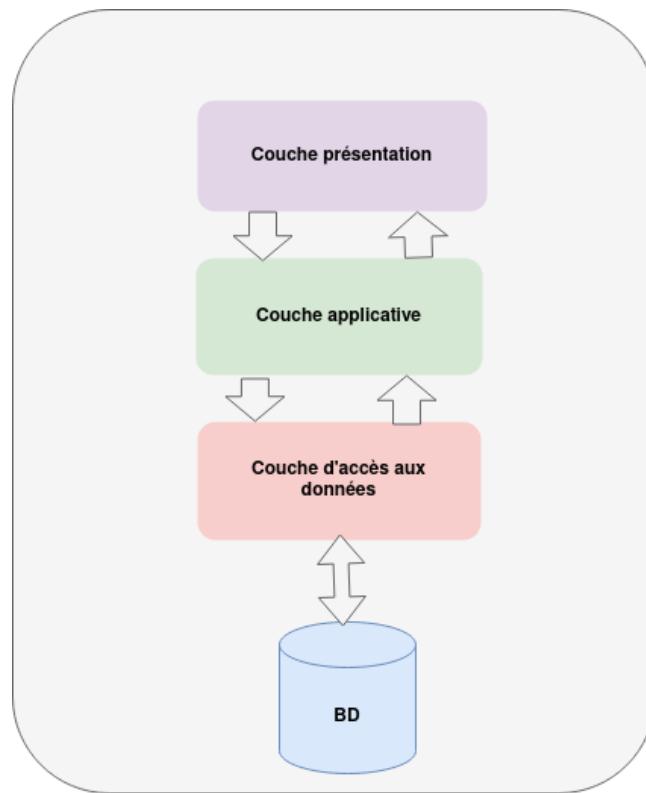


FIGURE 3.1 – Patron d'architecture "3 tiers"

## 2. Le patron interactif : MVC (Modèle-Vue-Contrôleur)

Le patron MVC facilite la structuration efficace du code source. Elle vous guide dans la création des fichiers nécessaires tout en définissant clairement leur rôle respectif. L'objectif fondamental du modèle MVC est de diviser la logique du code en trois parties distinctes, chacune étant représentée par des fichiers séparés :

- **Modèle** : cette partie gère les données de l'application. Il contient les données brutes de l'application. Il représente une abstraction des données et fournit des méthodes pour accéder, manipuler et notifier les changements de données.
- **Vue** : cette partie se concentre sur l'affichage les données provenant du modèle. La vue est responsable de la présentation des informations et de leur mise en forme pour une interaction avec l'utilisateur.
- **Contrôleur** : cette partie gère la logique du code qui prend des décisions. C'est en quelque sorte l'intermédiaire entre le modèle et la vue, gérant les interactions utilisateur, coordonnant les opérations sur les données et mettant à jour l'interface utilisateur en conséquence

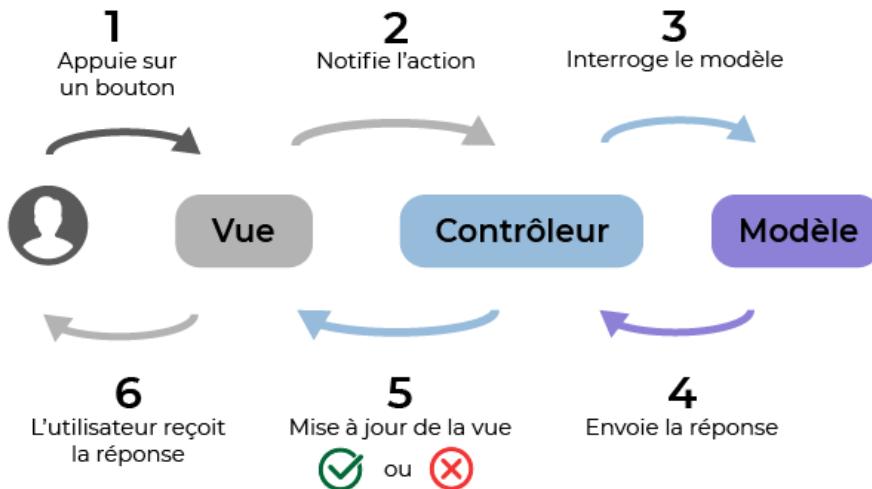


FIGURE 3.2 – Principe du patron d'architecture MVC

### 3.1.2 Les patrons de conception

Contrairement aux patrons d'architecture, les patrons de conception permettent de modéliser une structure partielle locale du système. Les patrons d'architecture forment un niveau d'abstraction (cacher certains détails) plus élevé par rapport aux patrons de conception.

#### 1. Le patron créationnel : Factory

Le factory pattern vise à résoudre un problème fondamental lors de l'instanciation, c'est-à-dire la création d'un objet concret d'une classe, dans la programmation orientée objet : créer un objet directement au sein de la classe, qui a besoin de cet objet ou devrait l'utiliser, est possible en principe, mais très rigide. Il lie la classe à cet objet particulier et rend impossible de modifier l'instanciation indépendamment de la classe. Le factory pattern évite un tel code en définissant d'abord une opération distincte pour la création de l'objet : la fabrique. Une fois appelée, elle génère l'objet, au lieu du constructeur de classe mentionné plus haut.

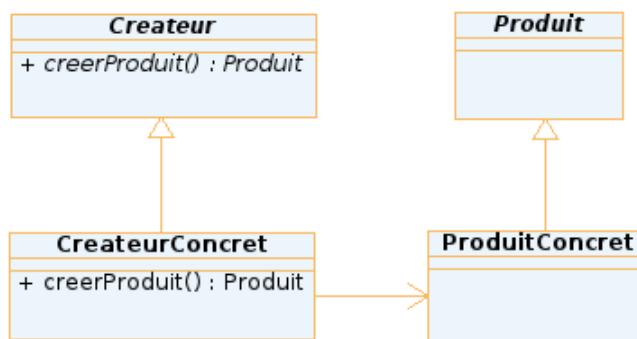
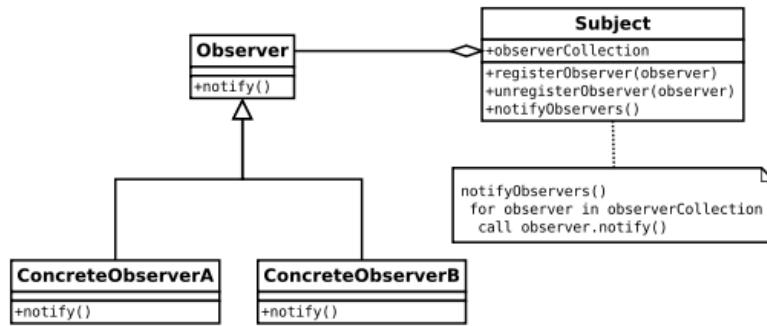


FIGURE 3.3 – Principe du patron de conception Factory

#### 2. Le patron comportemental : Observer

L'Observateur est un patron de conception comportemental qui permet de mettre en place un mécanisme de souscription pour envoyer des notifications à plusieurs objets, au

sujet d'événements concernant les objets qu'ils observent.



**FIGURE 3.4 – Principe du patron de conception Observer**

### 3.1.3 Diagrammes de classes des entités

Un diagramme de classes des entités est un type de diagramme utilisé en génie logiciel pour représenter les entités et les relations entre celles-ci dans un système informatique. Ce diagramme est considéré comme crucial dans la modélisation orientée-objet. Il s'agit d'une représentation statique qui ne prend pas en compte les aspects temporels et dynamiques.

La Figure 3.5 illustre le diagramme de classes des entités de notre projet.

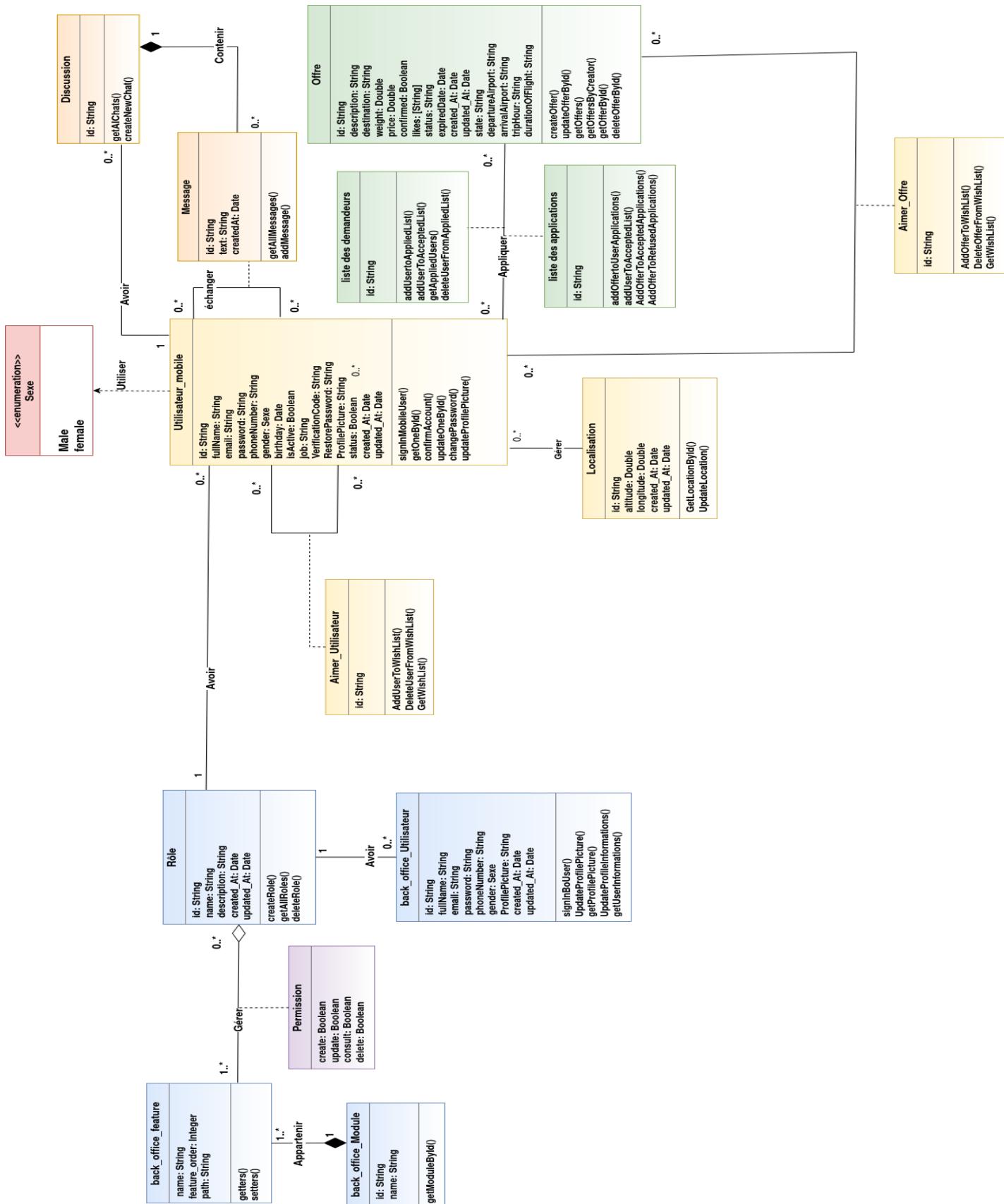


FIGURE 3.5 – Diagramme de classes des entités

Le tableau 3.1 donne une description brève de chacune des classes qui composent notre diagramme.

Classe	Description
Utilisateur Back-office	cette classe représente l'ensemble des administrateurs qui peuvent se connecter sur notre partie back-office.
Utilisateur mobile	cette classe expose l'ensemble des utilisateurs de notre partie mobile.
Rôle	Symbolise les différents rôles dont peut se revêtir un administrateur pour avoir accès à diverses fonctionnalités proposées dans la partie backoffice.
Back-office module	la présente classe expose les différents modules de l'application back-office tels que " Back-office Users" , "Mobile users", ...
Back-office feature	cette classe présente les fonctionnalités que l'application est capable de réaliser afin de répondre aux besoins de l'utilisateur.
Offre	la présente classe expose les différents informations concernant l'offre créée.
Localisation	cette classe modélise les informations relatives à une adresse d'un utilisateur de la partie mobile.
Discussion	Modélise les informations relatives à tous les discussions dont l'utilisateur mobile est inclu.
Message	Modélise les informations relatives à un message échangé entre deux utilisateurs.

**TABLE 3.1 – Description des classes**

## 3.2 Vue dynamique de l'application

### 3.2.1 Diagrammes de séquence détaillé

Les diagrammes de séquence sont une solution populaire de modélisation dynamique en langage UML, ils se concentrent plus précisément sur les lignes de vie, les processus et les objets qui vivent simultanément, et les messages qu'ils échangent entre eux pour exercer une fonction avant la fin de la ligne de vie.

Au niveau de cette partie nous présenterons les différents diagrammes de séquences détaillés et pour ceci nous sélectionnons les fonctionnalités les plus importantes existantes dans notre système d'application.

- **Diagramme de séquence «Créer un nouveau utilisateur» :**

Un utilisateur déjà authentifié et a la permission "create" dans son rôle peut remplir la formulaire du "create back-office user" par des champs valides en lui attribuant un rôle spécifique afin d'avoir un nouveau utilisateur back office ajouté dans notre base de données.

La figure 3.6 illustre le diagramme de séquence du cas d'utilisation «Créer un nouveau utilisateur»

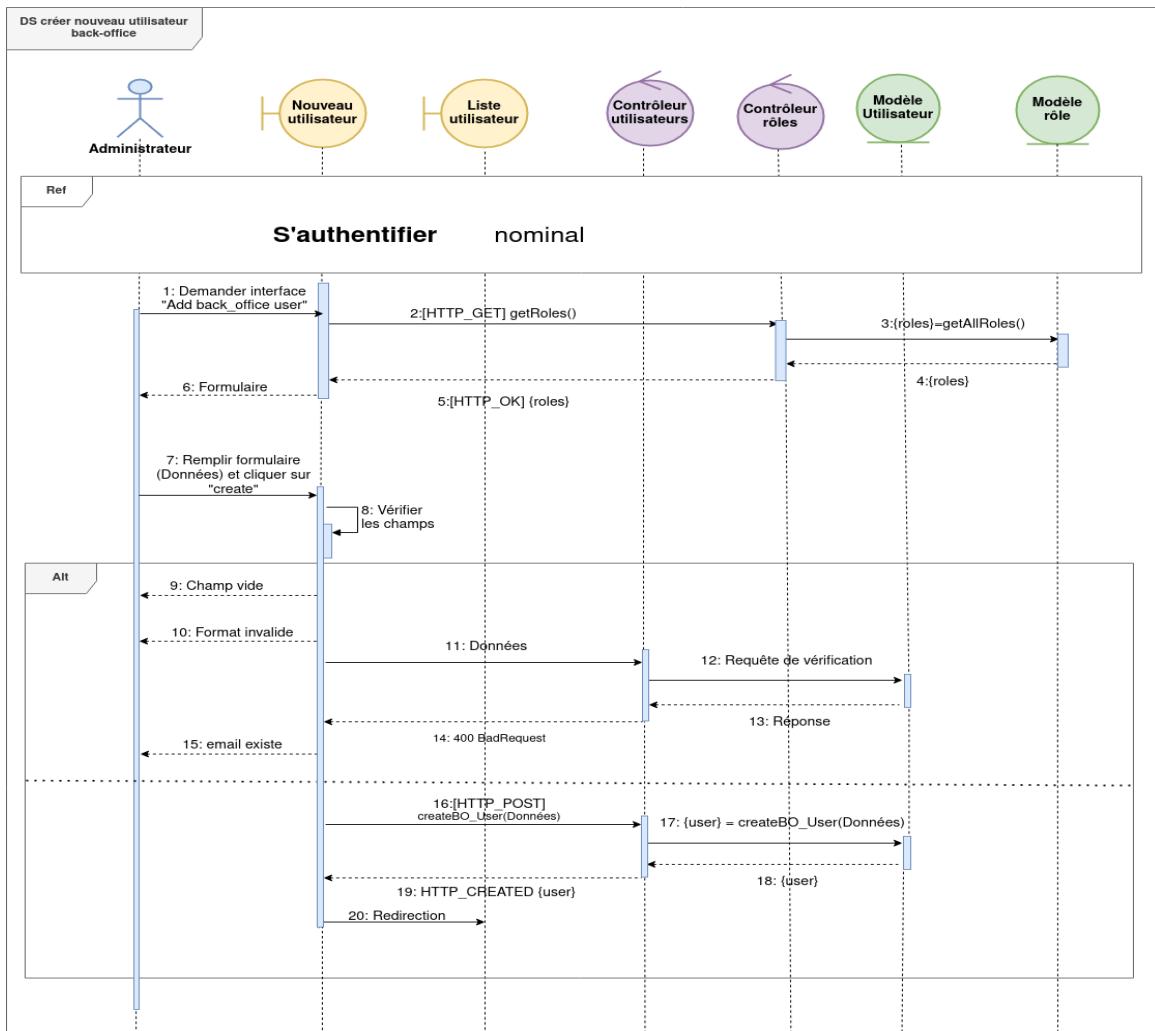
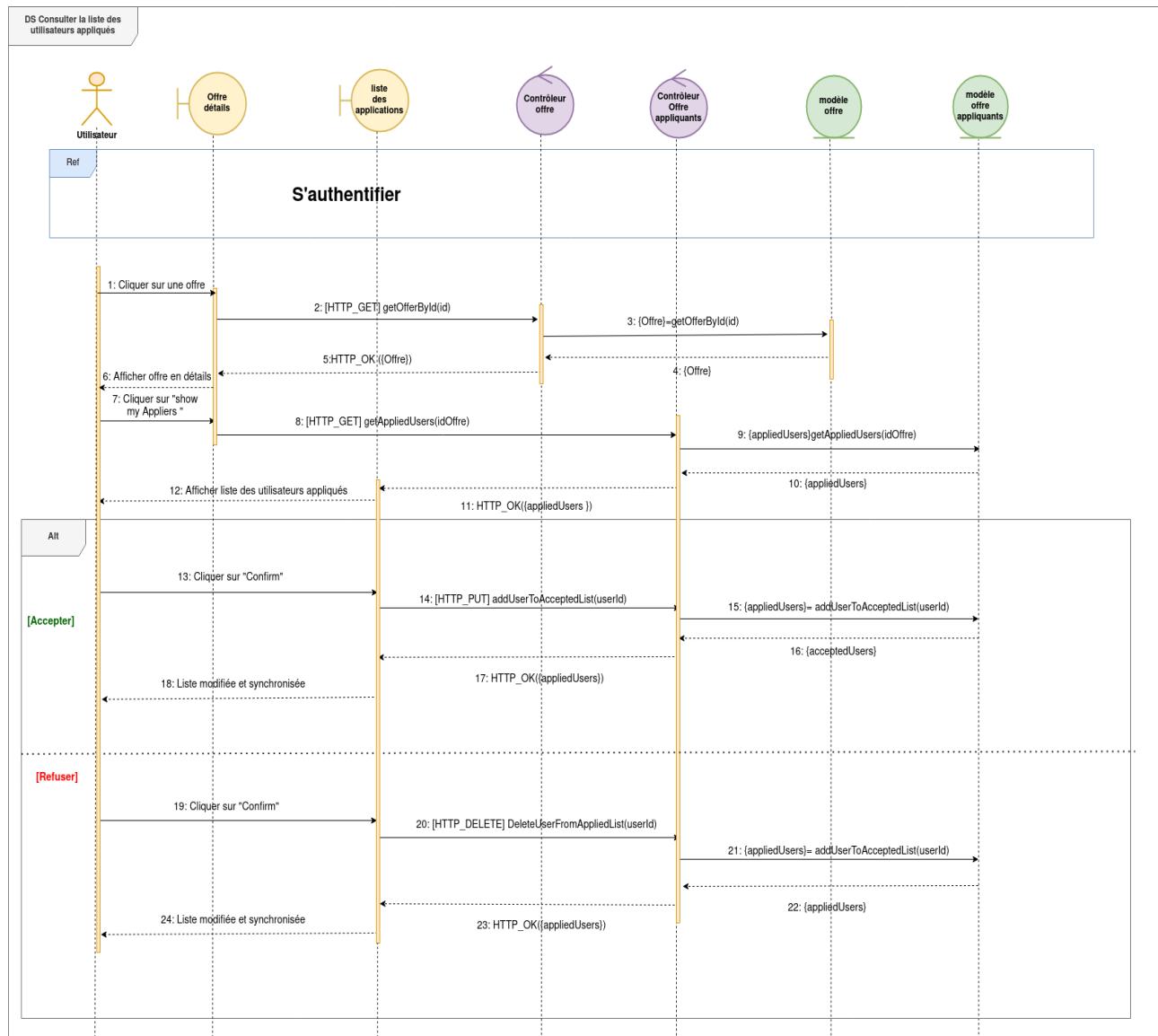


FIGURE 3.6 – Diagramme de séquence «Créer un nouveau utilisateur»

#### — Diagramme de séquence «Consulter la liste des demandeurs» :

Lorsqu'un utilisateur est déjà authentifié et qu'il est le créateur d'une offre, il dispose de priviléges spécifiques pour gérer les candidatures. Tout d'abord, il a le droit de consulter la liste des utilisateurs ayant postulé à son offre, ce qui lui permet d'avoir une vue d'ensemble des candidats intéressés. De plus, il peut accéder aux profils de ces utilisateurs afin de consulter leurs informations personnelles. Cette possibilité lui permet d'obtenir un aperçu détaillé de chaque candidat, facilitant ainsi le processus de sélection. Ensuite, en tant que créateur de l'offre, il est en mesure d'accepter un utilisateur spécifique parmi ceux qui ont postulé. Ce pouvoir de décision lui permet de choisir la personne la mieux adaptée à ses besoins et critères. Enfin, l'utilisateur créateur de l'offre a également la faculté de refuser une candidature si celle-ci ne correspond pas à ses attentes ou exigences. La figure 3.7 illustre le diagramme de séquence du cas d'utilisation «Consulter la liste des demandeurs»

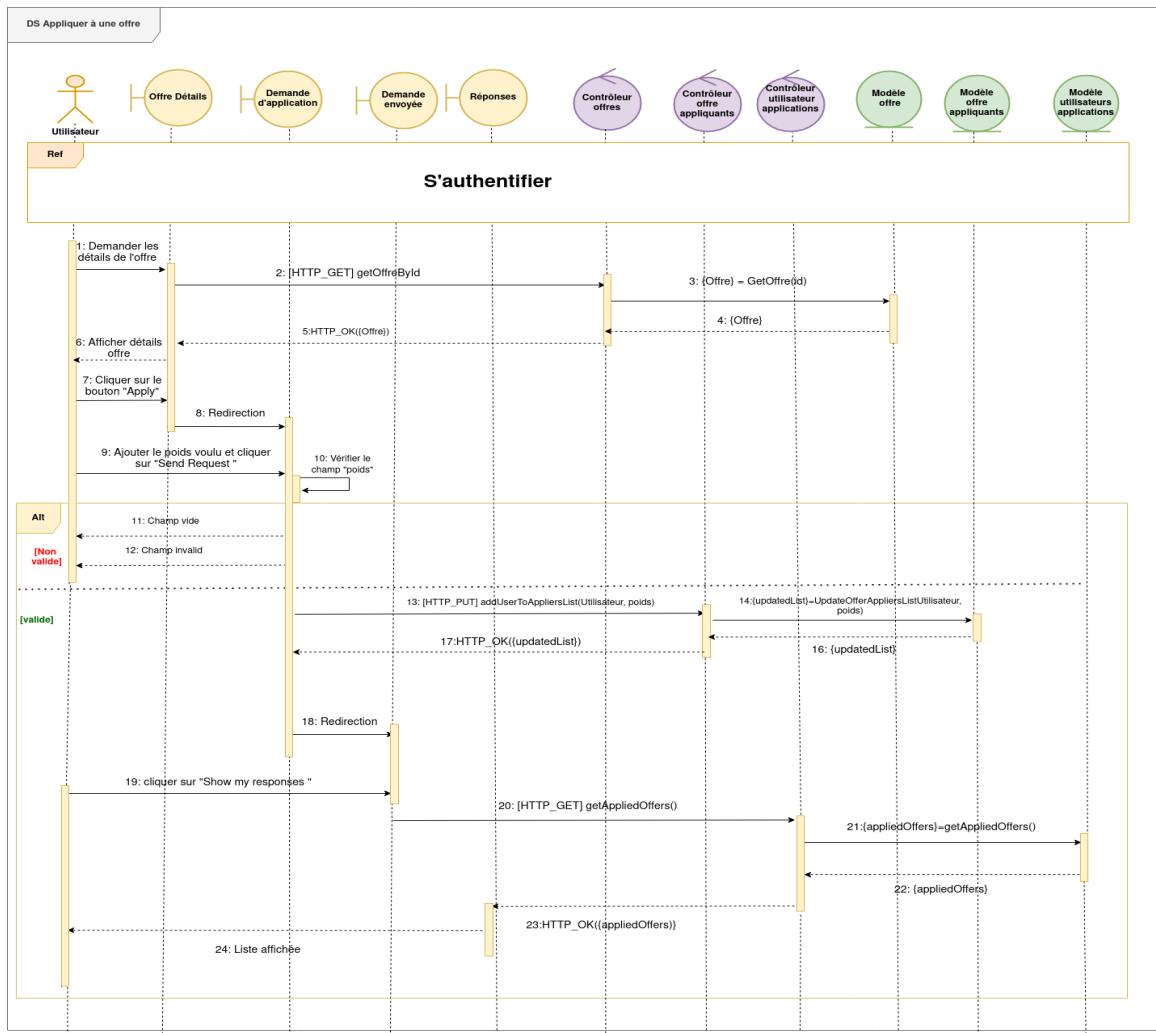


**FIGURE 3.7 – Diagramme de séquence «Consulter la liste des demandeurs»**

#### — Diagramme de séquence «Postuler à une offre» :

Un utilisateur déjà authentifié peut consulter les différentes offres valables afin de choisir une pour y appliquer. Pendant la procédure d'application l'utilisateur doit déclarer le poids voulu afin d'être ajouter comme étant demandeur de cette offre.

La figure 3.8 présente le diagramme de séquence du cas d'utilisation «Postuler à une offre».



**FIGURE 3.8 – Diagramme de séquence «Postuler à une offre»**

#### — Diagramme de séquence «Lancer une discussion» :

Une fois authentifié, un utilisateur peut explorer les profils d'autres utilisateurs et initier des discussions instantanées pour échanger des messages en temps réel. Cette fonctionnalité facilite les interactions directes et la communication en temps réel entre les utilisateurs.

La figure 3.9 montre le diagramme de séquence du cas d'utilisation «Lancer une discussion».

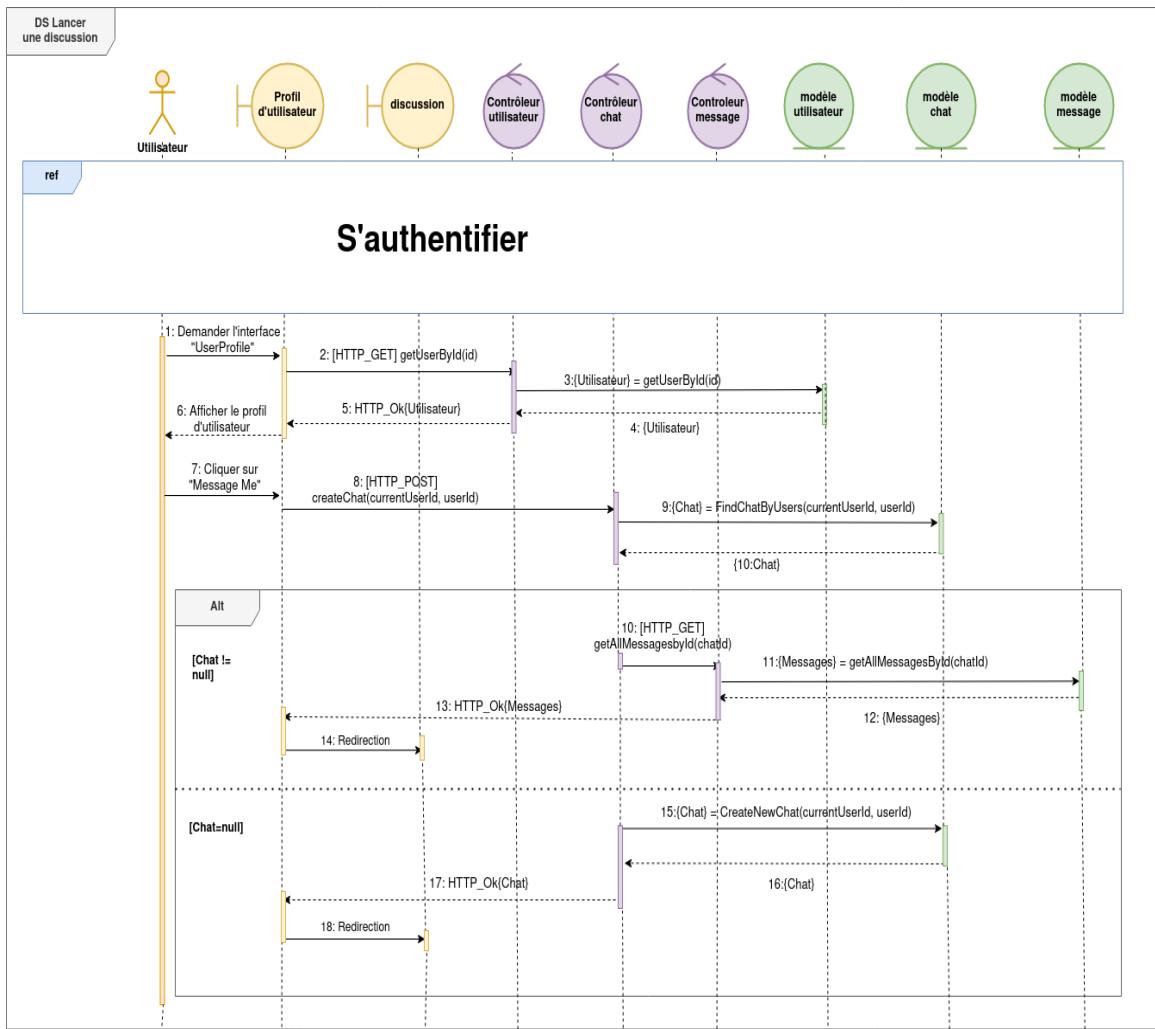
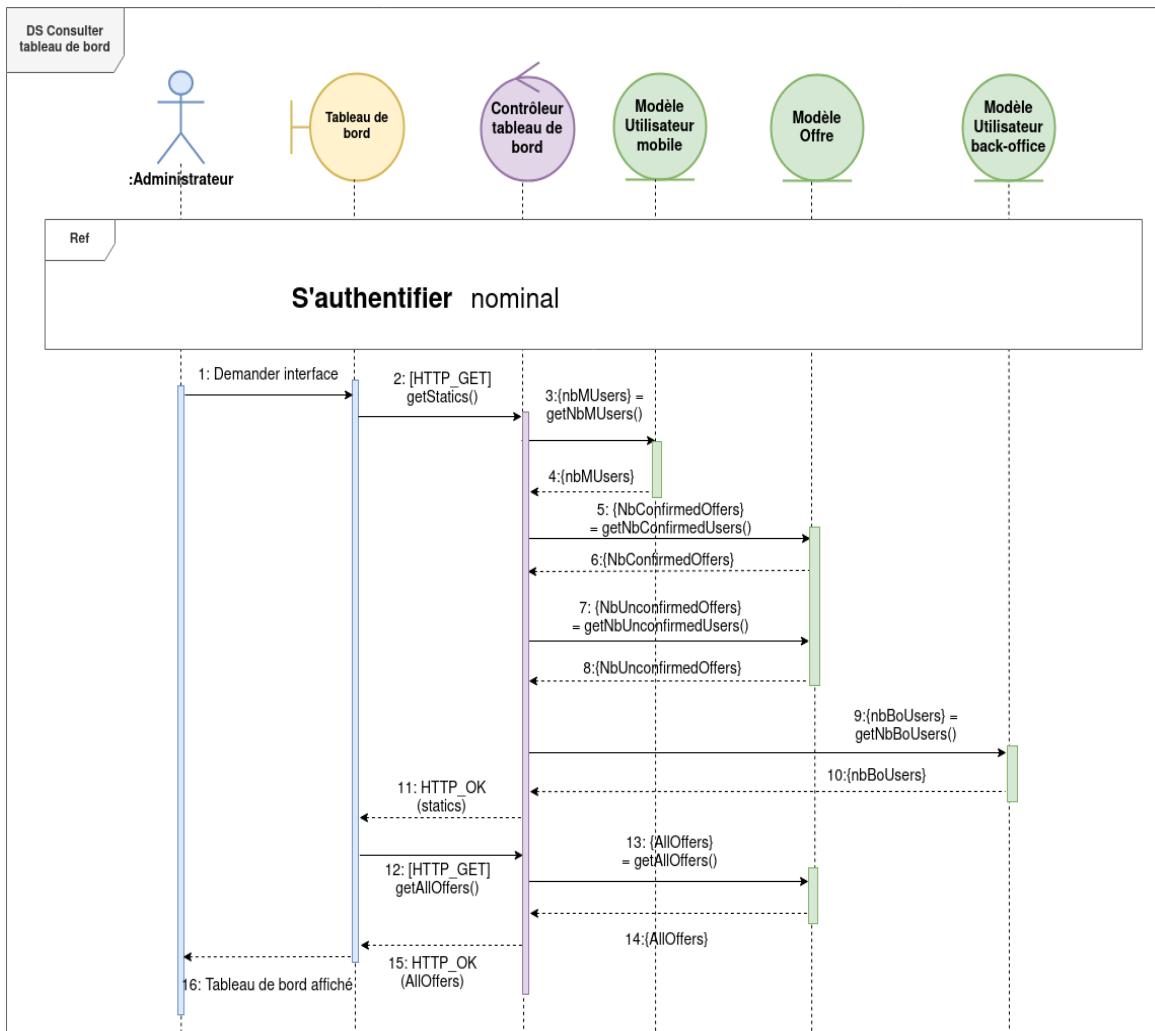


FIGURE 3.9 – Diagramme de séquence «Lancer une discussion»

#### — Diagramme de séquence «Consulter tableau de bord» :

Lorsqu'un administrateur est déjà authentifié, il a la possibilité de consulter diverses statistiques qui sont affichées dans son tableau de bord. Ce tableau de bord constitue un outil essentiel qui lui permet d'avoir une vue d'ensemble des informations clés et de prendre des décisions éclairées. Les statistiques disponibles peuvent inclure des données telles que le nombre total d'utilisateurs enregistrés, des offres confirmées, les offres non confirmées ou tout autre paramètre pertinent pour l'activité de l'administrateur.

La figure 3.10 montre le diagramme de séquence du cas d'utilisation «Postuler à une offre».



**FIGURE 3.10 – Diagramme de séquence «Consulter tableau de bord»**

### 3.2.2 Diagrammes d'activités

Le diagramme d'activités participe à la description du comportement d'un système et sa modélisation dynamique. La description du comportement peut concerner une méthode d'une classe ou les scénarios d'un ou plusieurs cas d'utilisation.

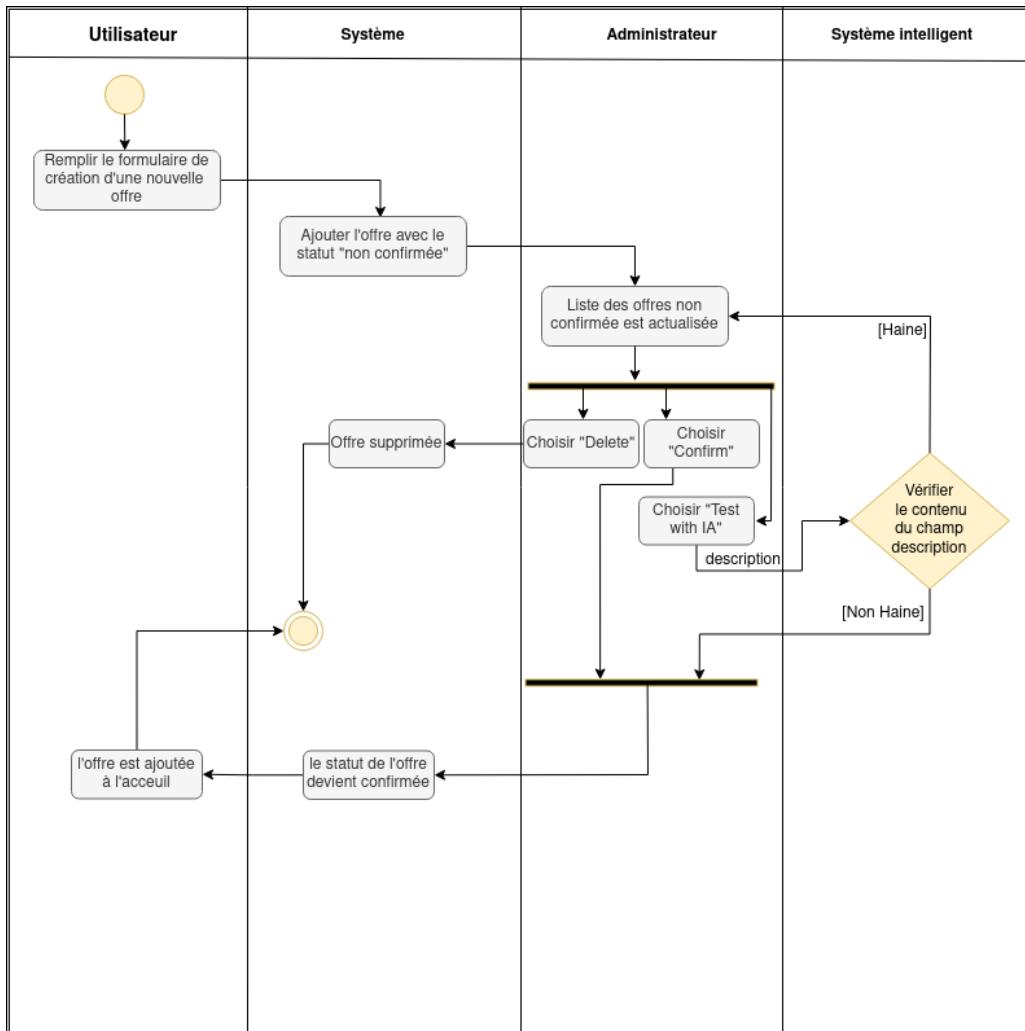
- **Diagramme d'activité du cas d'utilisations «Ajouter une nouvelle offre» et «Gérer les offres non confirmées»**

Une fois que l'utilisateur a rempli le formulaire de création d'une nouvelle offre, celle-ci est ajoutée avec le statut "non confirmé". L'administrateur dispose alors de la possibilité de consulter la liste des offres non confirmées afin de les gérer. Il peut effectuer plusieurs actions sur ces offres, telles que les accepter, les refuser ou même les soumettre à un test de confirmation en utilisant le module d'intelligence artificielle. En consultant la liste des offres non confirmées, l'administrateur peut prendre connaissance des nouvelles propositions et les évaluer en fonction des critères établis. Il peut accepter une offre si elle répond aux exigences et aux normes requises, permettant ainsi à l'offre de passer au statut "confirmé" et d'être rendue publique. D'un autre côté, l'administrateur a également le pouvoir de refuser une offre s'il estime qu'elle ne correspond pas aux attentes ou aux critères définis.

De plus, l'administrateur peut choisir d'utiliser le module d'IA pour tester une offre

spécifique. actions sur ces offres, telles que les accepter, les refuser ou même les soumettre à un testCela implique l'application de techniques d'apprentissage automatique ou d'autres méthodes d'analyse pour évaluer les performances de l'offre, la pertinence des informations fournies ou encore pour estimer sa viabilité.

La Figure 3.11 illustre le diagramme d'états-transitions du cas d'utilisation « Gérer ses offres » et présente les différents états que la classe utilisateur peut traverser.



**FIGURE 3.11** – Diagramme du cas d'utilisations «Ajouter une nouvelle offre» et «Gérer les offres non confirmées»

### 3.2.3 Diagrammes états-transitions

Un diagramme d'états-transitions est un type de diagramme comportemental en langage de modélisation unifié (UML) qui représente les transitions entre divers objets. Un automate désigne tout appareil qui enregistre l'état d'un objet à un moment donné et peut changer l'état ou provoquer d'autres actions selon les informations qu'il reçoit. Les états correspondent aux différentes combinaisons d'informations qu'un objet peut contenir et non la façon dont celui-ci se comporte.

#### — Diagramme d'états-transitions du cas d'utilisation «Gérer ses offres»

La Figure 3.12 illustre le diagramme d'états-transitions du cas d'utilisation « Gérer ses offres » et présente les différents états que la classe offre peut traverser.

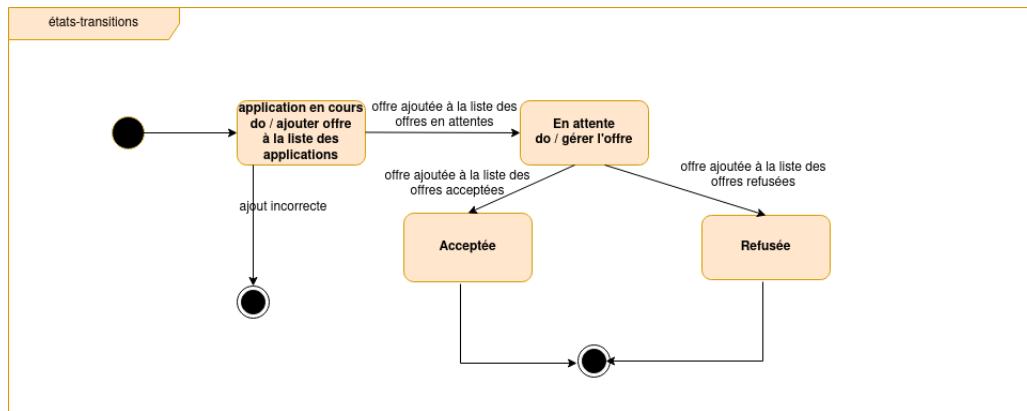


FIGURE 3.12 – Diagramme d'états-transitions «Gérer ses offres»

#### — Diagramme d'états-transitions du cas d'utilisation «Gérer des offres»

La Figure 3.13 illustre le diagramme d'états-transitions du cas d'utilisation « Gérer des offres » et présente les différents états que la classe utilisateur peut traverser.

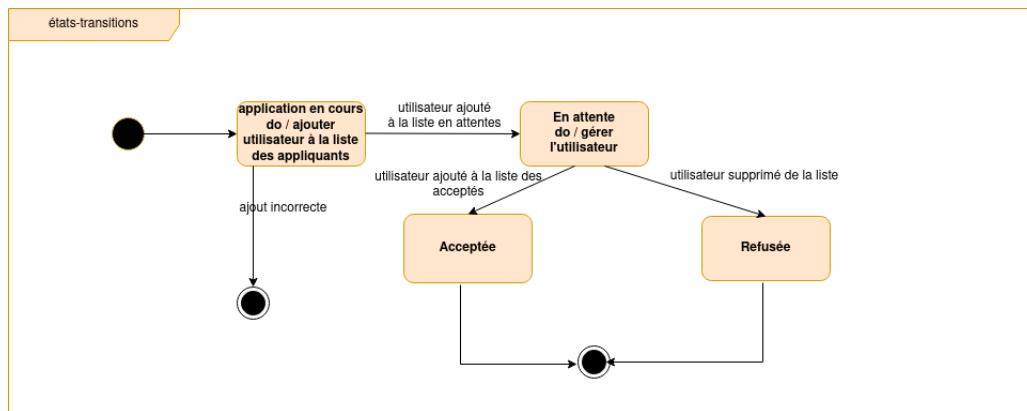


FIGURE 3.13 – Diagramme d'états-transitions «Gérer des offres»

## Conclusion

Dans ce chapitre, nous avons présenté à la fois l'architecture globale et détaillée de notre logiciel. Nous avons décrit la vue dynamique du système en utilisant différents types de diagrammes UML (diagrammes de séquences détaillé, diagramme d'activité et diagrammes d'états-transitions).

# Chapitre 4

## Réalisation

### Introduction

Nous passons lors de ce dernier chapitre à la réalisation de la solution proposée. Dans un premier lieu, nous allons présenter les technologies et les outils d'implémentation utilisés pour développer notre logiciel. Dans un second lieu, nous allons décrire l'architecture physique du logiciel. Enfin, nous allons illustrer le travail réalisé par des captures d'écran.

#### 4.1 Diagramme de déploiement

Le diagramme de déploiement, qui fait partie des diagrammes structurels, est utilisé pour décrire l'architecture physique des ressources matérielles d'un système. Il permet de mettre en évidence les relations entre les différents composants du système et de visualiser la disposition des nœuds.

Dans notre solution, les utilisateurs ont la possibilité d'accéder à notre système à travers une interface web ou une application mobile. ces deux parties sont connectées au même serveur. Ce serveur assure la communication avec la base de données.

La Figure [4.1](#) illustre le diagramme de déploiement de notre solution.

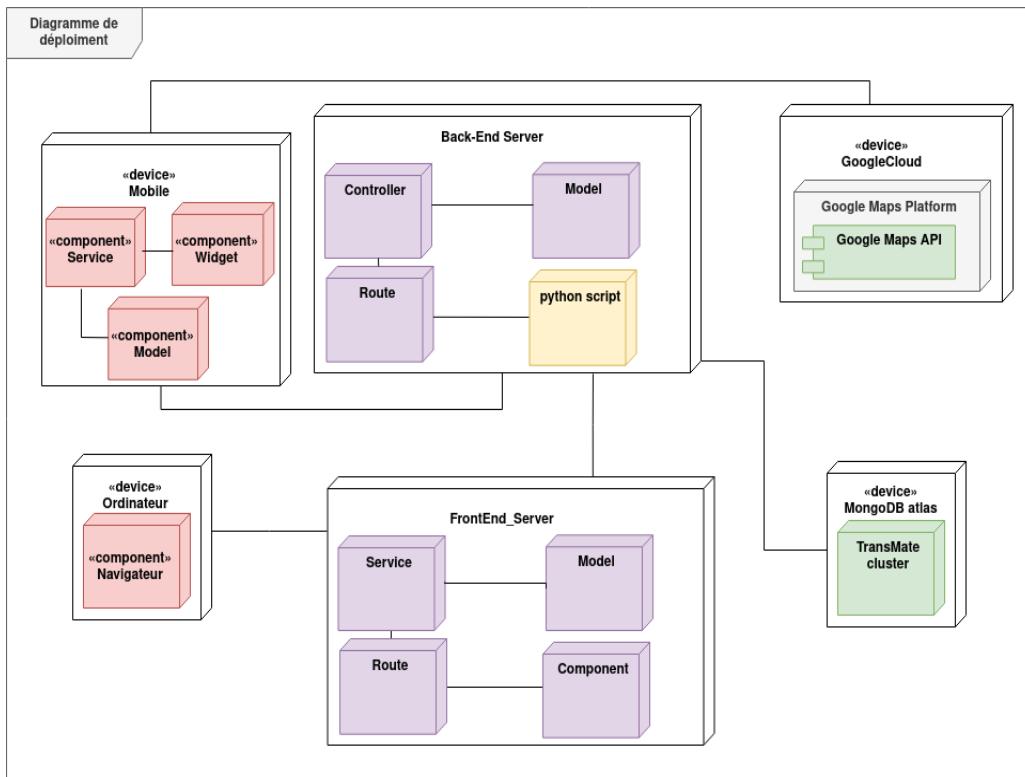


FIGURE 4.1 – Diagramme de déploiement

## 4.2 Environnement de développement

Tout au long de cette partie, nous allons présenter l'environnement matériel mis à la disposition du présent projet ainsi que l'environnement logiciel qui a permis l'aboutissement de la mise en œuvre de l'application.

### 4.2.1 Environnement matériel

Durant les différentes phases du stage, nous avons utilisé une machine DELL embarquant un système d'exploitation Linux et un émulateur virtuel ayant tout les deux les caractéristiques précises dans le tableau 4.1 :

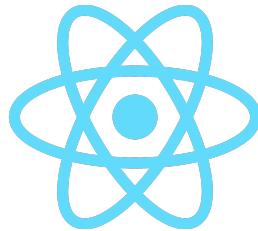
Machine	Ordinateur Portable	Émulateur Virtuel
Marque	DELL	Pixel 2 API TiramisuPrivacy-Sandbox
Système d'exploitation	Linux Ubuntu 20.04.4 LTS	android 11.0×86
Processeur	Intel Core i7-6700HQ (3.5GHz)	Android SDK platform 30(R)
Mémoire RAM	16 GO	1536 MB
Disque dur	1 TO HDD + 124 GO SSD	15 GO

TABLE 4.1 – Environnement matériel

### 4.2.2 Technologies de programmation

Après avoir précisé les outils matériels utilisés pour le développement, nous citons à cet égard les outils logiciels auxquels nous avons eu recours pour développer notre application.

- **React[1]** : c'est une bibliothèque JavaScript développée par Facebook en 2013, permet de créer des interfaces utilisateurs. Elle comporte un nombre très riche de fonctionnalités intéressantes (Routage, requêtage HTTP, gestion des formulaires, etc...). Cette bibliothèque possède un type d'architecture structurée permettant non seulement un développement rapide mais aussi une réalisation d'applications de qualité, faciles à maintenir et à faire évoluer. Nous avons utilisé React pour développer notre application web côté front-end.



*Logo ReactJs*

- **Flutter[2]** : La réalisation de notre application mobile a été développée grâce à la technologie Flutter. C'est un framework open source développé par Google en 2018 pour créer de magnifiques applications multiplateforme avec un seul langage de programmation en permettant aux développeurs d'exploiter les fonctionnalités natives des appareils.



*Logo Flutter*

- **NodeJs[3]** : Node.js est un environnement d'exécution JavaScript open-source basé sur le moteur JavaScript V8 de Google. Il permet d'exécuter du code JavaScript côté serveur. Node.js est conçu pour être performant et évolutif, offrant un modèle asynchrone et orienté événements qui le rend adapté à la création d'applications réseau et de serveurs web.



*Logo NodeJs*

- **ExpressJs[4]** : C'est un framework d'application web minimaliste et flexible pour Node.js qui offre un ensemble robuste de fonctionnalités pour les applications web et mobiles.

Nous avons utilisé ExpressJs pour développer la partie back-end de notre application web et mobile.



*Logo ExpressJS*

- **Socket.io[5]** : C'est une bibliothèque qui permet une communication bidirectionnelle et basée sur les événements avec une latence réduite entre un client et un serveur. Nous avons utilisé Socket.io pour développer l'échange des messages en temps réel.



*Logo Socket.io*

#### 4.2.3 Langages de programmation

- **Dart[6]** : Dart est un langage orienté objet optimisé pour le développement d'applications rapides sur n'importe quelle plateforme. Son objectif est de proposer le langage de programmation le plus productif pour le développement multiplateforme, associé à une plateforme d'exécution flexible pour les frameworks d'applications.



*Logo Dart*

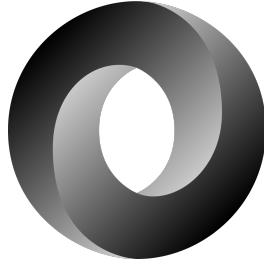
- **Python[7]** : C'est un langage de programmation facile à apprendre et puissant. Il possède des structures de données de haut niveau efficaces et une approche simple mais efficace de la programmation orientée objet. La syntaxe élégante et le typage dynamique de Python, associés à sa nature interprétée, en font un langage idéal pour le scripting et le développement rapide d'applications dans de nombreux domaines sur la plupart des plateformes. Nous avons python pour développer notre module intelligent.



*Logo Python*

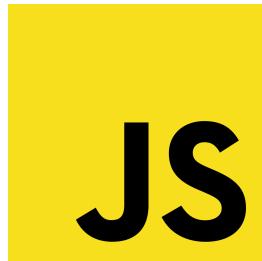
- **JSON[8]** : (JavaScript Object Notation) est un format d'échange de données léger. Il est facile à lire et à écrire pour les humains, et facile à analyser et à générer pour les machines. JSON est un format de texte totalement indépendant du langage, mais utilise

des conventions familières aux programmeurs des langages de la famille C, tels que C, C++, Java, JavaScript, Python, et bien d'autres. Ces caractéristiques font de JSON un langage idéal pour l'échange de données.



*Logo Json*

- **JavaScript[9]** : JavaScript est un langage de programmation dont la date de sa première apparition remonte à 1995. Il est utilisé pour le développement des applications coté client ainsi que les applications coté serveur. Il occupe à présent la première place des langages de programmation les plus utilisés.



*Logo JavaScript*

#### 4.2.4 Base de données

- **MongoDB[10]** : La gestion des données de nos applications est réalisée par MongoDB Atlas. C'est un programme de base de données orienté document, multiplateforme et disponible en tant que source. Classifié comme un programme de base de données NoSQL, MongoDB utilise des documents de type JSON avec des schémas optionnels.



*Logo MongoDB*

#### 4.2.5 Outils

Nous citons à cet égard les outils logiciels auxquels nous avons eu recours pour développer notre solution.

- **Visual studio code[11]** : Il s'agit d'un éditeur de code extensible gratuit et open-source développé par Microsoft. Nous avons utilisé ce logiciel lors du développement côté front-end de notre application web et côté back-end de nos applications web et mobiles.



*Logo visual studio code*

- **Postman[12]** : Postman est une plateforme dédiée aux API qui facilite la création et l'utilisation d'API. En simplifiant chaque étape du cycle de vie de l'API et en favorisant la collaboration, Postman vous permet de créer des API de meilleure qualité plus rapidement. Nous avons employé ce logiciel pour tester nos APIs.



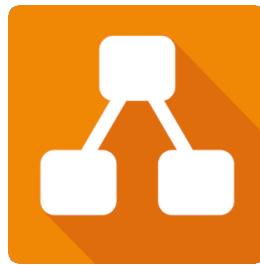
*Logo postman*

- **Android studio[13]** : est un environnement de développement pour développer des applications mobiles Android. Il est basé sur IntelliJ IDEA et utilise le moteur de production Gradle. Il peut être téléchargé sous les systèmes d'exploitation Windows, macOS, Chrome OS et Linux. Nous avons utilisé ce logiciel pour développer notre partie mobile côté front-end.



*Logo Android Studio*

- **Draw.io[14]** : Draw.io est un logiciel libre de création de diagrammes en ligne gratuit permettant de réaliser des flowcharts, des diagrammes de processus, organigrammes, UML et diagrammes de réseau. Nous avons employé Draw.io pour créer nos diagrammes UML.

*Logo Draw.io*

- **Overleaf[15]** : Overleaf est un éditeur LaTeX en ligne qui permet la collaboration en temps réel, le contrôle de version et l'utilisation de modèles LaTeX : un système de composition de haute qualité, il comprend des fonctionnalités conçues pour la production de documents techniques et scientifiques. Nous avons utiliser overleaf pour rédiger notre rapport.

*Logo Overleaf*

- **GitLab[16]** : Il s'agit d'un service d'hébergement de dépôts Git en ligne qui propose un contrôle de version et une gestion de dépôts basés sur Git, ainsi que des fonctionnalités spécifiques telles que la revue de code et la gestion des problèmes. Nous avons utilisé GitLab pour héberger les différents incrément de notre solution.

*Logo GitLab*

## 4.3 Réalisation des incréments

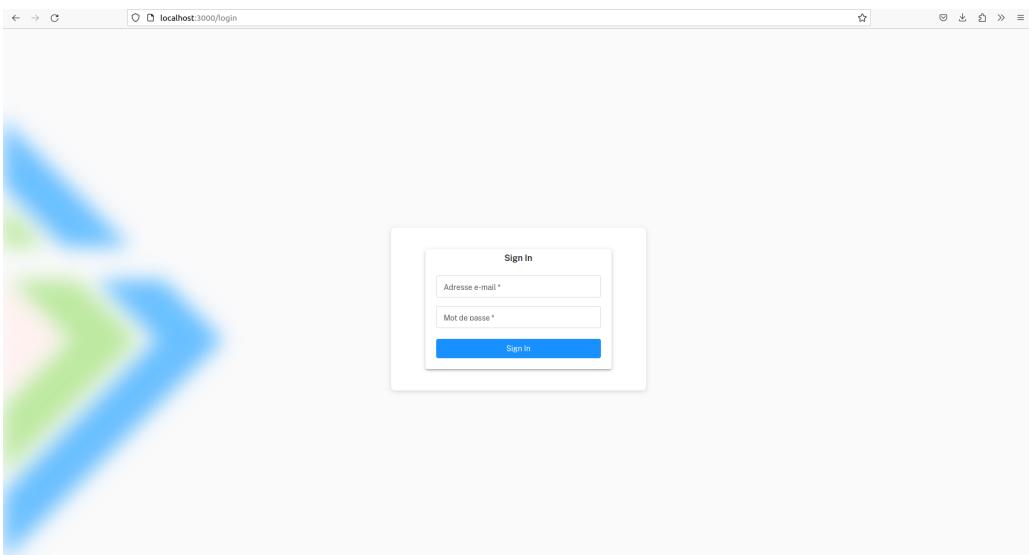
Dans cette section, nous nous concentrons sur la phase de réalisation de notre projet. Nous présentons les scénarios d'exécution en fournissant un aperçu détaillé de la mise en œuvre des fonctionnalités, ainsi que des interfaces principales de notre application pour chaque itération.

### 4.3.1 Réalisation de l'incrément "Partie Web"

- **Authentification pour l'administrateur** :

Avant de pouvoir accéder à la partie web, chaque administrateur doit obligatoirement saisir son adresse e-mail et son mot de passe via l'interface d'authentification. Notre système effectue une vérification pour s'assurer que les informations fournies correspondent à celles d'un administrateur enregistré dans la base de données.

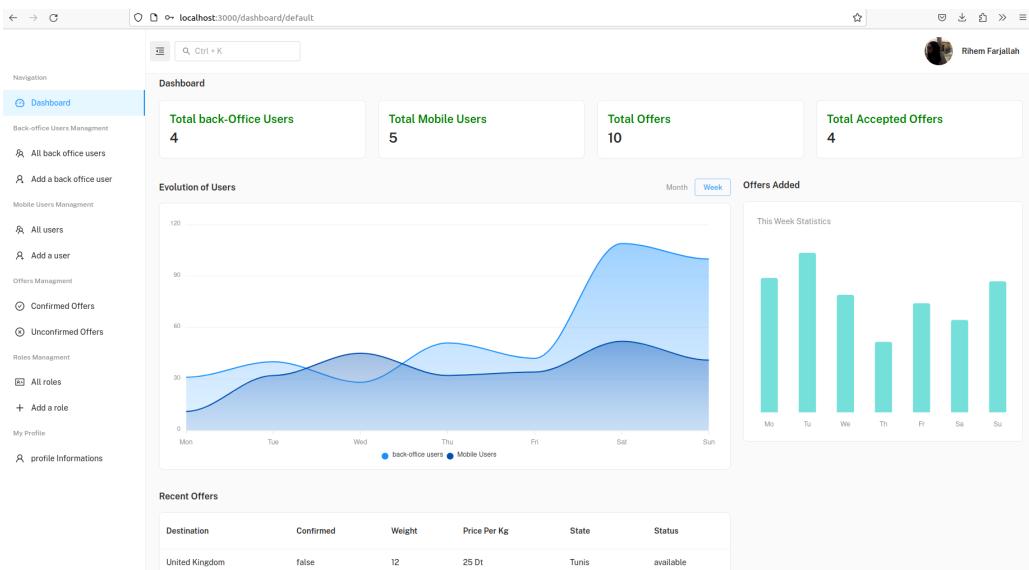
La figure 4.2 représente l'interface d'authentification relative à notre application web.



**FIGURE 4.2 – Interface «Authentification»**

#### — Tableau de bord :

Après avoir effectué l'authentification, l'administrateur aura accès à une interface qui affiche des statistiques. Cela comprend des statistiques relatives aux différents utilisateurs de notre application, aux offres confirmées et non confirmées et plus encore, comme illustré dans la figure 4.3.



**FIGURE 4.3 – Interface «Tableau de bord»**

#### — Gestion des utilisateurs mobile : L'administrateur dispose également de fonctionnalités de gestion des utilisateurs de notre application mobile. En effet, il a le pouvoir d'activer ou de désactiver le compte d'un utilisateur spécifique, et il peut également consulter

en détail les informations d'un compte spécifique. La figure 4.7 illustre l'interface en question.

The screenshot shows a web-based administration interface for managing users. On the left, a navigation sidebar lists various management categories like Dashboard, Back-office Users Management, Offers Management, and Roles Management. The 'All users' section is currently selected. The main content area displays a table titled 'All users' with columns for Profile Picture, Name, Email, Status, and Actions. Two users are listed: 'Siwar Naour' (Email: siwar.naour@gmail.com) with an 'Active' status and 'Rihem Farjallah' (Email: rihem.farjallah999@gmail.com) with an 'Inactive' status. Each user row includes a 'More Details' button and a 'Change Status' button.

**FIGURE 4.4 – Interface «Liste des utilisateurs mobile»**

#### — Gestion des offres non confirmées :

L'administrateur a le droit de confirmer ou de supprimer une offre spécifique lorsqu'il consulte la liste des offres non confirmées. Si l'offre répond aux critères requis, il peut la confirmer. Sinon, il peut la supprimer. De plus, l'administrateur peut utiliser un système intelligent en cours de test pour faciliter ce processus de confirmation. La figure 4.5 illustre l'interface en question.

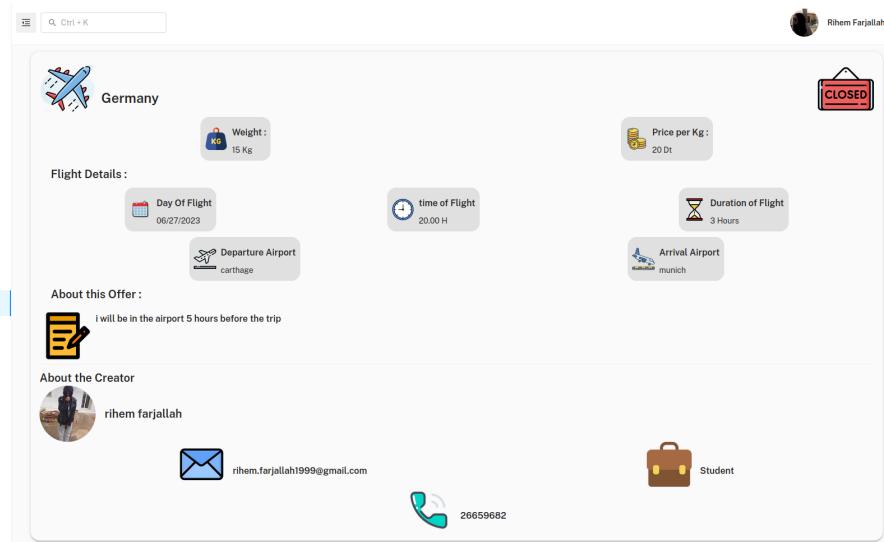
The screenshot shows a web-based administration interface for managing offers. The navigation sidebar includes sections for Offers Management, specifically 'Confirmed Offers' and 'Unconfirmed Offers'. The 'Unconfirmed Offers' section is selected. The main content area displays a table titled 'Unconfirmed Offers' with columns for Creator, Destination, Description, Weight in Kg, Price Per Kg, Created At, and Actions. Four offers are listed:

Creator	Destination	Description	Weight in Kg	Price Per Kg	Created At	Actions
Siwar Naour	United Kingdom	I will be in the airport three hours before my flight and I will be...	12 Kg	25 Dt	07/02/2023, 12:03:23 AM	[Test With IA] [Details] [✓ Confirm] [Delete]
Siwar Naour	France	I will be available in the airport three hours before the flight and...	12 Kg	13 Dt	07/01/2023, 11:47:18 PM	[Test With IA] [Details] [✓ Confirm] [Delete]
Rihem Farjallah	Germany	I will be in the airport 5 hours before the trip	15 Kg	20 Dt	06/25/2023, 11:32:00 PM	[Test With IA] [Details] [✓ Confirm] [Delete]
Siwar Naour	Germany	hi i want to share with you my next destination but shit	30 Kg	20 Dt	06/21/2023, 01:30:02 AM	[Test With IA] [Details] [✓ Confirm] [Delete]

**FIGURE 4.5 – Interface « Liste des offres non confirmées»**

L'administrateur a également le droit de consulter une offre spécifique afin d'obtenir toutes les informations nécessaires à son sujet. Cette consultation lui permet d'acquérir une compréhension approfondie de l'offre.

La figure 4.6 illustre l'interface correspondante à une offre spécifique.



**FIGURE 4.6 – Interface «Détails d'une offre»**

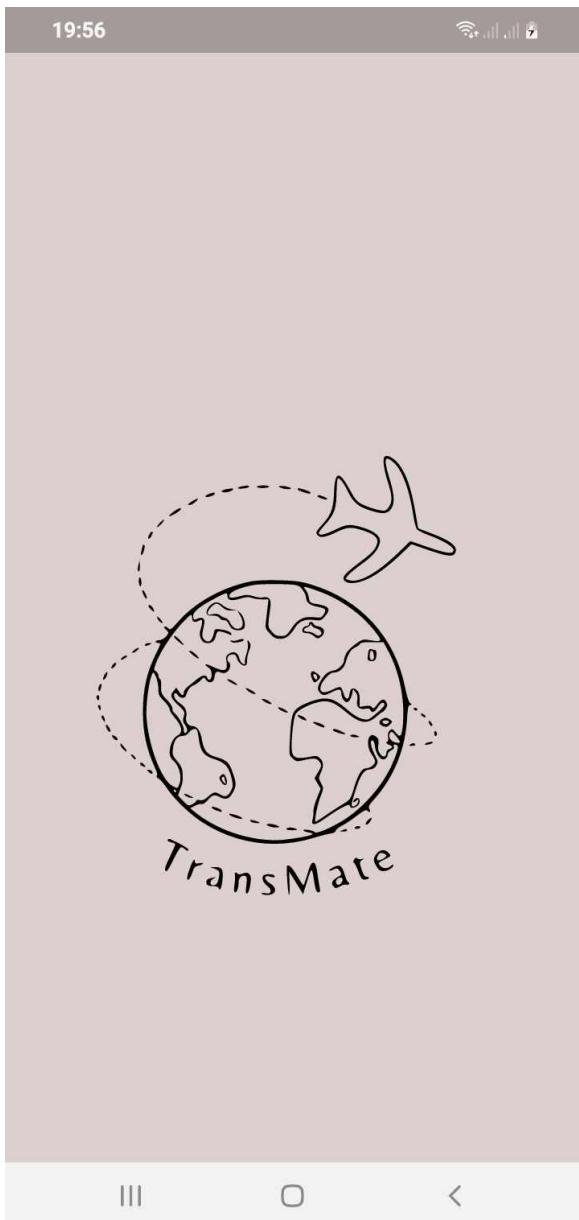
- **Création d'un nouveau rôle** : Lorsqu'un utilisateur de notre application web possède la permission "Create" dans son rôle, il est en mesure de créer un nouveau rôle en remplissant un formulaire et en sélectionnant les permissions à attribuer à ce nouveau rôle. La figure 4.7 représente l'interface de création d'un nouveau rôle pour un utilisateur qui ne possède pas la permission "Create".

This screenshot shows the 'Add a role' form. The sidebar navigation is identical to Figure 4.6. The main form has fields for 'Name\*' (input field) and 'Description\*' (input field). Below these is a section titled 'Select the permissions you want to give:' with checkboxes for 'create', 'update', 'consult', 'delete', and 'confirm'. A 'Create' button is at the bottom right.

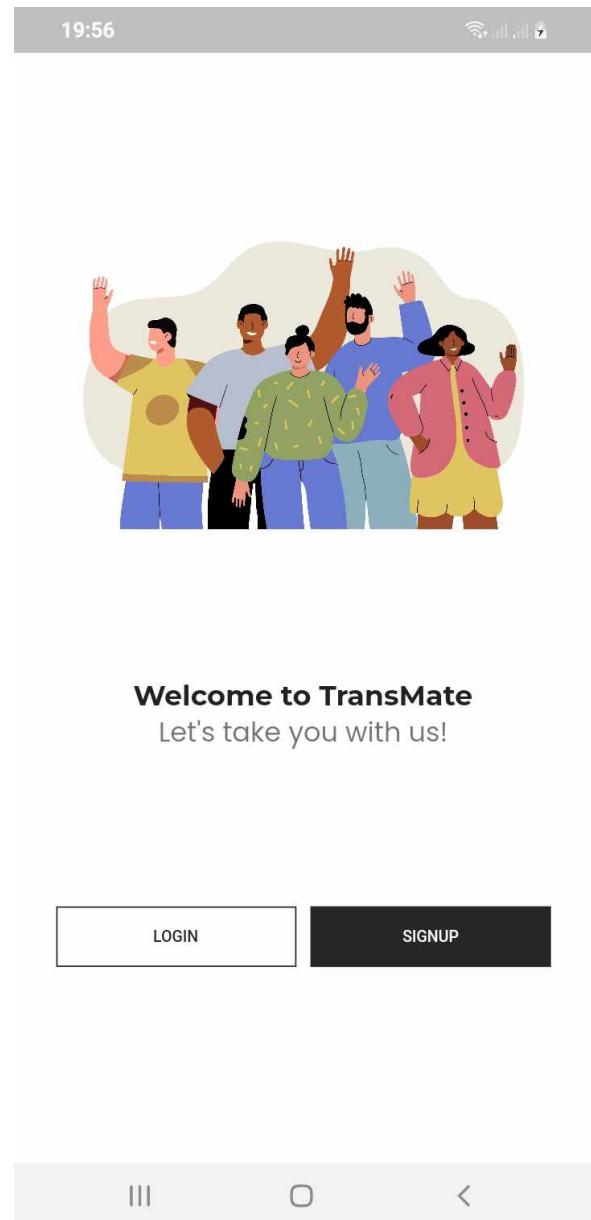
**FIGURE 4.7 – Interface «Ajout d'un nouveau rôle»**

### 4.3.2 Réalisation de l'incrément "Partie Mobile"

- **Interfaces d'accueil** :



(a)



(b)

**FIGURE 4.8 – Interfaces d'accueil**

Avant chaque accès à la partie web, chaque administrateur doit impérativement saisir son email et son mot de passe via l'interface d'authentification. L'application vérifie également si les informations saisies correspondent à celles d'un administrateur existant dans la base de données. La figure ?? illustre l'interface en question.

— **Interfaces d'accueil :**

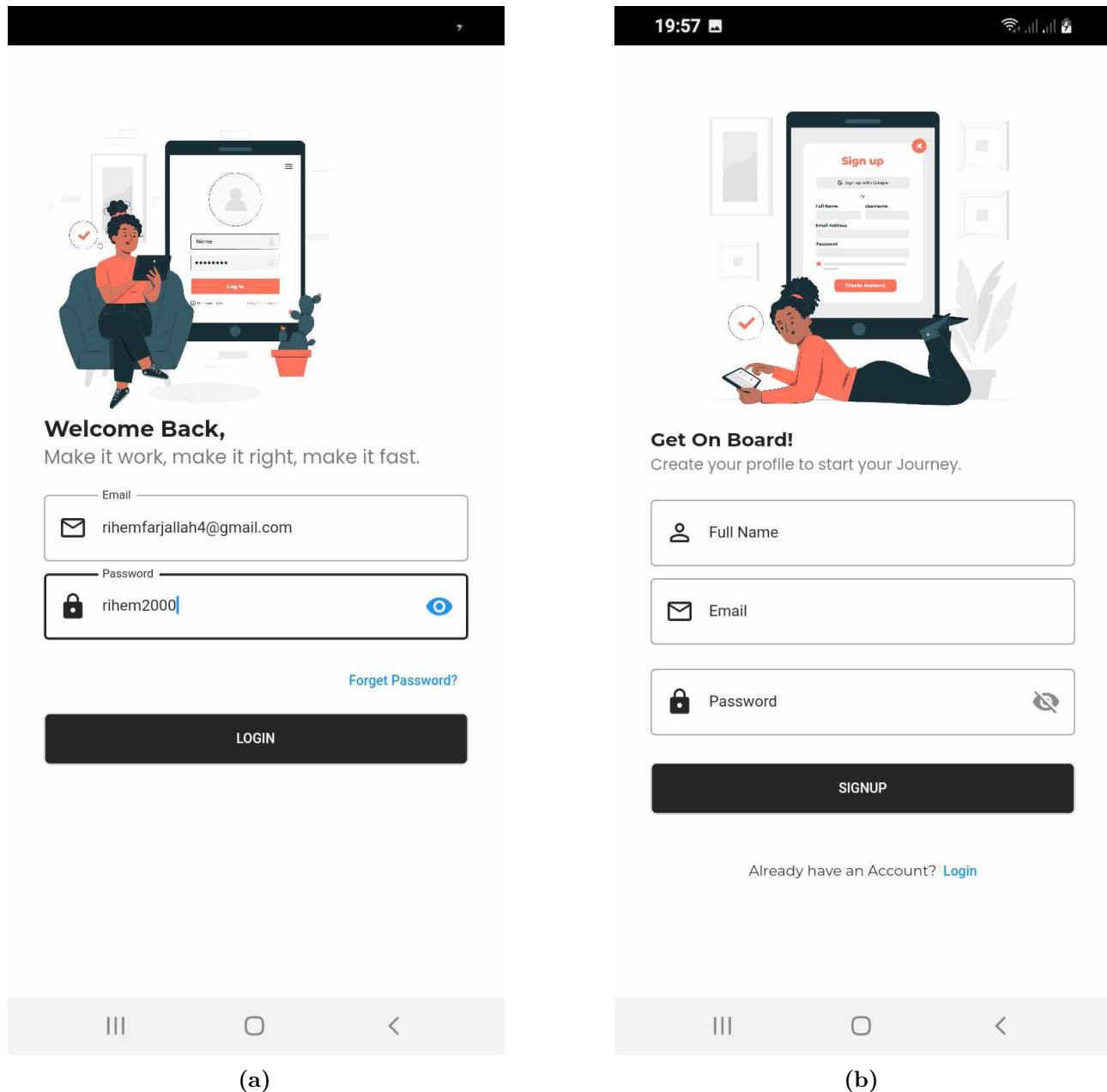


FIGURE 4.9 – Interfaces d'accueil

Avant chaque accès à la partie web, chaque administrateur doit impérativement saisir son email et son mot de passe via l'interface d'authentification. L'application vérifie également si les informations saisies correspondent à celles d'un administrateur existant dans la base de données. La figure ?? illustre l'interface en question.

— **Interfaces d'accueil :**

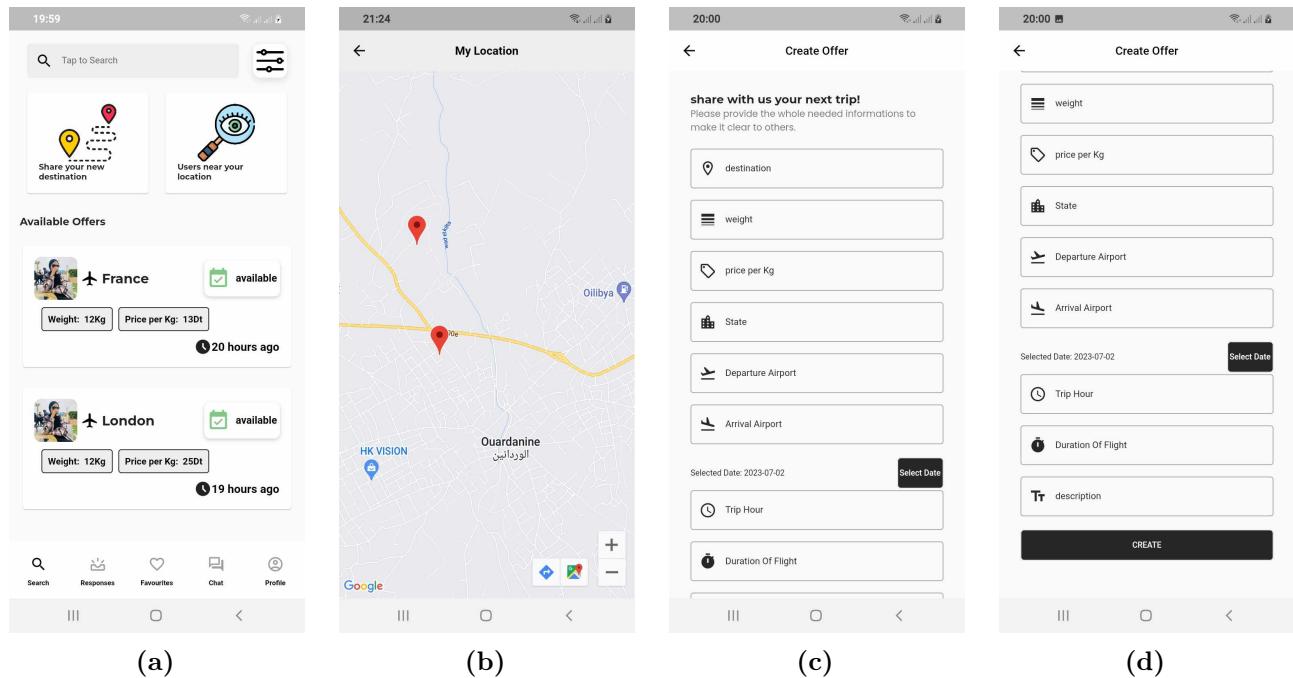


FIGURE 4.10 – Interfaces d'accueil

Avant chaque accès à la partie web, chaque administrateur doit impérativement saisir son email et son mot de passe via l'interface d'authentification. L'application vérifie également si les informations saisies correspondent à celles d'un administrateur existant dans la base de données. La figure ?? illustre l'interface en question.

#### — Interfaces d'accueil :

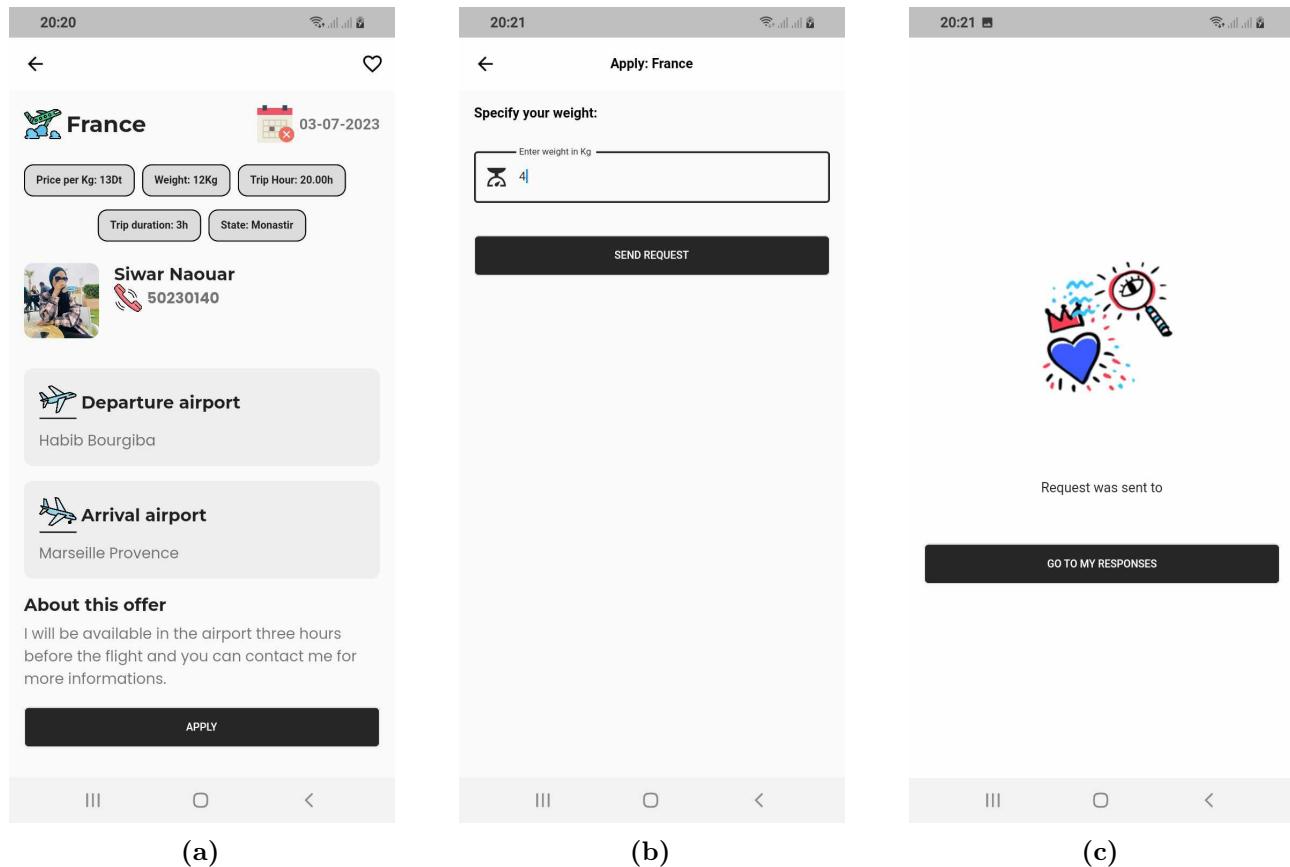


FIGURE 4.11 – Interfaces d'accueil

Avant chaque accès à la partie web, chaque administrateur doit impérativement saisir son email et son mot de passe via l'interface d'authentification. L'application vérifie également si les informations saisies correspondent à celles d'un administrateur existant dans la base de données. La figure ?? illustre l'interface en question.

— **Interfaces d'accueil :**

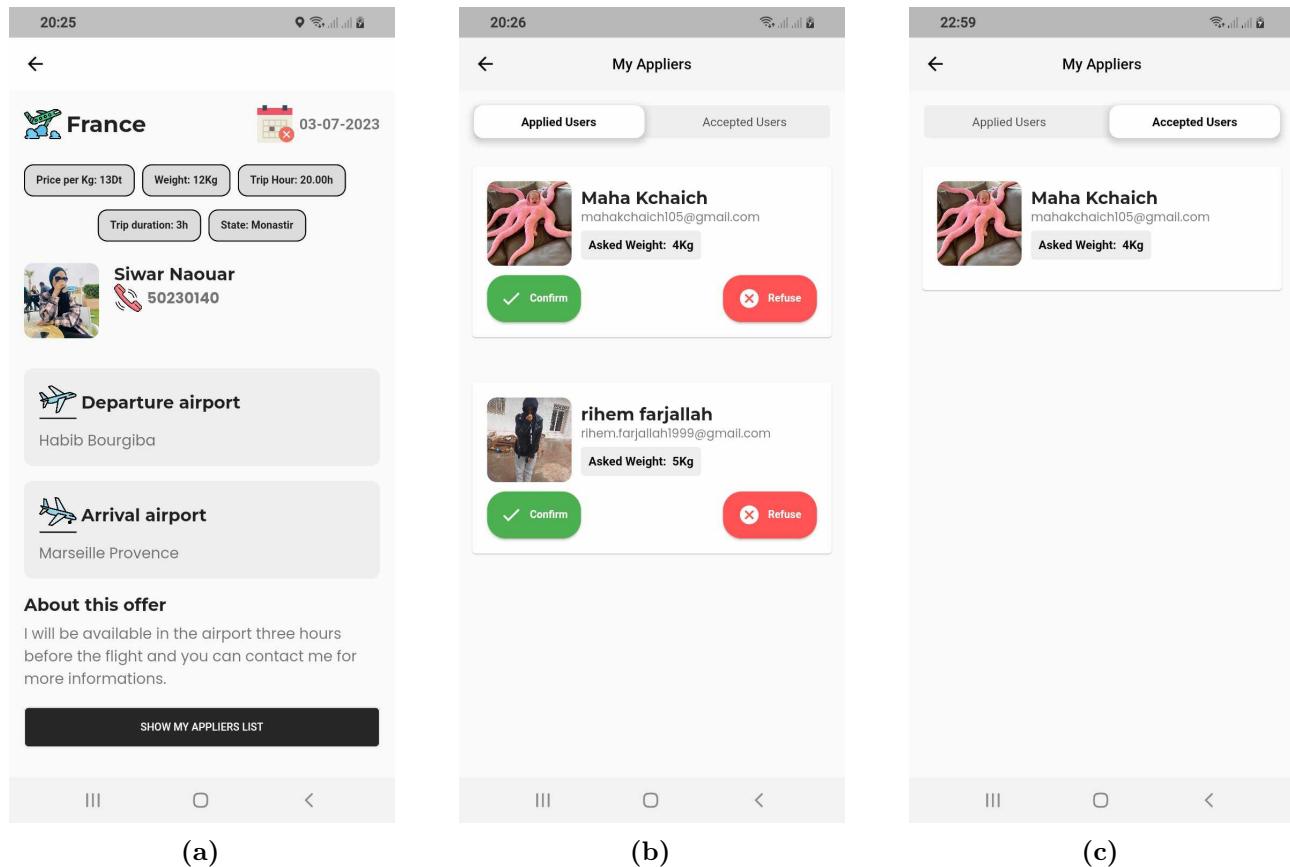


FIGURE 4.12 – Interfaces d'accueil

Avant chaque accès à la partie web, chaque administrateur doit impérativement saisir son email et son mot de passe via l'interface d'authentification. L'application vérifie également si les informations saisies correspondent à celles d'un administrateur existant dans la base de données. La figure ?? illustre l'interface en question.

— **Interfaces d'accueil :**

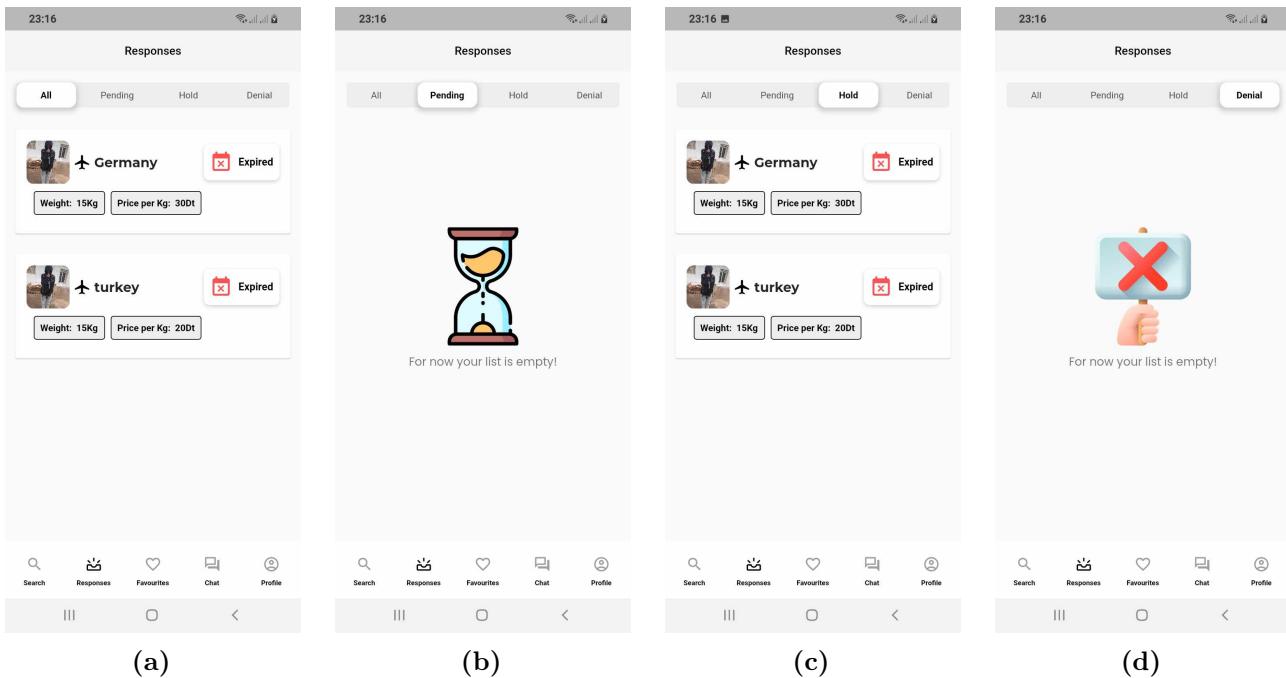


FIGURE 4.13 – Interfaces d'accueil

Avant chaque accès à la partie web, chaque administrateur doit impérativement saisir son email et son mot de passe via l'interface d'authentification. L'application vérifie également si les informations saisies correspondent à celles d'un administrateur existant dans la base de données. La figure ?? illustre l'interface en question.

#### — Interfaces d'accueil :

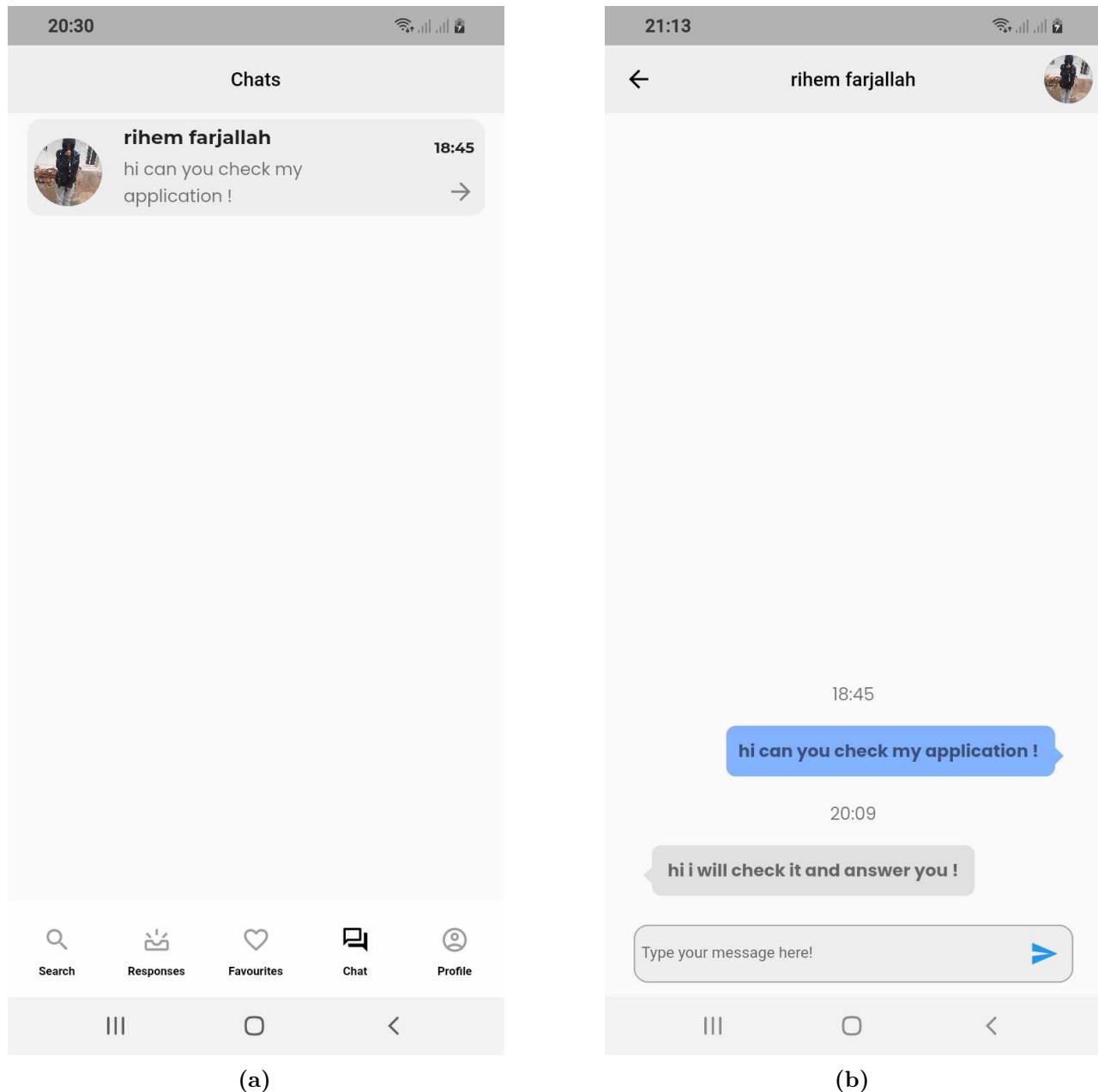


FIGURE 4.14 – Interfaces d'accueil

Avant chaque accès à la partie web, chaque administrateur doit impérativement saisir son email et son mot de passe via l'interface d'authentification. L'application vérifie également si les informations saisies correspondent à celles d'un administrateur existant dans la base de données. La figure ?? illustre l'interface en question.

— **Interfaces d'accueil :**

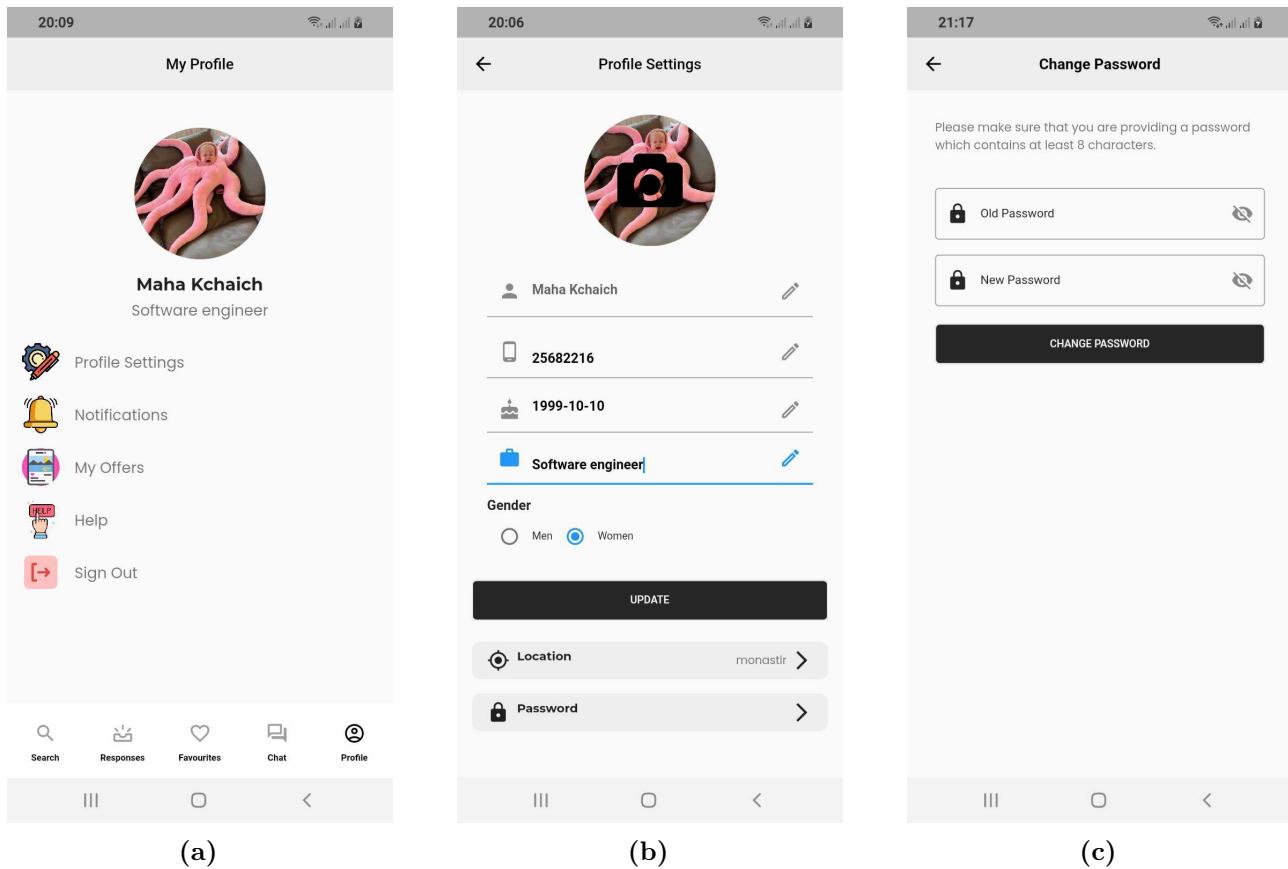
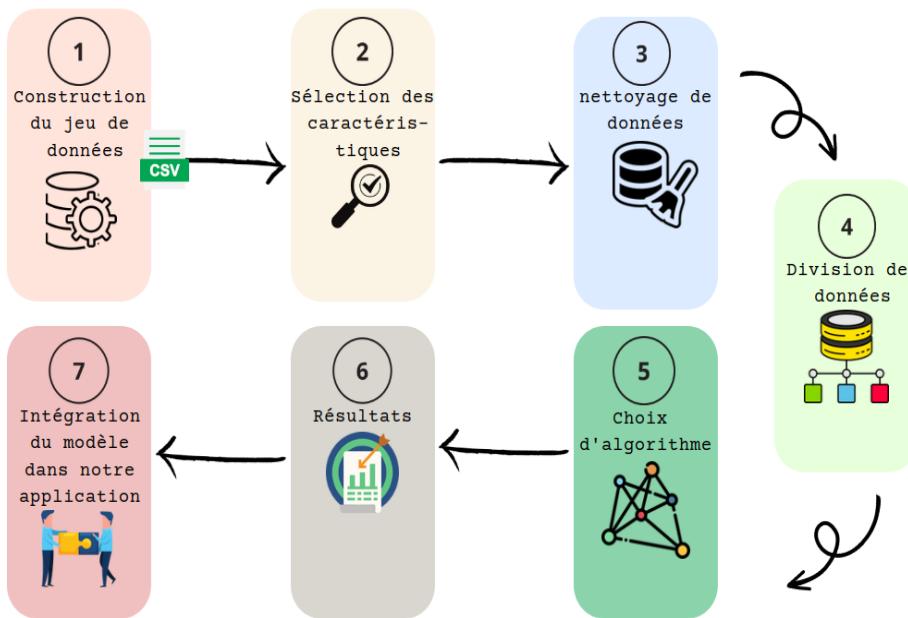


FIGURE 4.15 – Interfaces d'accueil

Avant chaque accès à la partie web, chaque administrateur doit impérativement saisir son email et son mot de passe via l'interface d'authentification. L'application vérifie également si les informations saisies correspondent à celles d'un administrateur existant dans la base de données. La figure ?? illustre l'interface en question.

### 4.3.3 Réalisation de l'incrément "Module IA"

Lors de la discussion de notre méthodologie de travail, nous avons décidé de construire une figure globale présentant chaque étape de notre parcours. La figure 4.16 a été mise en place afin d'aider chaque membre de notre équipe à comprendre précisément ce que nous allons faire.

**FIGURE 4.16 – Flux de travail**

### — Construction du jeu de données :

Dans notre projet, nous souhaitons que nos résultats attendus soient aussi proches de la réalité que possible, afin d'avoir un modèle intelligent capable d'être lancé après 6 mois de collecte de données. Pour atteindre cet objectif, nous avons contacté l'équipe de SAHTICARE pour demander des données relatives au contenu publié par des utilisateurs. Heureusement, notre demande a été approuvée et nous avons obtenu l'accès aux données. En utilisant ces données fournies, nous avons pu compiler un fichier CSV prêt à être importé dans notre script python.

### — Préparation du jeu de données :

#### 1. La sélection des caractéristiques :

La sélection des caractéristiques est une étape cruciale dans la construction des modèles. L'objectif est d'identifier les colonnes qui sont pertinentes et informatives pour le processus de confirmation du contenu, tout en éliminant celles qui ne contribuent pas significativement à la prédiction. La figure 4.17 montre les différentes caractéristiques que notre fichier CSV contient. Nous allons garder uniquement les colonnes "class" et "content". En fait, la colonne "class" contient la valeur 1 lorsque le contenu de la colonne "content" est inapproprié, et 0 sinon.

	A	B	C	D	E	F	
1	id	count	hate_speech	offensive_language	neither	class	content

**FIGURE 4.17 – Caractéristiques du jeu de données initiales**

#### 2. Nettoyage des données

Afin d'obtenir un contenu compréhensible par notre algorithme et prêt à être utilisé, nous avons créé une fonction comprenant des opérations telles que la mise en minuscules du texte, la suppression des éléments entre crochets, des URL, des balises HTML, de la ponctuation et des mots contenant des chiffres. De plus, elle filtre

les mots en fonction d'une liste prédéfinie de mots vides (stopwords) et applique le stemming pour réduire les mots à leur forme racine. En utilisant cette fonction, nous avons préparé et nettoyé notre colonne "content" afin d'obtenir des textes adaptés à la tâche d'apprentissage automatique, en éliminant les informations indésirables et en normalisant le texte.

### 3. Division des données

Le jeu de données est composée de 80560 lignes catégorisées comme suit :

- 64448 lignes pour l'entraînement.
- 16112 lignes pour le test.

La figure 4.18 présente la répartition des données dans notre jeu de données.

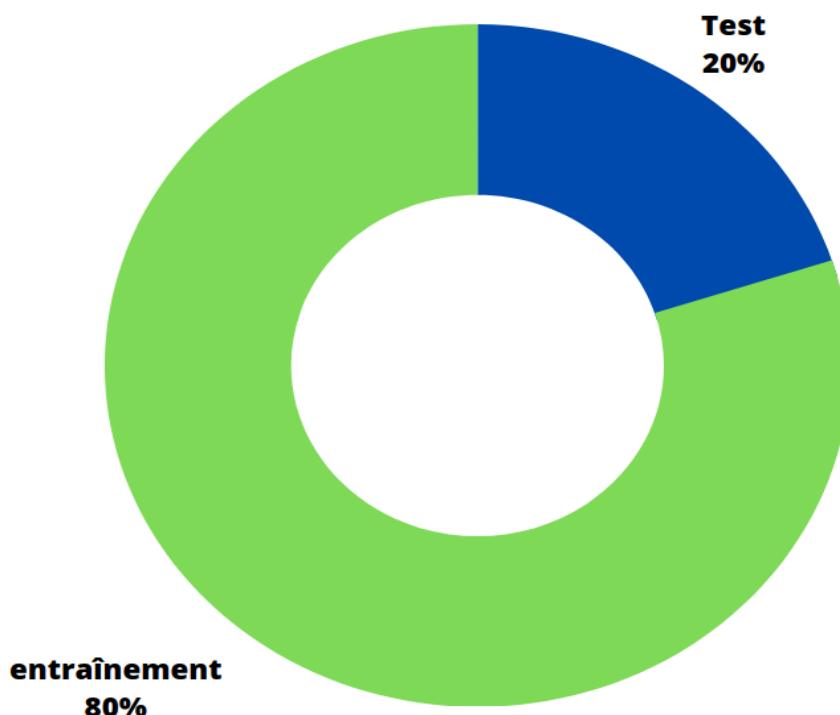


FIGURE 4.18 – Répartition du jeu de données

#### — Algorithme choisi :

L'algorithme LSTM (Long Short-Term Memory) est une variante des réseaux de neurones récurrents (RNN) spécialement conçue pour traiter et modéliser des séquences de données avec des dépendances à long terme. Contrairement aux RNN traditionnels, qui peuvent rencontrer des problèmes de disparition ou d'explosion du gradient lors de la rétropropagation, les LSTM permettent de conserver et d'apprendre des informations sur de longues séquences, ce qui en fait un choix populaire pour les tâches de modélisation de séquences, telles que l'analyse de texte. Nous allons utiliser cet algorithme pour évaluer si

le contenu du champ "description" saisi par l'utilisateur lors de la création d'une nouvelle offre est approprié ou non. Cette approche nous permettra de tester la pertinence du contenu en fonction de critères prédéfinis. Les LSTM sont particulièrement adaptés pour analyser et comprendre le contenu textuel, grâce à leur capacité à gérer les dépendances à long terme. Ils peuvent identifier des motifs et des contextes plus complexes dans le texte, ce qui nous permettra de déterminer si le contenu est approprié ou non.

L'architecture LSTM repose sur des unités de mémoire appelées "cellules LSTM". Chaque cellule LSTM est composée de trois principales composantes :

1. **Portes (gates)** : Les portes sont des mécanismes de régulation qui contrôlent le flux d'informations à travers la cellule LSTM. Les trois types de portes utilisés dans une cellule LSTM sont :
  - Porte d'oubli (forget gate)** : Détermine quelles informations de la cellule mémoire précédente doivent être oubliées.
  - Porte d'entrée (input gate)** : Détermine quelles nouvelles informations doivent être ajoutées à la cellule mémoire.
  - Porte de sortie (output gate)** : Détermine quelle partie de la cellule mémoire doit être exposée en tant que sortie.
2. **Cellule mémoire** : La cellule mémoire est responsable de stocker et de maintenir les informations à long terme. Elle est mise à jour en fonction des entrées actuelles et des sorties des portes.
3. **État caché (hidden state)** : L'état caché est la sortie de la cellule LSTM à un certain moment. Il peut être utilisé comme entrée pour la cellule LSTM à l'étape suivante ou comme sortie pour la tâche spécifique à résoudre.

La figure 4.19 illustre la composition d'une cellule LSTM.

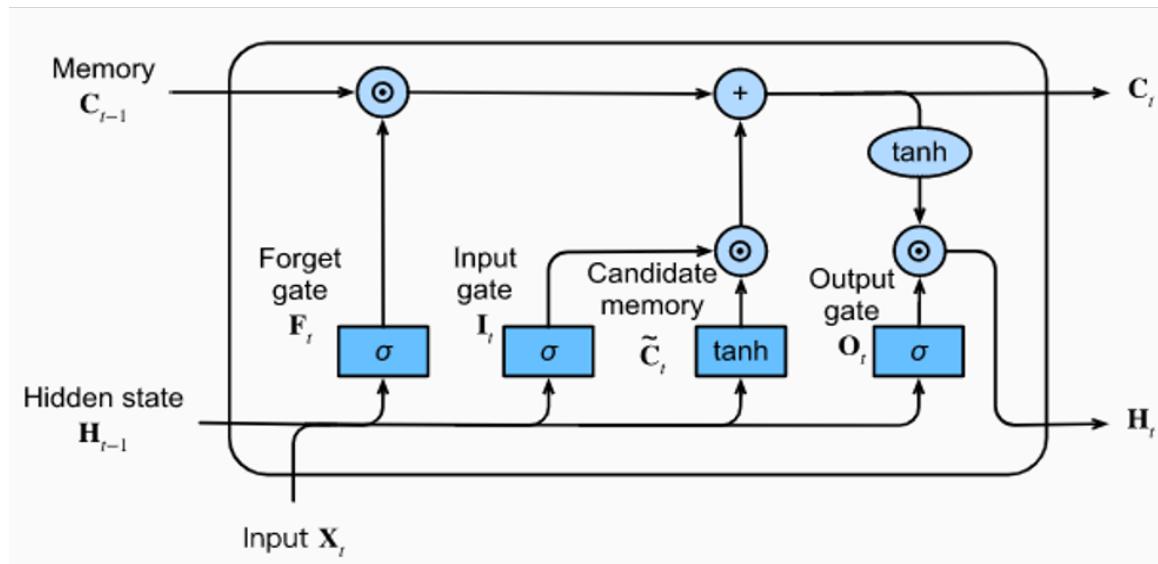


FIGURE 4.19 – Architecture d'une cellule LSTM

L'algorithme LSTM fonctionne en séquence, en traitant les données séquentiellement dans le temps. À chaque pas de temps, les portes de la cellule LSTM sont mises à jour en fonction de l'entrée actuelle et de l'état précédent, et la cellule mémoire est mise à jour en conséquence. L'état caché est calculé en utilisant la sortie de la cellule LSTM et peut être utilisé pour prédire des valeurs ou pour d'autres tâches de traitement ultérieures.

#### — Résultats des tests :

L'accuracy per epoch (précision par époque) est une métrique utilisée lors de l'entraînement d'un modèle d'apprentissage automatique, en particulier dans les réseaux de neurones profonds. Elle mesure la précision du modèle sur les données d'entraînement à chaque époque pendant le processus d'apprentissage.

Lorsque nous entraînons notre modèle, nous avons déjà divisé notre jeu de données en un

ensemble d'entraînement et un ensemble de test. À chaque époque, le modèle passe par l'ensemble d'entraînement, effectue des prédictions sur les données et calcule la précision en comparant les prédictions aux étiquettes de classe réelles. La figure 4.20 présente le changement de valeur de la précision par époque dans notre modèle.

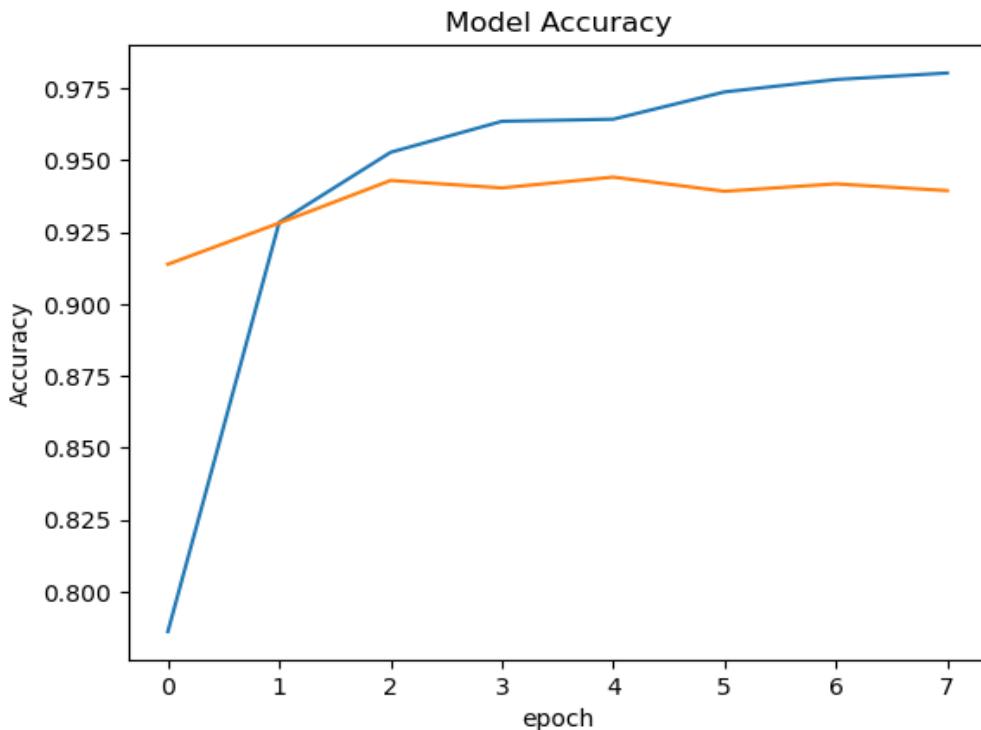


FIGURE 4.20 – Précision par époque

La précision est une mesure couramment utilisée pour évaluer la performance d'un modèle de classification dans le domaine de l'apprentissage automatique. Elle permet de quantifier la capacité d'un modèle à effectuer des prédictions correctes. Elle est calculée en comparant les prédictions faites par le modèle aux étiquettes réelles des données. Elle représente le pourcentage de prédictions correctes par rapport à l'ensemble des prédictions. La figure 4.21 illustre la précision résultante de notre modèle ainsi que la perte qui représente l'erreur entre les prédictions faites par le modèle et les étiquettes réelles des données.

```
444/444 [=====] · 18s 41ms/step · loss: 0.2114 · accuracy: 0.9354
```

FIGURE 4.21 – taux de précision et perte

#### — Intégration :

Pour intégrer notre script Python dans notre back-end développé avec Express.js, nous avons utilisé la bibliothèque childProcess. Cette bibliothèque nous permet d'exécuter le script Python en tant que processus enfant.

En utilisant la fonctionnalité childProcess, nous avons défini un endpoint de l'API, '/classify', qui traite les requêtes GET. Lorsqu'une requête est reçue, le texte envoyé par le client est extrait de la requête. Ensuite, nous construisons une commande pour exécuter le script Python en incluant le chemin vers le script et le texte en tant qu'argument.

Grâce à la fonction exec de childProcess, le script Python est exécuté en tant que processus enfant. Le résultat de l'exécution, qu'il s'agisse d'une sortie ou d'une erreur, est capturé via le callback fourni. Si une erreur se produit lors de l'exécution du script, nous affichons l'erreur dans la console et renvoyons une réponse d'erreur appropriée au client.

Si tout se passe bien, nous récupérons la sortie du script Python, qui représente la prédiction, et nous renvoyons cette prédiction en tant que réponse au client.

Grâce à cette intégration, nous avons créé une API qui permet à notre partie front-end d'envoyer des requêtes de classification de texte. Le backend exécute le script Python pour effectuer la classification et renvoie les résultats au client. Cela nous permet de combiner les fonctionnalités de notre script Python avec notre backend développé en Express.js.

## Conclusion

Au cours de ce dernier chapitre nous avons présenté l'environnement matériel et logiciel pour l'implémentation du logiciel, l'architecture physique à l'aide d'un diagramme de déploiement et les résultats de développement à l'aide des imprimés écran de quelques interfaces de notre plateforme.

# Conclusion générale

Ce rapport a présenté le travail réalisé au sein de la société de développement informatique Dot-IT dans le cadre du projet de fin d'études. Il s'agit d'une conception et réalisation d'une plateforme web et mobile ayant pour finalité la gestion efficace des distributions de produits.

En effet, nombreuses sociétés de distribution en Tunisie demeurent jusqu'à ce jour démunies d'un système informatisé capable de réaliser le traitement des commandes et le suivi de leur distribution jusqu'au point de vente et restent sur des moyens traditionnels où rien n'est centralisé et par conséquent l'exécution de certaines opérations de gestion censée être simple devient de plus en plus compliquée sans oublier l'incapacité d'avoir une vue d'ensemble sur la santé de la société. Ces défaillances ainsi que les points faibles de certaines plateformes disponibles sur le marché nous ont conduit à développer une solution proposant à ses utilisateurs une expérience de qualité grâce à des fonctionnalités évoluées et une utilisation simplifiée.

Le présent rapport détaille les différentes phases de réalisation du projet. Nous avons décrit, tout d'abord, le cadre général en présentant une panoplie de solutions existantes. Ensuite, nous nous sommes concentrés sur l'identification des acteurs et la description des différentes fonctionnalités qu'ils peuvent exploiter. L'illustration des aspects statiques et dynamique, dans la troisième partie, avec les diagrammes de classes, séquences et DCP nous a permis de mieux appréhender la structure et le comportement de notre système. Finalement nous avons mentionné les technologies dont nous avons fait recours et présenté les principales interfaces graphiques. Cette solution développée a permis de valider nos objectifs initiaux et peut donner suite à une

éventuelle évolution enrichie par de nouvelles fonctionnalités telles que :

- Créer une application web et mobile : Pour permettre aux responsables des points de vente de passer personnellement leurs commandes.
- Implémenter un module de paiement par carte de crédit : Lors de l'achèvement d'une distribution ou la réalisation d'une vente, le responsable de - 101 -la boutique pourra payer le montant de la facture en utilisation sa carte de crédit sur le TPE du commercial.
- Modifier la commande : Dans le cas où le responsable du point de vente changerait son avis lors de réception de la commande, le commercial pourra ajouter ou supprimer des produits.
- Proposer automatiquement le véhicule de distribution : En fonction de la catégorie, volume et poids des produits contenus dans la commande, le système pourra proposer le type de véhicule adéquat pour la distribution.
- Messagerie instantanée : Puisque les TPE ne peuvent pas effectuer des appels, la création d'une messagerie instantanée facilitera la communication du commercial avec l'administrateur.

- Gérer les droits d'accès : Le super admin pourra créer un nouveau rôle avec ses priviléges personnalisés et l'affecter à un administrateur lors de la création de son compte.

En dernier lieu, ce projet a été une expérience enrichissante en termes d'apprentissage, mise en pratique des connaissances théoriques en utilisant des frameworks populaires et puissants sans oublier l'amélioration des aptitudes de communication qui représenteront sans doute des avantages lors de l'intégration du monde professionnel.

# Netographie

- [1] dernière visite : le 23/06/2023. URL : <https://fr.legacy.reactjs.org/>.
- [2] dernière visite : le 23/06/2023. URL : <https://flutter.dev/multi-platform>.
- [3] dernière visite : le 23/06/2023. URL : <https://nodejs.org/en/about>.
- [4] dernière visite : le 23/06/2023. URL : <https://expressjs.com/>.
- [5] dernière visite : le 23/06/2023. URL : <https://socket.io/docs/v4/>.
- [6] dernière visite : le 23/06/2023. URL : <https://dart.dev/overview>.
- [7] dernière visite : le 23/06/2023. URL : <https://docs.python.org/3/tutorial/index.html>.
- [8] dernière visite : le 23/06/2023. URL : <https://www.json.org/json-en.html>.
- [9] dernière visite : le 23/06/2023. URL : <https://bootcamp.berkeley.edu/blog/most-in-demand-programming-languages/>.
- [10] dernière visite : le 23/06/2023. URL : <https://www.mongodb.com/>.
- [11] dernière visite : le 23/06/2023. URL : <https://code.visualstudio.com/docs>.
- [12] dernière visite : le 23/06/2023. URL : <https://www.postman.com/company/about-postman/>.
- [13] dernière visite : le 23/06/2023. URL : <https://developer.android.com/studio>.
- [14] dernière visite : le 23/06/2023. URL : <https://draw.io/>.
- [15] dernière visite : le 23/06/2023. URL : <https://www.overleaf.com>.
- [16] dernière visite : le 23/06/2023. URL : <https://about.gitlab.com>.