



Ecole Supérieure d'Informatique et  
d'Analyse et des Systèmes



Université Mohammed-V de Rabat

## Rapport du projet fédérateur du semestre 4

---

### Sujet

Conception et réalisation d'une application mobile  
pour la livraison et la gestion des commandes.

---

#### Soutenu par :

TAIR Zouhir  
OUKHALEK Ismail

#### Sous la direction de :

M.Mohammed SENHADJI  
Mme.Karima MOUMANE  
M. Rachad TAOUFIK  
Mme. Naoual CHAOUNI

Année Universitaire : 2020-2021





*A nos chers parents*

*Les mots nous manquent pour exprimer toute la  
reconnaissance, la fierté et le profond amour que nous vous  
portons pour les sacrifices que vous avez consentis pour notre  
réussite.*

*Que Dieu vous préservent bonne santé et longue vie.*

*A nos chers frères*

*Nous espérons avoir atteint le seuil de vos espérances.*

*Nous vous remercions pour le soutien moral et  
l'encouragement que vous avez accordés. Nous vous  
souhaitons un brillant avenir.*

*A Toute personne qui vient de chercher son nom ici.*

*Un mot final pour vous tous, Nous espérons que nous étions à la  
hauteur de vos espérances et sachez que jamais nous pourrions  
oublier ce que vous avez fait pour nous.*

## 0.1 Remerciements

On adresse nos sincères remerciements à nos encadrants Mme. Karima MOUMANE, M. ZELLOU Ahmed, M. RACHAD Toufik et M. Naoual CHAOUNI, qui ont veillé pas à pas à l'élaboration de ce projet, on les remercie pour leur disponibilité, leur collaboration et leurs précieux conseils pour pouvoir nous mettre sur la bonne voie. On remercie également tout le cadre professoral de département de la filière de l'Ingénierie Web et d'Informatique Mobile qui nous assure une formation riche et qualifiante.

Nous tenons aussi à remercier les membres du jury M. Mohammed SENHADJI Mme. Karima MOUMANE pour leur attention et leur intérêt qu'ils portent à notre travail.

Nos vifs remerciements vont également à nos familles, nos amis et nos collègues pour leur aide précieuse et leur encouragement.

Nous espérons que le présent écrit reflète notre travail assez clairement, et souligne avec pertinence les différents côtés que nous avons dû aborder. Comme nous souhaitons que la réalisation de ce projet soit à la hauteur des attentes de nos professeurs.

## 0.2 Résumé

Ce rapport présente les différentes étapes d'analyse, conception et de réalisation d'une application mobile pour la livraison et la gestion des commandes dans les restaurants et les snacks. Cette application va faciliter la tâche pour les clients, les employées et les livreurs en leur permettant de facilement accéder et bénéficier des différents services que cette application propose.

Afin de mener à bien notre mission, il s'est avéré indispensable d'entamer le travail par une étude fonctionnelle permettant d'identifier la problématique, ainsi que les caractéristiques du système à développer. Puis, nous avons opté pour une étude conceptuelle afin de mieux définir le comportement de notre application ; ceci à travers les diagrammes UML, de séquence et de classe. Ce qui a rendu la phase d'implémentation plus spontanée. Le développement a été réalisé en utilisant les langages XML et JAVA ainsi que ANDROID STUDIO comme environnement logiciel.

**Mots clés :** UML, XML, JAVA, ANDROID STUDIO, FIREBASE, Gestion. Livraison.

### 0.3 **Abstract**

This report presents the different steps of analysis, design and implementation of a mobile application for the delivery and management of orders in the restaurant. This application will make it easier for the employees, customers and administrators of the restaurant. These can manage or place an order easily.

In order to carry out our mission, it proved necessarily to start the work with a functional study ;to identify the problem, as well as the characteristics of the system to be developed. Then, we opted for a conceptual study to define a better behavior of our application through UML, sequence and class diagrams. Which made the implementation phase more spontaneous..The development was carried out using XML and JAVA technology and the ANDROID STUDIO as the software environment.

**Keys words :** UML, XML, JAVA, ANDROID STUDIO, FIREBASE, Management,Delivry.

## Liste des abréviations

***XML***     *Extensible Markup Language*

***JAVA***     *A general purpose, high-level, object-oriented, cross-platform  
programming language developed by Sun Microsystems.*

***FB***     *Fire +base*

## 0.4 Table des figures

Figure 1 diagramme de cas d'utilisation.....	21
Figure 2 :Diagramme de séquence client.....	23
Figure 3 :diagramme de sequence admin.....	24
Figure 4 :diagramme de séquence livreur.....	25
Figure 5 :Diagramme de classe .....	26
Figure 6 :pseudo code algorithme A*.....	31
Figure 7 :schémat de l'architecture d'Android.....	33
Figure 8 :Architecture Java.....	34
Figure 9 :Dalvik .....	36
Figure 10 :XML .....	37
Figure 11 :java .....	38
Figure 12 :Android studio.....	39
Figure 13 :logo firebase.....	39
Figure 14 :google places api .....	40
Figure 15 :Login and Sign Up.....	42
Figure 16 :espace client .....	42
Figure 17 :faire une commande.....	43
Figure 18 :historique des commandes. ....	44
Figure 19 :espace restaurant.....	45
Figure 20 : Gérer les commandes.....	46
Figure 21 :reçoit de notification.....	47
Figure 22 :voir la géolocalisation d'un livreur.....	48
Figure 23 :affectation d'un livreur.....	49
Figure 24 :espace livreur .....	50



## 0.5 Table des matières

<b>1.</b>	<b>Contexte général du projet</b>	<b>13</b>
1.1.	<i>Introduction</i>	13
1.2.	<i>Problématique</i>	13
1.3.	<i>Objectif général du projet</i>	13
1.4.	<i>Besoins et objectifs</i>	14
1.5.	<i>Méthodologie de travail</i>	15
1.6.	<i>Conclusion</i>	15
<b>2.</b>	<b>Analyse et conception</b>	<b>146</b>
2.1.	<i>Analyse</i>	
2.1.1.	<i>Description du projet</i>	
2.1.2.	<i>Les besoins</i>	
2.1.3.	<i>Identification des acteurs</i>	
2.1.4.	<i>Diagramme des cas d'utilisation</i>	
2.2.	<i>Conception</i>	
2.2.1.	<i>Diagramme de séquence</i>	
2.2.2.	<i>Diagramme de classe</i>	
2.3.	<i>Conclusion</i>	
<b>3.</b>	<b>Intelligence Artificielle et système de recommandation</b>	<b>17</b>
3.1.	<i>Introduction</i>	
3.2.	<i>Définition</i>	
3.3.	<i>Historique des Systèmes de Recommandation</i>	
3.4.	<i>Algorithme A*</i>	

3.4.1.	<i>Présentation</i>	
3.4.2.	<i>Propriétés</i>	
3.4.3.	<i>Description</i>	
3.4.4.	<i>Pseudo-code</i>	
<b>4.</b>	<b>Réalisation du projet</b>	<b>20</b>
4.1.	<i>Présentation de l'environnement du travail</i>	20
4.1.1.	Architecture de l'application	
4.1.2.	Les technologies déployées	
4.2.	<i>Implémentation</i>	
4.2.1.	<i>Page d'accueil</i>	
4.2.2.	<i>Espace Client</i>	
4.2.3.	<i>Espace administrateur</i>	

# Introduction générale

Depuis de nombreuses décennies, les systèmes d'informations sont présents dans les organisations. D'abord sous forme physique, ensuite sous forme électronique, ils occupent quotidiennement une place de choix dans les organisations, d'une part à cause du renforcement de la concurrence sur les marchés et la masse d'informations à gérer, et d'autre part grâce au développement constant des nouvelles technologies de l'information qui apportent des solutions plus pertinentes. C'est dans ce contexte que plusieurs restaurants essayent de profiter au maximum possible de ces technologies afin d'améliorer leur productivité et faire face à des problèmes pénibles.

Certes, les difficultés que connaît la plupart des restaurants à savoir la gestion de commandes de leurs clients, surtout les problèmes qu'ils rencontrent dans la livraison de leur repas et la carte de menu qui ne laisse pas la tâche facile aux clients de retrouver un plat. Vu que ce menu n'a pas d'image du plat proposé et qu'il change souvent, cette forme de travail cause souvent des préjudices. Pour cela, le choix d'un système informatique avec ses influences devant permettre de fluidifier le processus de traitement des commandes et d'éviter ainsi la falsification constatée en forte période reste indispensable.

Dans ce cadre, nous avons pensé à développer une application mobile permettant de gérer les échanges de données entre les clients et les administrateurs du site.

Notre projet vise donc non seulement à créer une application mobile qui aide l'employé (administrateur) à automatiser la tâche de gestion des commandes, des menus, et affecter un livreur à une commande, mais aussi à consulter le menu et les commandes pour les clients ainsi que cette application va permettre au livreur de voir en temps réel le trajet qui le sépare avec sa destination.

Dans ce rapport, nous allons essayer de mettre en évidence les étapes suivies pour la mise en œuvre de ce projet. Et ceci à travers quatre chapitres :

Tout d'abord, le premier chapitre est consacré au « Contexte général du projet » et permet de placer le projet dans son contexte et dégager les caractéristiques du système courant, ses limites et puis une description du projet et les fonctionnalités qu'il apporte.

Le second chapitre intitulé « Etude conceptuelle » consiste en une modélisation des interactions entre les acteurs et objets du système via les diagrammes UML de séquences et puis de classe.

Le troisième chapitre intitulé « Intelligence Artificielle et système de recommandation » où on va décrire la partie intelligente de notre application.

Le dernier chapitre « Réalisation » aborde de manière détaillée tous les outils utilisés pour le développement de notre application, ainsi que quelques captures d'écran de la version finale de notre système.

# <sup>1</sup>Chapitre 1

---

## **1.1** Contexte général du projet

*Dans ce chapitre nous allons procéder à l'étude du système existant ; dégager ses limites afin de pouvoir proposer un modèle adéquat qui va répondre aux besoins des utilisateurs.*

# **1. Contexte général du projet**

## ***1.1 Introduction***

Le déroulement de tout projet ne peut être isolé de son cadre, ce premier chapitre est donc consacré à la présentation du contexte général de notre projet. Ainsi, nous commencerons par une présentation de la problématique avec le système actuel, ensuite nous présenterons l'objectif général de notre projet, puis les besoins et les objectifs spécifiques et nous monterons la méthodologie de travail par la suite.

## ***1.2 Problématique***

La gestion au sein des restaurants se fait manuellement. On remarque ainsi la mauvaise organisation du travail dans ces derniers ce qui cause d'énormes problèmes soit pour les employés, soit pour les clients, soit pour les livreurs qui sont insatisfaits des services proposés lors de la saisie d'une information. Ainsi l'information n'est pas toujours précise ni disponible ; d'où la nécessité d'une application pour faire et gérer les commandes et automatiser le processus de la livraison et le rendre mieux digitalisé.

## ***1.3 Objectif général du projet***

Notre objectif est de créer une application qui permet de gérer facilement les services proposés par n'importe quelle restaurant. En assurant une fluidité de communication entre les clients et les employés afin de mieux contrôler et suivre le processus des commandes.ainsi que digitaliser le service de livraison et mettre fin au problème rencontré dans cette phase.

## ***1.4 Besoins et objectifs***

- Permettre aux administrateurs de :
  - Consulter et gérer les commandes,
  - Choisir l'état des commandes (validés, annulés)
  - Affecter un livreur à une commande.
  
- La mise en place d'une application destiné à faciliter au client :
  - La consultation des différents restaurants qui existe dans l'application
  - La sélection du plat qu'il désire savoir sa facture
  - Consulter la liste de ses commandes.
  - Permettre au client de laisser l'avis sur son expérience avec les services proposés.
  
- Permettre aux livreurs de voir en temps réel le trajet qui le sépare de sa destination.

## ***1.5 Méthodologie de travail***

- Faire une analyse globale pour le projet.
- Concevoir l'application par le langage UML.
- Développer une application mobile avec Java et XML.
- Concevoir et développer une base de données avec FIREBASE et l'intégrer dans l'application

## ***1.6 Conclusion***

Ce chapitre sert de point de départ à l'élaboration du projet, dans la mesure où il décrit la problématique, l'objectif général à atteindre, en présentant les besoins de l'application ainsi que la méthodologie de travail.



# 2 Chapitre 2

---

# Analyse et conception

*Ce chapitre aborde la phase d'analyse et de conception du sujet où on va spécifier les besoins fonctionnels de l'application, le diagramme du cas d'utilisation général, les diagrammes de séquence ainsi que le diagramme de classe, tout en s'appuyant sur le langage de modélisation UML*

## 2. Analyse et conception

### 2.1 Analyse

#### 2.1.1 Description du projet

Afin de développer un service de gestion des commandes de repas , nous avons pensé à réaliser une application qui va garantir la gestion moderne et efficace du processus de commande. Dans ce chapitre, nous présentons tout d'abord les acteurs concernés de notre système. Puis, nous entamons l'étude des besoins fonctionnels et non fonctionnels. Ces besoins seront exprimés sous la forme de diagrammes de cas d'utilisation qui permettent de détailler les scénarios possibles que peuvent réaliser les différents acteurs afin de développer une application informatique qui permettra la gestion des commandes produits.

#### 2.1.2 Les besoins

➤ **Gestion des commandes :**

L'ajout des commandes, la manipulation sur les commandes...

➤ **Gestion des Plats :**

L'ajout des produits , suppression , mise-à-jour des produits ...

➤ **Collection et analyse des données :**

Avoir une idée générale sur les repas les plus demandées

➤ **Gestion de livraison :**

Affecter les commandes au livreur choisi au le plus proche.  
Permettre le livreur de voir la localisation de restaurant et le client

### 2.1.3 *Identification des acteurs*

Pour cette application, nous avons identifié plusieurs acteurs et quelques cas d'utilisations répondant aux besoins de chaque acteur.

Acteur	Cas d'utilisation
Restaurant	<ul style="list-style-type: none"> <li>• Gestion des commandes.</li> <li>• Validation/annulation des commandes.</li> <li>• Avoir des statistiques sur les ventes.</li> <li>• Affecter une commande à un livreur disponible.</li> </ul>
Client	<ul style="list-style-type: none"> <li>• Commander des produits.</li> <li>• Consulter les commandes.</li> <li>• Laisser un feedback.</li> <li>• Consulter « Reviews » et « Rating »</li> </ul>
Livreur	<ul style="list-style-type: none"> <li>• Accepter une commande</li> <li>• Avoir la localisation prévue</li> <li>• Avoir la possibilité d'être guidé par l'application vers la localisation prévue.</li> </ul>

### 2.1.4 *Diagramme des cas d'utilisation*

Dans cette section, nous présenterons les besoins de notre système de manière formelle. C'est -à-dire en utilisant le diagramme de cas d'utilisation du langage de modélisation UML. Ce diagramme décrit les différentes interactions entre le système et l'acteur en déterminant les besoins de l'utilisateur et tout ce que doit faire le système pour l'acteur.

La figure , ci-après, représente le diagramme de cas d'utilisation général de notre système :



Figure 1

Diagramme de cas d'utilisation

## **2.1 Conception**

### **2.2.1 Diagrammes de séquence**

Les diagrammes de séquence permettent de représenter des collaborations entre objets selon un point de vue temporel.

#### **2.2.1.1 Diagramme de séquence N°1 : Créer un compte, s'authentifier et commander :**

Le client a la possibilité de commander un repas de menu après s'être authentifié. Celui-ci doit tout d'abord créer un compte pour s'authentifier et puis faire une commande.

Scénario : faire une commande

- Le client s'authentifie en saisissant son login et mot de passe.
- Le client accède au menu et choisit le repas et la quantité.
- Le client remplit le formulaire en saisissant les informations nécessaires (adresse, nom, date , ...).
- La commande est créée.
- Le client peut laisser un feedback si la commande a été livrée.

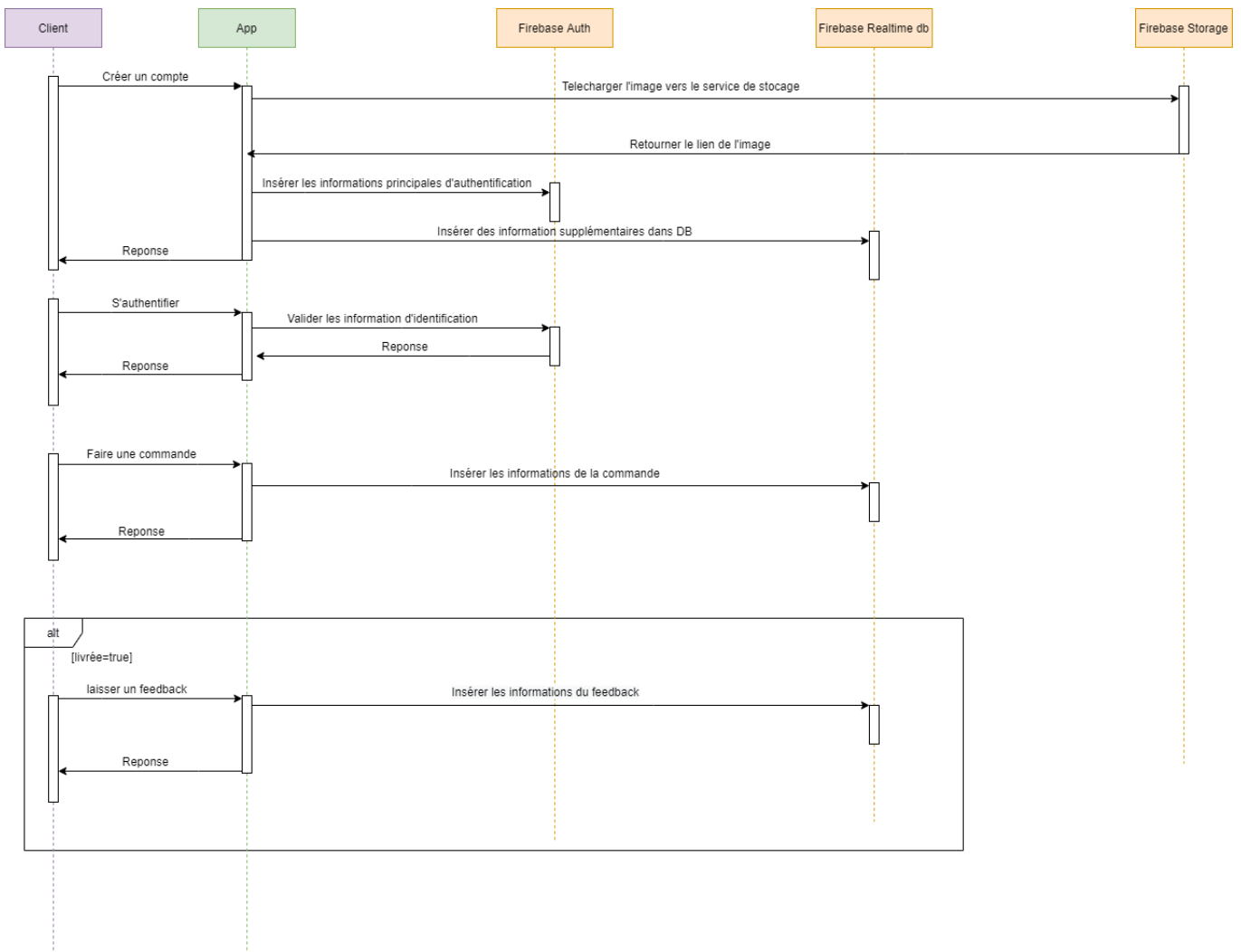


Figure 2 :Diagramme de séquence client

### 2.2.1.2 Diagramme de séquence N°2 : Gérer les commandes (Approuver/Annuler) et les analyser :

L'Administrateur de restaurant a intérêt à approuver les commandes livrées et les affecter au livreur ou les annuler en cas de souci, pour ce faire, il doit tout d'abord s'authentifier, puis consulter l'activité des commandes (Ordres) et marquer la commande et l'affecter au l'un des livreurs disponibles ou l'annuler ou la laisser comme demande en attente.

L'Administrateur a aussi intérêt à visualiser les statistiques des ventes (les quantités vendues pour chaque repas rating et feedback).

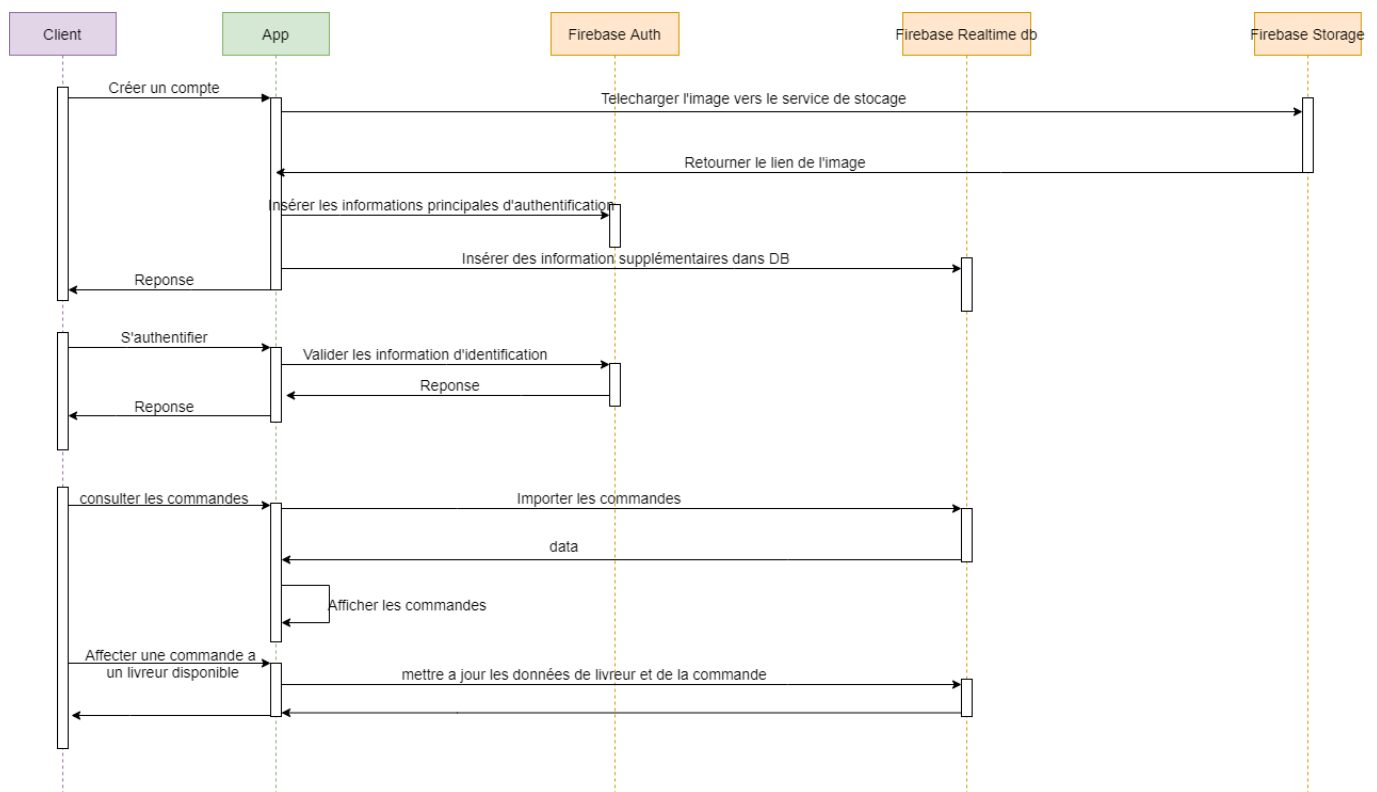


Diagramme de séquence admin

Figure 3 :

### 2.2.1.3 Diagramme de séquence N°3 : Gérer la livraison (Accepter / localiser)

Le livreur a intérêt de livrer la commande affectée a lui. Le livreur n'a pas le choix de refuser la commande qu'il accepte. Le livreur va être capable de visualiser la localisation de restaurant après qu'il accepte la commande et quand il atteint la restaurant il directement avoir l'accès à la localisation du Client.



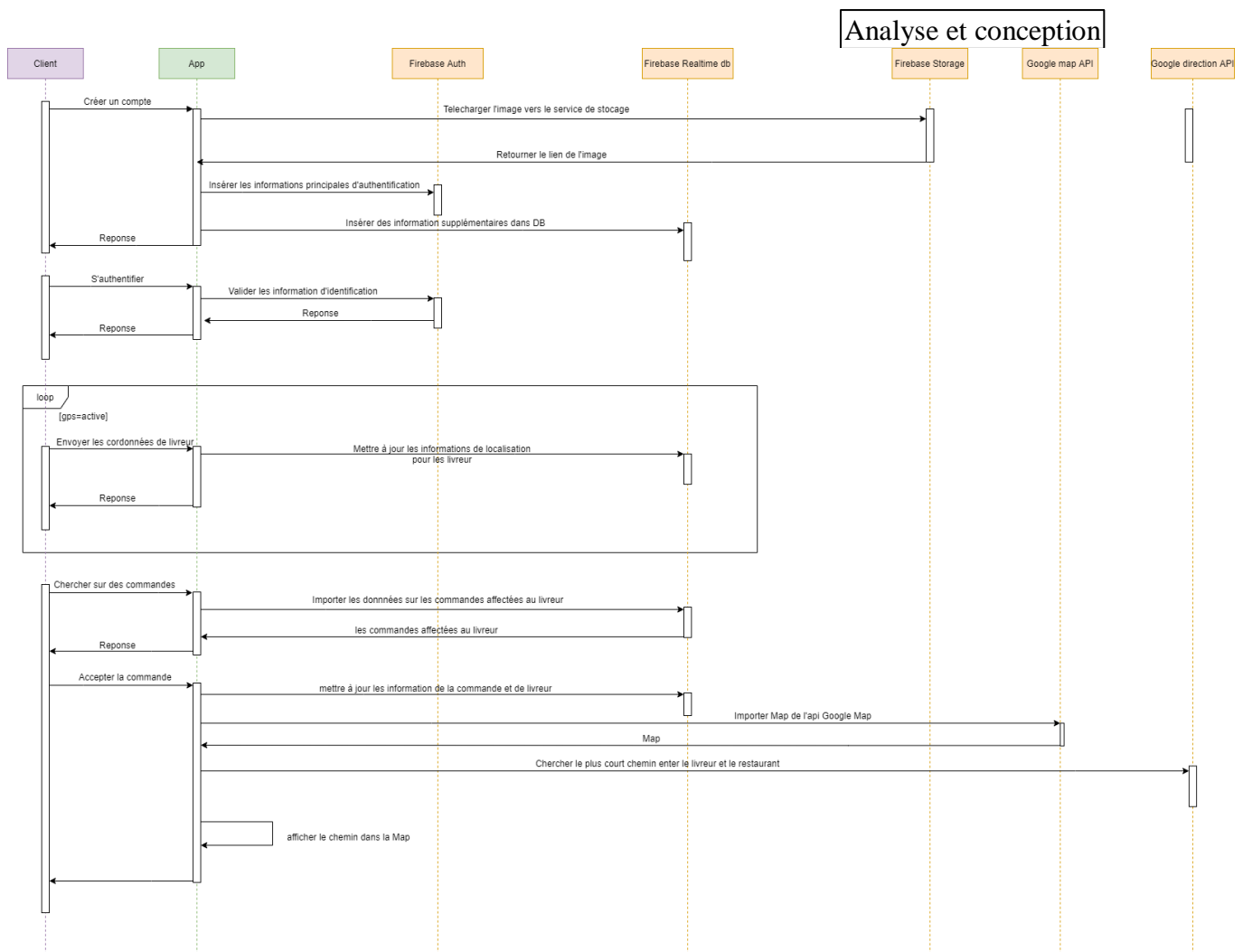


Figure 4 :diagramme de séquence livreur

## 2.2.2 *Diagramme de classe*

Le diagramme de classes est sans doute le diagramme le plus important à représenter pour les méthodes d'analyse et de conception objet. En effet, il permet de spécifier qui intervient à l'intérieur du système.

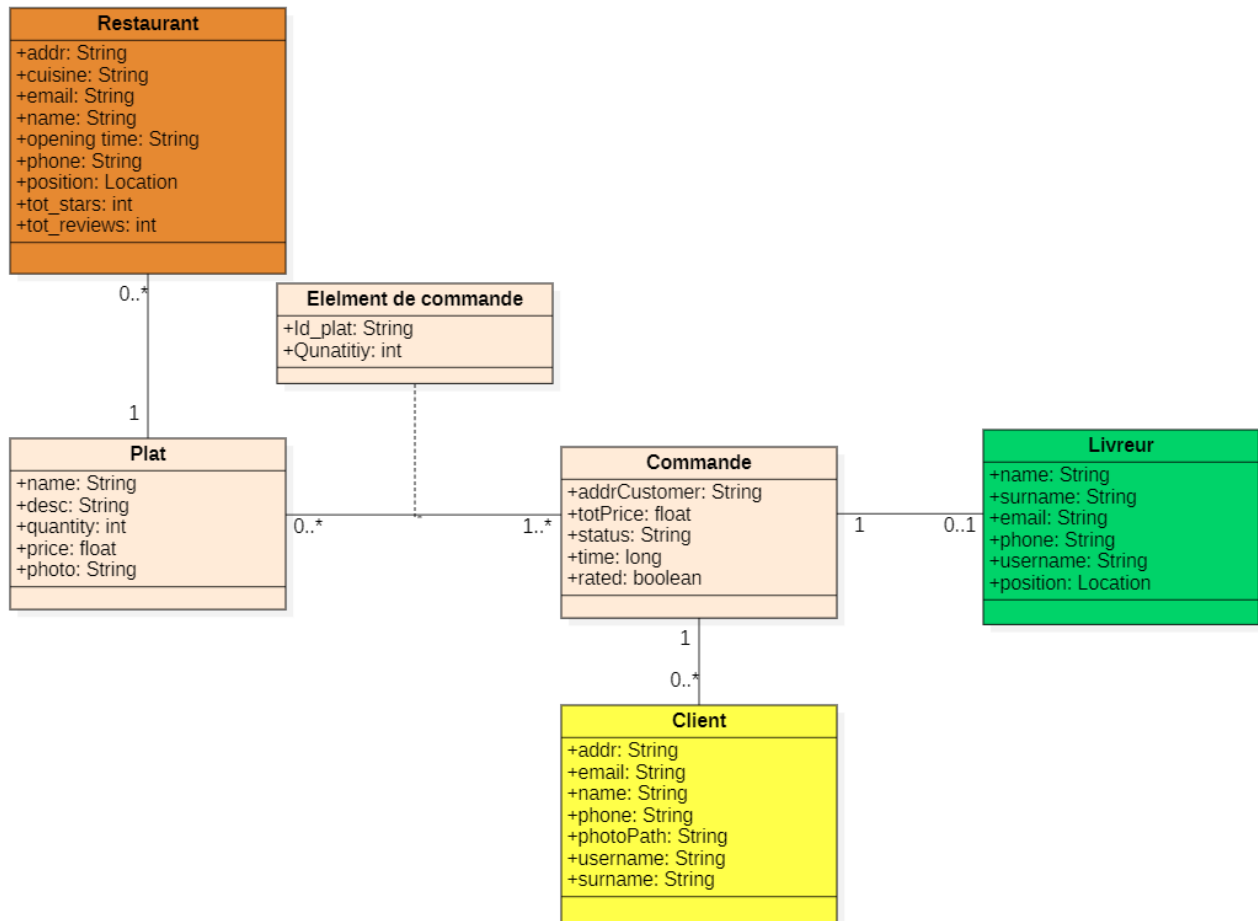


Figure 5 :Diagramme de classe

## Conclusion

Dans ce chapitre, nous avons modélisé notre site web en apportant des réponses à nos questions de modélisation et de conception. En s'appuyant sur l'analyse des besoins de notre application. Nous avons pu la modéliser sous forme de diagrammes de cas d'utilisation puis de séquence, puis concevoir notre application sous forme de diagrammes de classe, reste à créer notre application

## <sup>3</sup> Chapitre 3

---

### **3.1 Intelligence Artificielle et système de recommandation**

*Dans ce chapitre nous allons procéder à l'étude de la partie intelligente qui existe dans notre application à savoir ; le système de recommandation et les algorithmes de recherches de plus court chemin.*

# 1.Introduction

Le développement du mobile a créé un besoin de nouvelles techniques pour aider les utilisateurs à trouver ce qu'ils recherchent mais aussi pour faire savoir qu'une information existe, ces techniques sont appelées les Systèmes de Recommandation (SRs). Une définition des systèmes de recommandation a été donnée dans ce chapitre ainsi que les fondements, concepts de base et notions liées au domaine. Ensuite, en se basant sur la manière dont les recommandations sont formulées, les principaux types de techniques de recommandation sont identifiés, en précisant le Filtrage Collaboratif (FC) et le Filtrage basé Contenu (FBC), ainsi que les problèmes dont souffrent ces systèmes. Une panoplie de mesures de similarité et d'évaluation a été présentée, ainsi que les nouveaux critères d'évaluation pour les systèmes de recommandation.

## 2. Définition et fonctionnement des Systèmes de Recommandation

Les Systèmes de Recommandation (SR) identifient automatiquement les préférences des utilisateurs à travers leurs interactions avec le système, en se basant sur le feedback implicite, le feedback explicite], ou les deux feedbacks, pour leur suggérer des recommandations en utilisant le filtrage d'information. Le filtrage d'information est l'expression utilisée pour décrire une variété de processus dédiés à la fourniture de l'information adéquate aux personnes qui en ont besoin. Son but est de sélectionner et suggérer aux utilisateurs, à partir de larges volumes d'informations générés dynamiquement, les informations jugées pertinentes pour eux. Par conséquent, le filtrage d'information peut être vu aussi comme étant le processus d'élimination de données indésirables sur un flux entrant, plutôt que la recherche de données spécifiques sur ce flux. Le processus du filtrage d'information commence lors de l'utilisation du système par des personnes ayant des désirs et des objectifs qui sont stables, à long terme ou périodiques. Par conséquent, cela conduit à des besoins réguliers en information, qui peuvent évoluer lentement au cours du temps au fur et à mesure que les conditions, les objectifs et les connaissances changent. Le filtrage est basé sur des descriptions des individus et des groupes appelées profils, qui représentent généralement leurs intérêts à long terme, afin de répondre à leurs besoins. De tels intérêts ou profils engagent les

utilisateurs dans un processus passif de recherche d'information, car ils reçoivent les informations à partir du système sans avoir à les rechercher. D'un autre côté, les producteurs des items disponibles en ligne (incluant documents, services ou produits) prennent en charge la distribution de leurs produits dès qu'ils sont générés. Pour accomplir cette tâche, les items sont associés à une représentation de leur contenu appelée profil item, qui est ensuite comparée aux profils des utilisateurs ou des groupes. Par la suite, les utilisateurs consultent les items suggérés et expriment leurs avis sur eux en les évaluant par rapport à leur réponse aux besoins des utilisateurs. Cette évaluation peut mener à la modification des profils des utilisateurs et des domaines de leurs intérêts. Afin de générer des recommandations, un certain nombre d'approches de filtrage ont été présentées dans la littérature, où les plus utilisées sont le Filtrage Collaboratif (CF), et le Filtrage basé sur le Contenu (FBC). Les deux techniques ont des forces et des faiblesses, où l'hybridation entre eux a été rapidement adoptée pour profiter de leurs avantages.

### 3. Historique des Systèmes de Recommandation

La notion du filtrage d'information a vu le jour dans le domaine de la Recherche d'Information (RI) quand la tendance s'est tournée vers la personnalisation des résultats présentés à l'utilisateur à la suite d'une requête. Puis, l'intérêt progressif porté à cette notion de filtrage et l'exploitation des méthodes de fouilles de données (Data Mining) dans ce contexte, a donné naissance aux systèmes de recommandation comme des systèmes voisins, mais indépendants des systèmes de RI du fait que les SRs sont spécialisés beaucoup plus dans le traitement des profils à long terme (les requêtes dans la RI représentent des profils à court terme). Le premier système de recommandation, a été introduit par Goldberg et nommé Tapestry. Deux ans plus tard, les chercheurs du GroupLens<sup>1</sup> ont présenté leur premier Système de Recommandation en parallèle avec le système Ringo. Jusqu'à l'année 1997, les auteurs ont utilisé le terme « Filtrage Collaboratif » au lieu de « Système de Recommandation ». Cette dernière appellation a été stabilisée en cette année par Resnick et Varian. L'année 1997 a reconnu également l'apparition du premier SR hybride 'Fab' créé par Balavonic et Shoham, et qui combine le Filtrage basé sur le Contenu avec le Filtrage Collaboratif. Ensuite, en 2001, la notion du FC basé item a été introduite par Sarwar et al. Et elle a étendu le champ de popularité des SRs du secteur académique vers le secteur commercial. Cette approche a été exploitée plus tard par Linden dans le portail Web d'Amazon.com.

### 4-algorithme A\*.

en intelligence artificielle, l'algorithme de recherche A\* (qui se prononce À étoile, ou A star en anglais) est un algorithme de recherche de chemin dans un graphe entre un nœud initial et un nœud final tous deux donnés 1. En raison de sa simplicité, il est souvent présenté comme exemple typique d'algorithme

de planification, domaine de l'intelligence artificielle. L'algorithme  $A^*$  a été créé pour que la première solution trouvée soit l'une des meilleures, c'est pourquoi il est célèbre dans des applications comme les jeux vidéo privilégiant la vitesse de calcul sur l'exactitude des résultats. Cet algorithme a été proposé pour la première fois par Peter E. Hart, Nils John Nilsson et Bertram Raphael en 1962. Il s'agit d'une extension de l'algorithme de Dijkstra de 1959 (p. 30-31 dans 3).

## 4-a Présentation

L'algorithme  $A^*$  est un algorithme de recherche de chemin dans un graphe entre un nœud initial et un nœud final. Il utilise une évaluation heuristique sur chaque nœud pour estimer le meilleur chemin y passant, et visite ensuite les nœuds par ordre de cette évaluation heuristique. C'est un algorithme simple, ne nécessitant pas de prétraitement, et ne consommant que peu de mémoire.

## 4-b Propriété

Un algorithme de recherche qui garantit de toujours trouver le chemin le plus court à un but s'appelle « algorithme admissible ». Si  $A^*$  utilise une heuristique qui ne surestime jamais la distance (ou plus généralement le coût) du but,  $A^*$  peut être avéré admissible. Une heuristique qui rend  $A^*$  admissible est elle-même appelée « heuristique admissible ».

Si l'évaluation renvoie simplement toujours zéro, qui n'est jamais une surestimation, alors,  $A^*$  exécutera une implémentation possible de l'algorithme de Dijkstra et trouvera toujours la solution optimale. La meilleure heuristique, bien qu'habituellement impraticable pour calculer, est la distance minimale réelle (ou plus généralement le coût réel) au but. Un exemple d'une heuristique admissible pratique est la distance à vol d'oiseau du but sur la carte.

On peut démontrer que  $A^*$  ne considère pas plus de nœuds que tous les autres algorithmes admissibles de recherche, à condition que l'algorithme alternatif n'ait pas une évaluation heuristique plus précise. Dans ce cas,  $A^*$  est l'algorithme informatique le plus efficace garantissant de trouver le chemin le plus court.

## 4-c Description

$A^*$  commence à un nœud choisi. Il applique à ce nœud un « coût » (habituellement zéro pour le nœud initial).  $A^*$  estime ensuite la distance qui sépare ce nœud du but à atteindre. La somme du coût et de l'évaluation représente le coût heuristique assigné au chemin menant à ce nœud. Le nœud est alors ajouté à une file d'attente prioritaire, couramment appelée open list.

L'algorithme retire le premier nœud de la file d'attente prioritaire (en raison du fonctionnement d'une file d'attente, le nœud à l'heuristique la plus basse est retiré en premier). Si la file d'attente est vide, il n'y a aucun chemin du nœud initial au nœud d'arrivée, ce qui interrompt l'algorithme. Si le nœud retenu est le nœud d'arrivée,  $A^*$  reconstruit le chemin complet et s'arrête. Pour cette reconstruction on se sert d'une partie des informations sauvées dans la liste communément appelé closed list décrite plus bas.

Si le nœud n'est pas le nœud d'arrivée, de nouveaux nœuds sont créés pour tous les nœuds contigus admissibles ; la manière exacte de faire dépend du problème à traiter. Pour chaque nœud successif,  $A^*$

calcule son coût et le stocke avec le nœud. Ce coût est calculé à partir de la

somme du coût de son ancêtre et du coût de l'opération pour atteindre ce nouveau nœud.

L'algorithme maintient également la liste de nœuds qui ont été vérifiés, couramment appelée closed list.

Si un nœud nouvellement produit est déjà dans cette liste avec un coût égal ou inférieur, aucune opération n'est faite sur ce nœud ni sur son homologue se trouvant dans la liste.

Après, l'évaluation de la distance du nouveau nœud au nœud d'arrivée est ajoutée au coût pour former l'heuristique du nœud. Ce nœud est alors ajouté à la liste d'attente prioritaire, à moins qu'un nœud identique dans cette liste ne possède déjà une heuristique inférieure ou égale.

Une fois les trois étapes ci-dessus réalisées pour chaque nouveau nœud contigu, le nœud original pris de la file d'attente prioritaire est ajouté à la liste des nœuds vérifiés. Le prochain nœud est alors retiré de la file d'attente prioritaire et le processus recommence.

Les deux structures open list et closed list ne sont pas nécessaires si on peut garantir que le premier chemin produit à n'importe quel nœud est le plus court. Cette situation surgit si l'heuristique est non seulement admissible mais aussi « monotone », signifiant que la différence entre l'heuristique de deux nœuds quelconques reliés ne surestime pas la distance réelle entre ces nœuds. Ce n'est possible que dans de très rares cas.

## 4-d Pseudo-code

```
Structure nœud = {
    x, y: Nombre
    cout, heuristique: Nombre
}
```

```
depart = Nœud(x=_, y=_, cout=0, heuristique=0)
```

```
Fonction compare2Nœuds(n1:Nœud, n2:Nœud)
    si n1.heuristique < n2.heuristique
        retourner 1
    ou si n1.heuristique == n2.heuristique
        retourner 0
    sinon
        retourner -1
```

```
Fonction cheminPlusCourt(g:Graphe, objectif:Nœud, depart:Nœud)
    closedList = File()
    openList = FilePrioritaire(comparateur=compare2Nœuds)
    openList.ajouter(depart)
    tant que openList n'est pas vide
        u = openList.defiler()
        si u.x == objectif.x et u.y == objectif.y
            reconstituerChemin(u)
            terminer le programme
        pour chaque voisin v de u dans g
            si non(v existe dans closedList ou v existe dans openList avec un coût inférieur)
                v.cout = u.cout + 1
                v.heuristique = v.cout + distance([v.x, v.y], [objectif.x, objectif.y])
                openList.ajouter(v)
        closedList.ajouter(u)
    terminer le programme (avec erreur)
```

Figure 6 :pseudo code algorithme A\*

# Chapitre 4

---

## Réalisation

*Dans ce chapitre dédié à l'étude technique et à l'implémentation, nous commençons par définir les outils de développement utilisés pour l'implémentation de notre application web. Ensuite nous passons à la présentation de notre application.*



## 3. Réalisation du projet

### 3.1 Présentation de l'environnement du travail

#### 3.1.1 Architecture de l'application

### Le noyau Linux

On a opté pour l'architecture Android dans notre projet

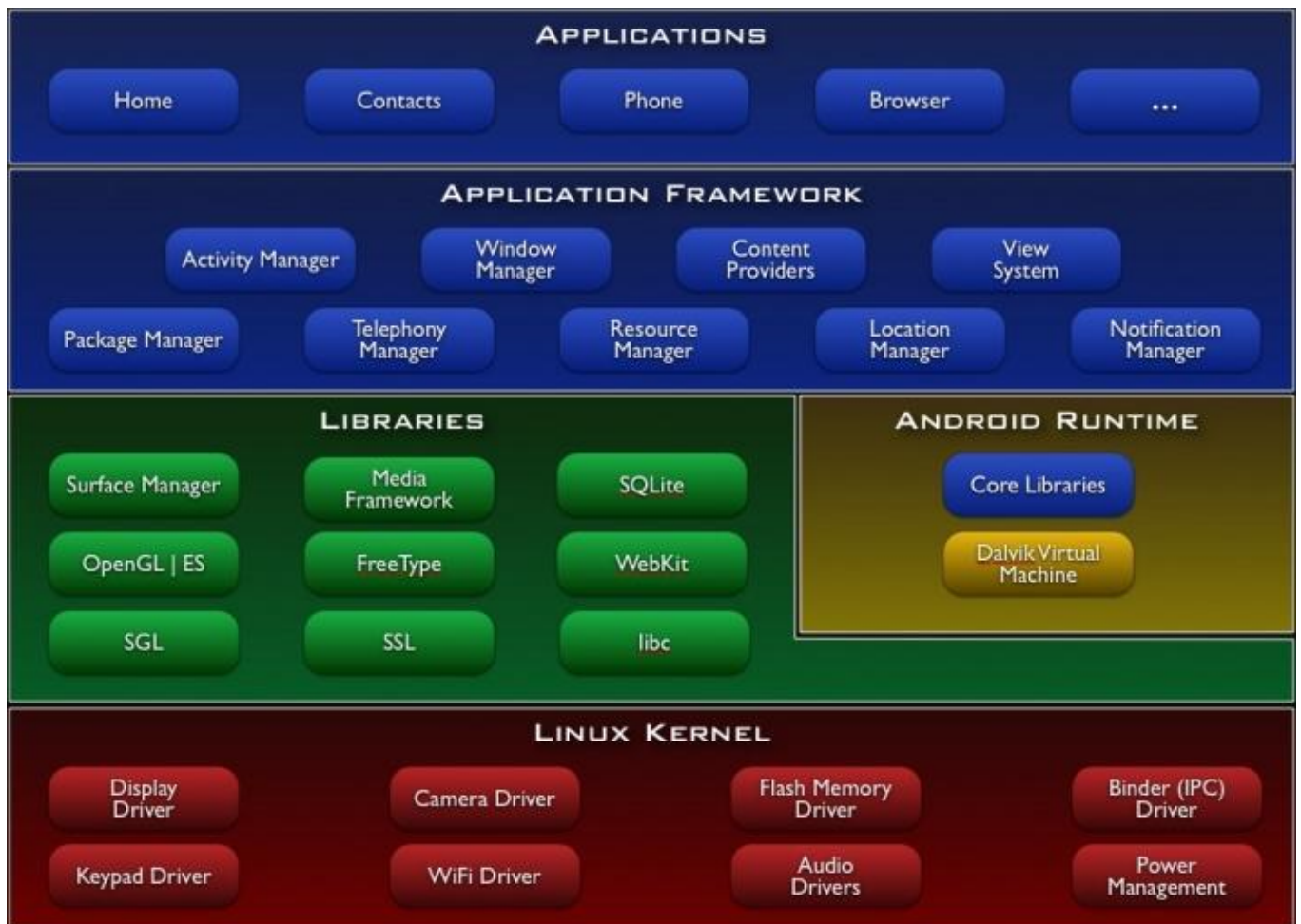


Figure 7 : schémat de l'architecture d'Android

On peut y observer toute une pile de composants qui constituent le système d'exploitation. Le sens de lecture se fait de bas en haut, puisque le composant de plus bas niveau (le plus éloigné des utilisateurs) est le noyau Linux et celui de plus haut niveau (le plus proche des utilisateurs) est constitué par les applications.

Je vous avais déjà dit que le système d'exploitation d'Android se basait sur Linux. Si on veut être plus précis, c'est le noyau (« kernel » en anglais) de Linux qui est utilisé. Le noyau est l'élément du système d'exploitation qui permet de faire le pont entre le matériel et le logiciel. Par exemple, les pilotes WiFi permettent de contrôler la puce WiFi. Quand Android veut activer la puce WiFi, on peut imaginer qu'il utilise la fonction « allumerWifi() », et c'est au constructeur de spécifier le comportement de « allumerWifi() » pour sa puce. On aura donc une fonction unique pour toutes les puces, mais le contenu de la fonction sera unique pour chaque matériel.

La version du noyau utilisée avec Android est une version conçue spécialement pour l'environnement mobile, avec une gestion avancée de la batterie et une gestion particulière de la mémoire. C'est cette couche qui fait en sorte qu'Android soit compatible avec tant de supports différents.

## Le moteur d'exécution d'Android

C'est cette couche qui fait qu'Android n'est pas qu'une simple « implémentation de Linux pour portables ». Elle contient certaines bibliothèques de base du Java accompagnées de bibliothèques spécifiques à Android et la machine virtuelle « Dalvik ».

La figure suivante est un schéma qui indique les étapes nécessaires à la compilation et à l'exécution d'un programme Java standard.

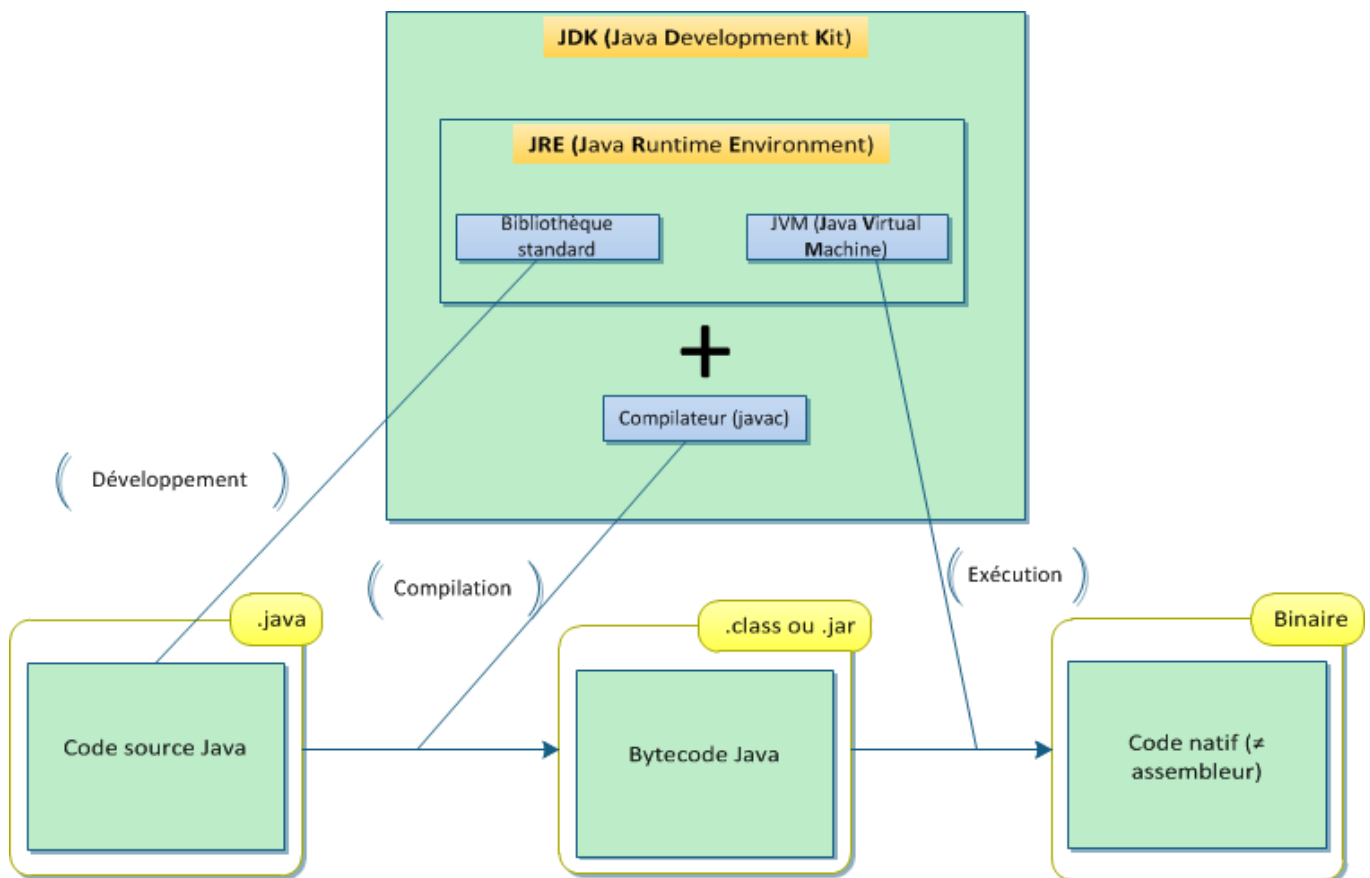


Figure 8 : Architecture Java

Le code est une suite d'instructions que l'on trouve dans un fichier .java qui sera traduit en une autre suite d'instructions dans un autre langage que l'on appelle le « bytecode ». Ce code est contenu dans un fichier.class. Le bytecode est un langage spécial qu'une machine virtuelle Java peut comprendre et interpréter. Les différents fichiers .class sont ensuite regroupés dans un .jar, et c'est ce fichier qui est exécutable. En ce qui concerne Android, la procédure est différente. En fait, ce que vous appelez Java est certainement une variante particulière de Java qui s'appelle « Java SE ». Or, pour développer des applications pour Android, on n'utilise pas vraiment Java SE. Pour ceux qui savent ce qu'est « Java ME », ce n'est pas non plus ce framework que l'on utilise (Java ME est une version spéciale de Java destinée au développement mobile, mais pas pour Android donc).

La version de Java qui permet le développement Android est une version réduite amputée de certaines fonctionnalités qui n'ont rien à faire dans un environnement mobile. Par exemple, la bibliothèque graphique Swing n'est pas supportée, on trouve à la place un système beaucoup plus adapté. Mais Android n'utilise pas une machine virtuelle Java ; une machine virtuelle tout étudiée pour les systèmes embarqués a été développée, et elle s'appelle « Dalvik ». Cette machine virtuelle est optimisée pour mieux gérer les ressources physiques du système. Elle permet par exemple de laisser moins d'empreinte mémoire (la quantité de mémoire allouée à une application pendant son exécution) ou d'utiliser moins de batterie qu'une machine virtuelle Java.

La plus grosse caractéristique de Dalvik est qu'elle permet d'instancier (terme technique qui signifie « créer une occurrence de ». Par exemple, quand vous créez un objet en java, on instancie une classe puisqu'on crée une occurrence de cette classe) un nombre très important d'occurrences de lui-même : chaque programme a sa propre occurrence de Dalvik et elles peuvent vivre sans se perturber les unes les autres. La figure suivante est un schéma qui indique les étapes nécessaires à la compilation et à l'exécution d'un programme Android standard.

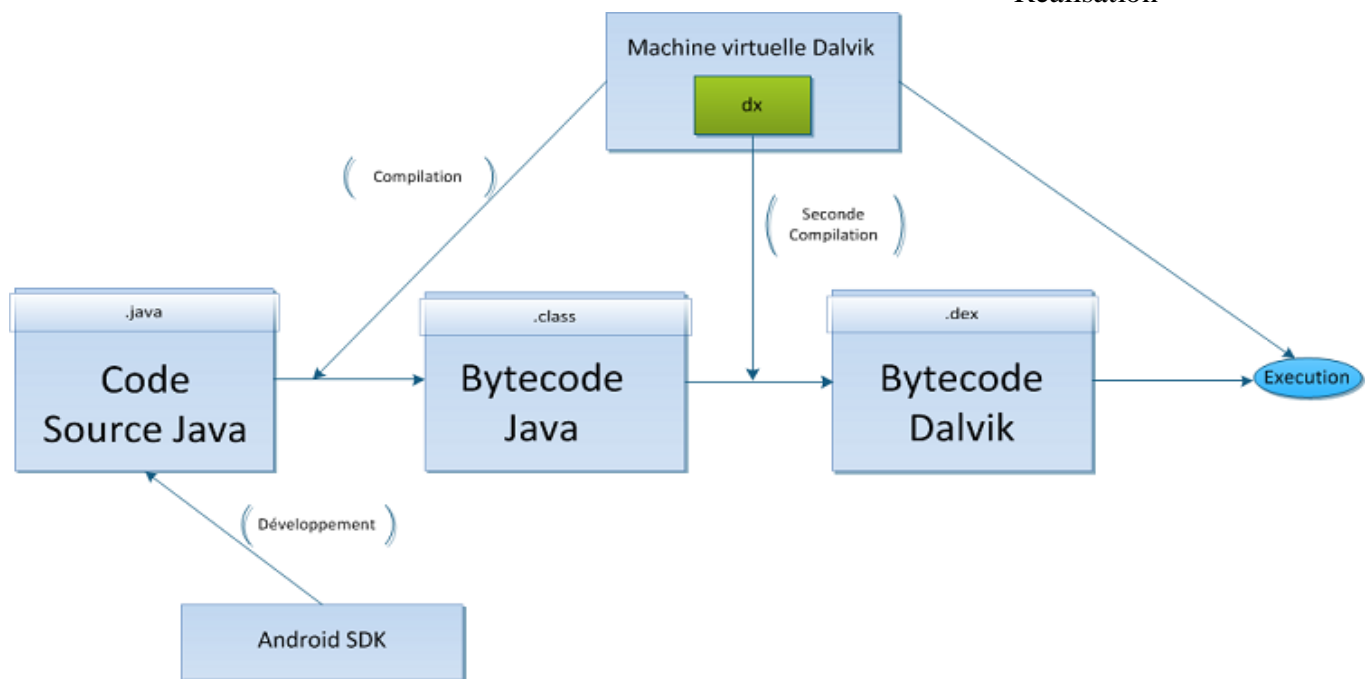


Figure 9 :Dalvik

On voit bien que le code Java est ensuite converti en bytecode Java comme auparavant. Mais souvenez-vous, je vous ai dit que le bytecode Java ne pouvait être lu que par une machine virtuelle Java, mais que Dalvik n'était pas une machine virtuelle Java. Il faut donc procéder à une autre conversion à l'aide d'un programme qui s'appelle « dx » qui s'occupe de traduire les applications de bytecode Java en bytecode Dalvik, qui, lui, est compréhensible par la machine virtuelle

### 3.1.2 *Les technologies déployées :*

## **XML**

Le XML, pour Extensible Markup Language, désigne un langage informatique (ou métalangage pour être plus précis) utilisé, entre autres, dans la conception des sites Web et pour faciliter les échanges d'informations sur Internet. Ce langage de description a pour mission de formaliser des données textuelles. Il s'agit, en quelque sorte, d'une version améliorée du langage HTML avec la création illimitée de nouvelles balises.

Comme le langage HTML, le XML permet la mise en forme de documents via l'utilisation de balises. Développé et standardisé par le World Wide Web Consortium à la fin des années 1990, il répondait à l'objectif de définition d'un langage simple, facile à mettre en application.

Le XML se classe dans la catégorie des langages de description (il n'est ni un langage de programmation, ni un langage de requêtes). Il est donc naturellement utilisé pour décrire des données en s'appuyant sur des balises et des règles personnalisables.

Avec la généralisation de la connexion HTTP, le XML a encore gagné en popularité en devenant une solution habituelle pour créer un nouveau protocole.



*Figure 10 :XML*

# JAVA

*Le strict nécessaire pour créer une application Android*

Java est le langage Android le plus utilisé dans le développement mobile. L'un de ses plus grands avantages est que les logiciels créés avec ce langage peuvent être **facilement installés et exécutés sur différents systèmes d'exploitation**, que ce soit Windows, Mac OS, Linux ou autre. Avec un petit coup de main de Google, qui vous fournit l'environnement de développement **Android Studio**, vous pourrez créer une application android bien plus complexe.



Figure 11 :java

# Android Studio

## *L'indispensable*

Avant de commencer la création de votre application mobile, il vous faudra encore installer et configurer Android Studio. N'attendez pas jusqu'au moment de connaître Java et XML par cœur pour commencer à l'utiliser. Cet environnement de développement **offre des ressources qui vous faciliteront l'apprentissage.**



*Figure 12 :*Android studio

# Firebase

Firebase est un ensemble de services d'hébergement pour n'importe quel type d'application (Android, iOS, Javascript, Node.js, Java, Unity, PHP, C++ ...). Il propose d'héberger en NoSQL et en temps réel des bases de données, du contenu, de l'authentification sociale (Google, Facebook, Twitter et Github), et des notifications, ou encore des services, tel que par exemple un serveur de communication temps réel. Lancé en 2011 sous le nom d'Envolv, par Andrew Lee et par James Templin, le service est racheté par Google en octobre 2014. Il appartient aujourd'hui à la maison mère de Google : Alphabet.

Toute l'implémentation et la gestion serveur de Firebase est à la charge exclusive de la société Alphabet. Les applications qui utilisent Firebase intègrent une bibliothèque qui permet les diverses interactions possibles.



*Figure 13 :*logo firebase

# Places Api

L'API Places est un service qui renvoie des informations sur les lieux à l'aide de requêtes HTTP. Les lieux sont définis dans cette API comme des établissements, des emplacements géographiques ou des points d'intérêt importants.



*Figure 14 :google places api*



## 1. *Implémentation*

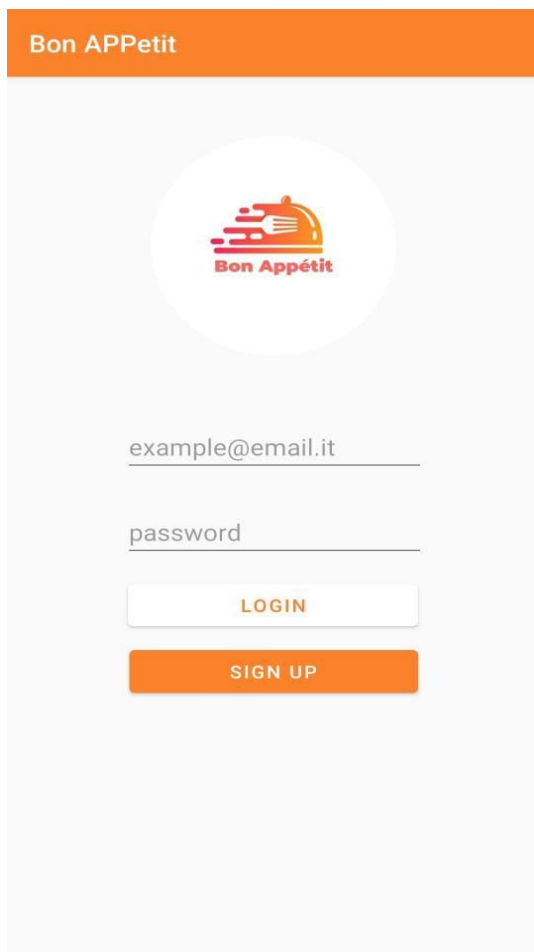
### 3.2.1 *Login and Sign Up*

Bon Appétit est une application Android pour gérer tout le processus de livraison de nourriture. L'application est conçue pour 3 utilisateurs différents :

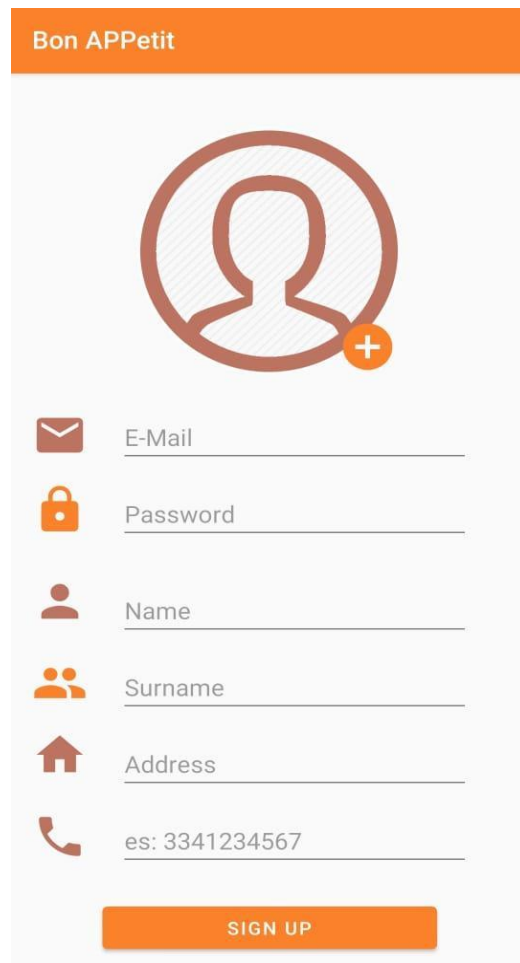
- Client
- Restaurateur
- Livreur

Chacun d'eux doit s'inscrire en remplissant le formulaire affiché.

Lorsque vous appuyez sur le bouton de confirmation, les informations du compte sont enregistrées dans la base de données d'authentification Firebase.



The login screen features an orange header with the text "Bon APPetit". Below the header is a circular logo containing a stylized orange and red fork and knife with the text "Bon Appétit" underneath. The main area is light gray and contains two input fields: the first is labeled "example@email.it" and the second is labeled "password". Below these fields are two buttons: a white button with orange text labeled "LOGIN" and an orange button with white text labeled "SIGN UP".



The sign up screen features an orange header with the text "Bon APPetit". Below the header is a circular profile icon placeholder with a red outline and a small orange circle with a white plus sign in the bottom right corner. The main area is light gray and contains six input fields, each preceded by a red icon: an envelope icon for "E-Mail", a lock icon for "Password", a person icon for "Name", two people icons for "Surname", a house icon for "Address", and a telephone icon for "es: 3341234567". Below these fields is a single orange button with white text labeled "SIGN UP".

Figure 15 :Login and Sign Up

### 3.2.1 **ESPACE CLIENT:** **Comment ca fonctionne:**

- La liste des restaurants s'affiche et est basée sur Firebase RecyclerView ; Les données sont extraites de la base de données au moyen d'une requête.
- L'utilisateur peut sélectionner un critère de filtre pour la recherche en fonction des favoris, de la cuisine ou du classement des avis
- Il n'est pas permis de commander dans un restaurant fermé.

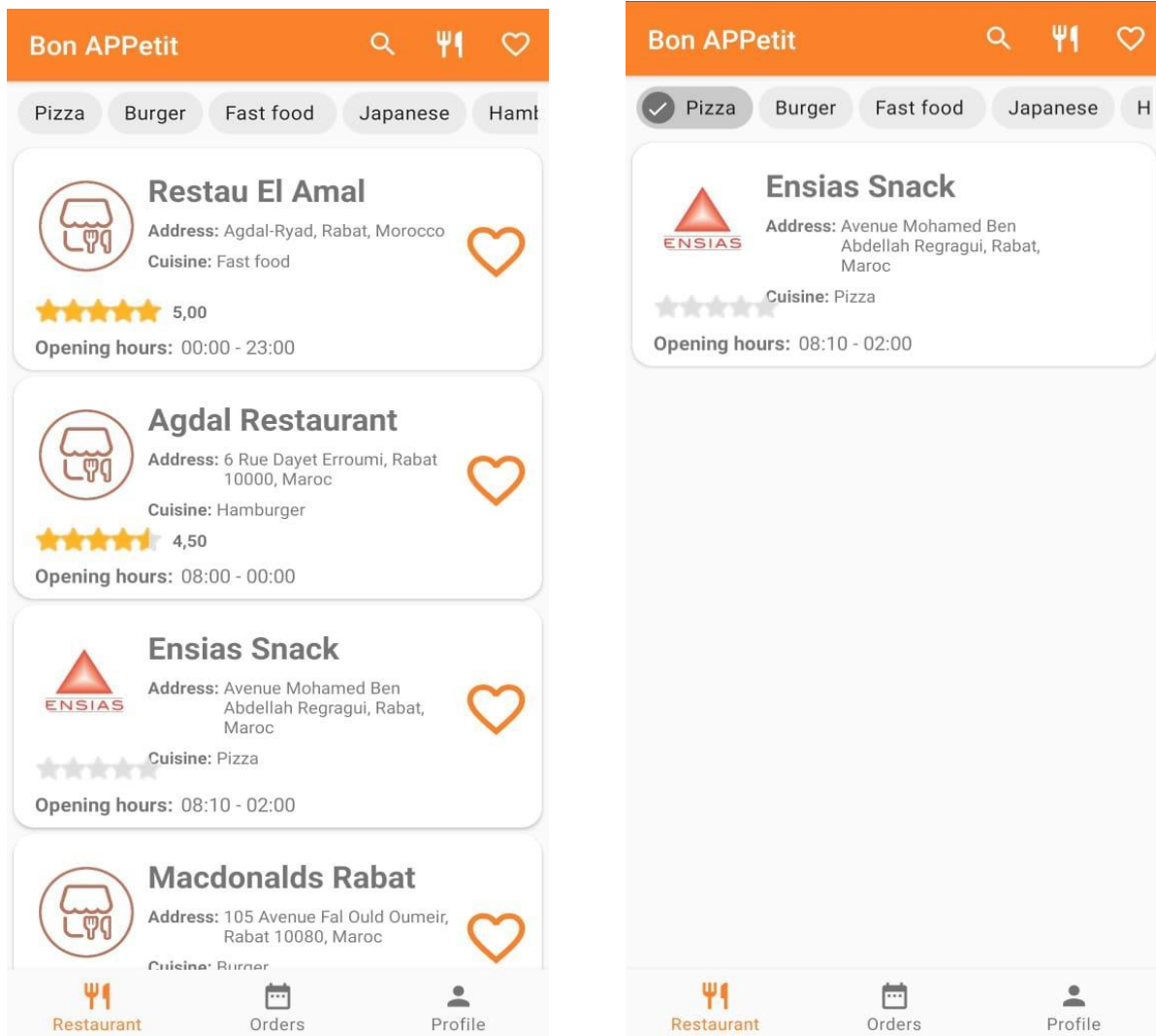


Figure 16 :espace client

### *Faire une commande*

Une fois un restaurant sélectionné, il est possible de consulter sa carte et ses avis. L'utilisateur peut sélectionner la quantité souhaitée de plats à commander, puis la confirmer en sélectionnant l'heure de livraison.

Les fragments « commande » et « révision » sont gérés au moyen d'un adaptateur de page. Une fois la commande confirmée, une structure de données associée est enregistrée dans la base de données Firebase

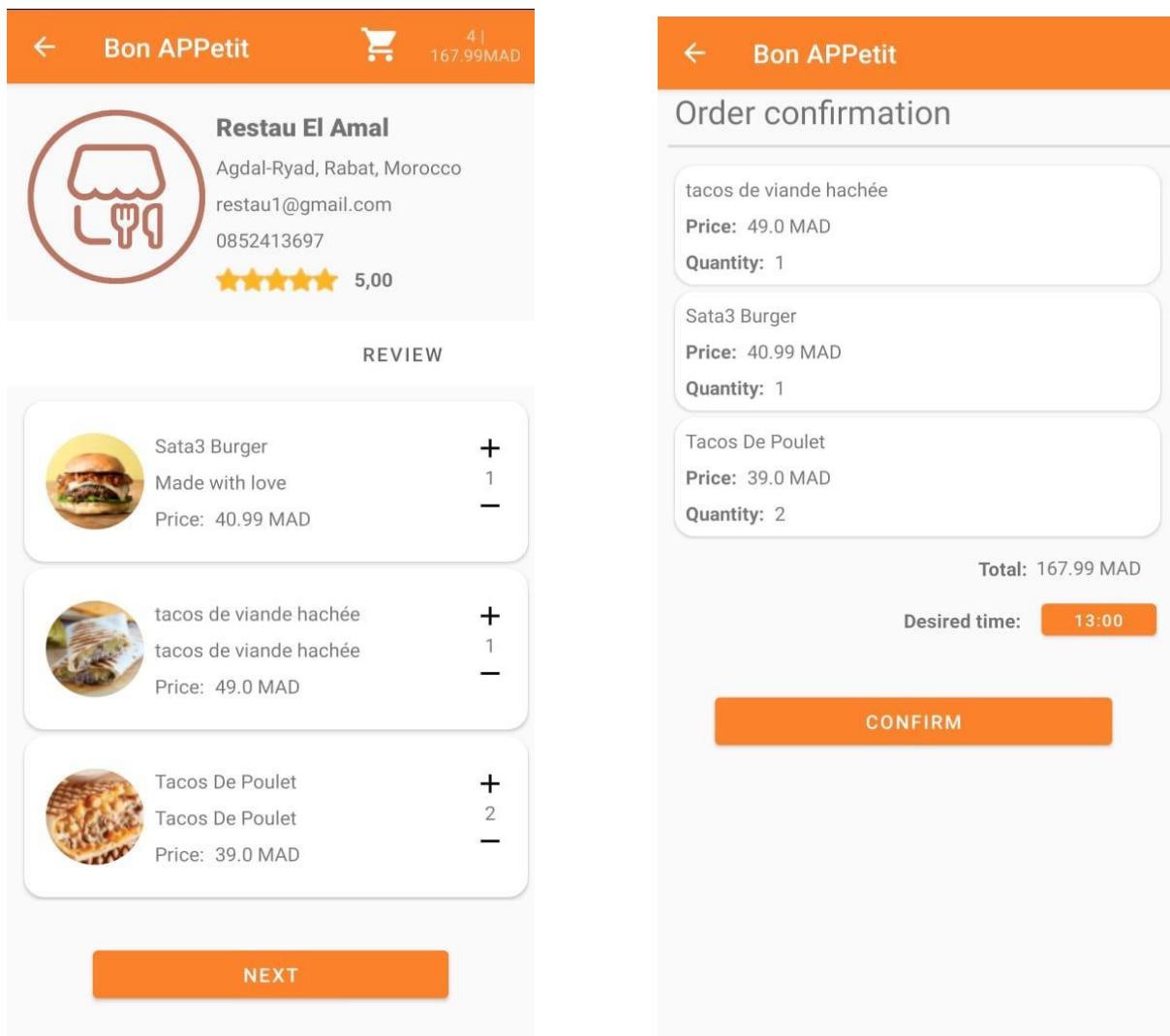


Figure 17 :faire une commande

## Historique des commandes

Un historique des commandes est disponible sur l'application client afin que l'utilisateur puisse vérifier les détails de chaque commande et la revoir une fois qu'elle est livrée.

L'utilisateur peut également suivre sa commande en vérifiant l'état :

- Commande envoyée
- Commande refusée
- En train de Livraison
- Commande livrée

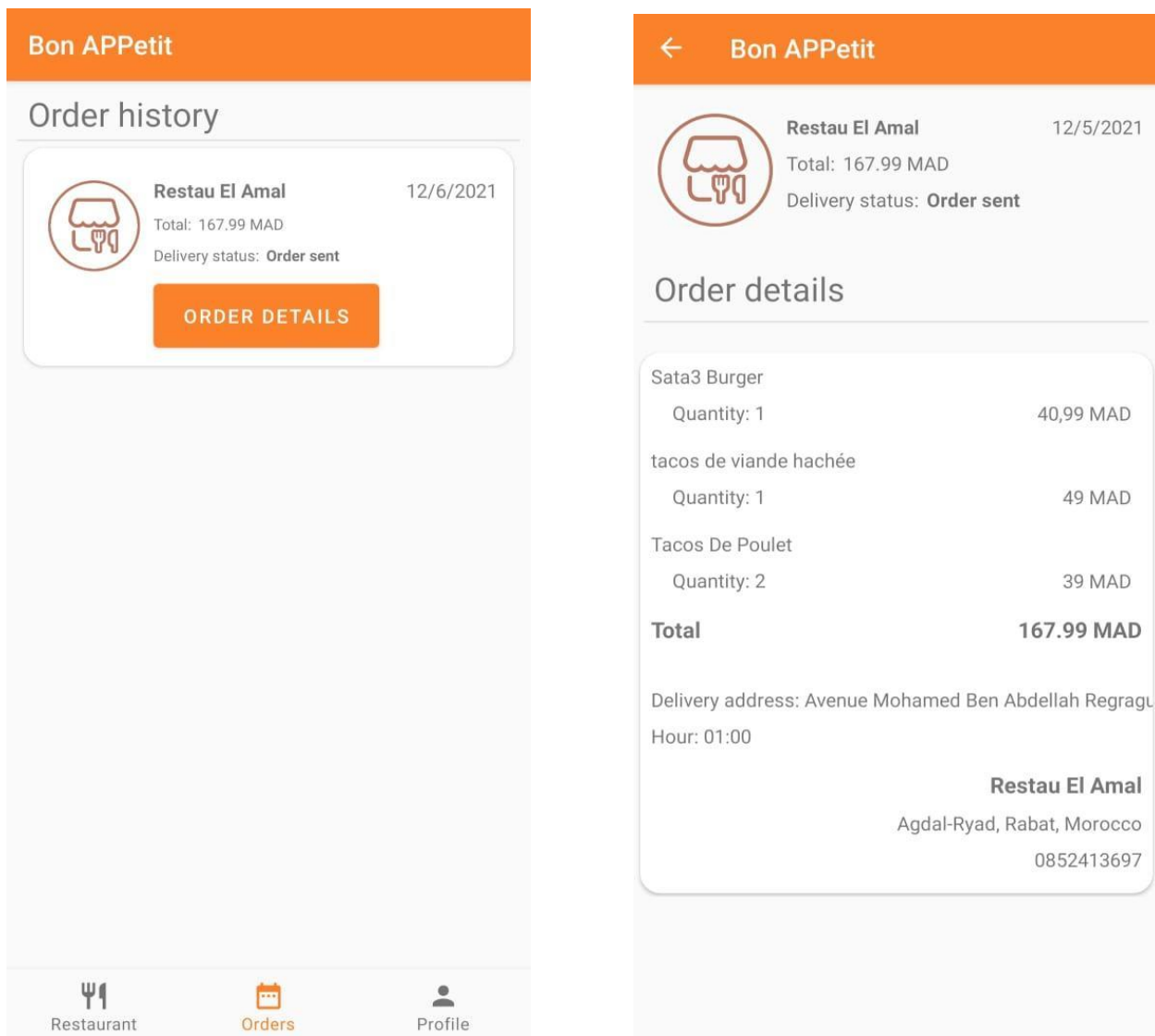


Figure 18 :historique des commandes.

### 3.2.3 Espace des restaurant et snacks

Le profil du restaurateur reprend toutes les informations insérées lors de la phase d'inscription. Les données peuvent être modifiées en cliquant sur le menu Options.

Le fragment de profil comprend également une section d'évaluation qui comprend tous les avis donnés par les clients.

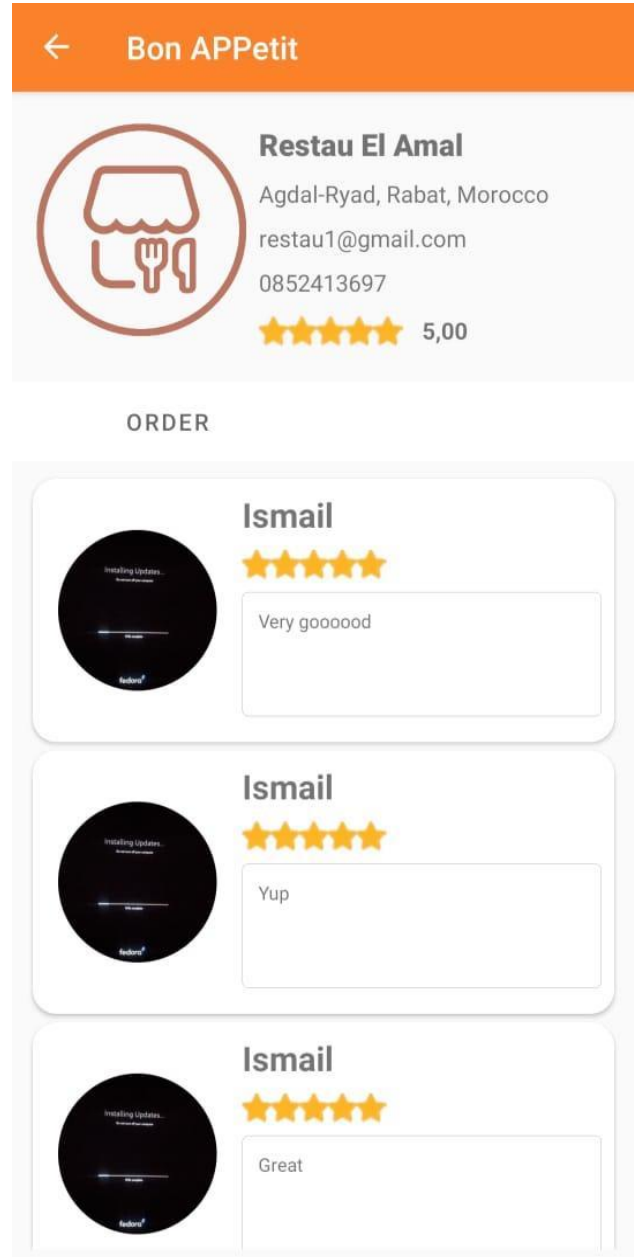
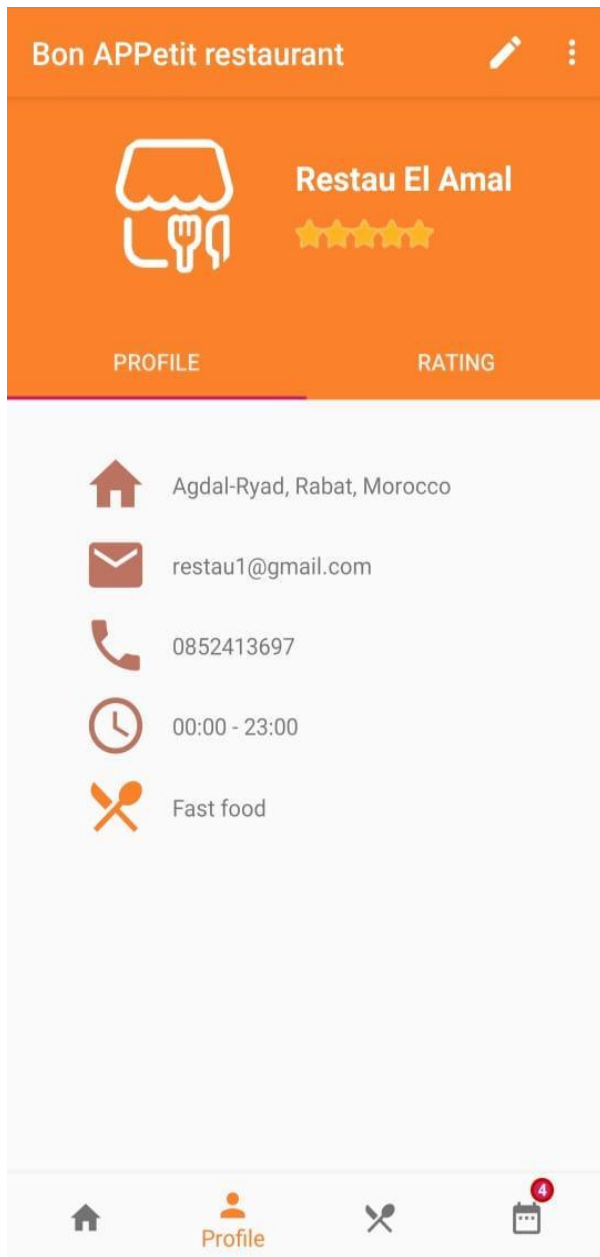


Figure 19 :espace restaurant

### Ajout d'un nouvel article au menu de l'offre quotidienne

L'opération d'ajout se fait sur une activité dédiée où l'utilisateur insère toutes les informations liées au plat. Une fois le bouton de confirmation enfoncé, le nouvel élément est lié à RecyclerView qui affiche toutes les offres de menu.

Chaque élément peut être mis à jour ou supprimé de la liste au moyen de deux boutons dédiés.

**Bon APPetit restaurant**

**Daily Offers**

**Sata3 Burger**  
Made with love  
**40.99 MAD**  
Quantity: 999

**tacos de viande hachée**  
tacos de viande hachée  
**49.0 MAD**  
Quantity: 23

**Tacos De Poulet**  
Tacos De Poulet  
**39.0 MAD**  
Quantity: 31

**Form Fields:**

Name: \_\_\_\_\_

Description: \_\_\_\_\_

Price: \_\_\_\_\_ MAD

Available quantity: \_\_\_\_\_ 0

**CONFIRM**

**Bottom Navigation Bar:** Home, Profile, Daily Offer, Notifications (4)

Figure 20 : Gérer les commandes.

## Cycle de vie de la commande

1-Le restaurateur reçoit la commande par notification sur le BottomNavigationView

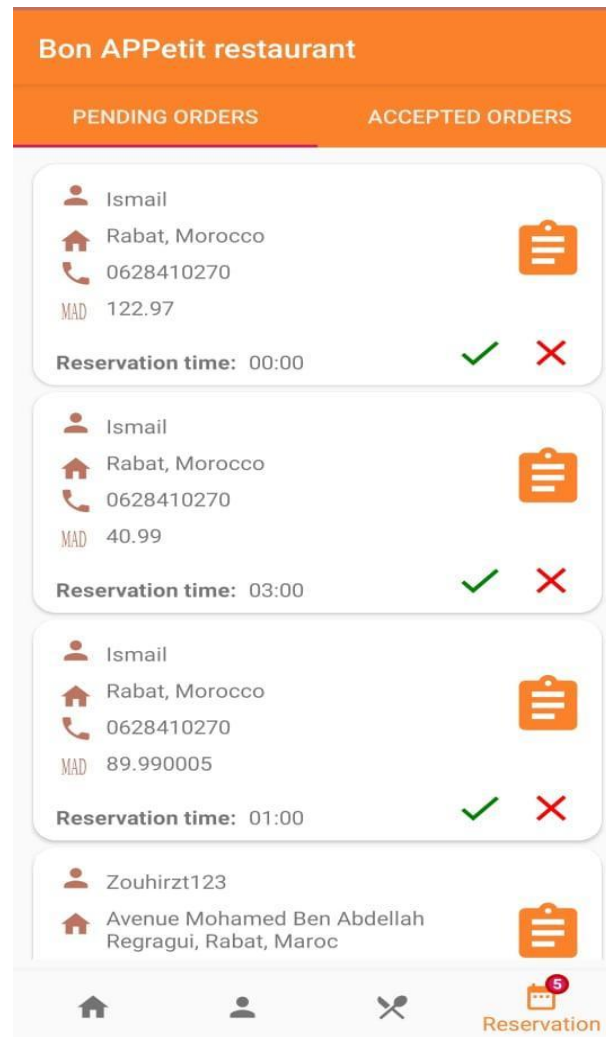


Figure 21 :recoit de notification

2-L'administrateur peut lui-même sélectionner le coureur sur la carte.



Figure 22:voir la géolocalisation d'un livreur



3-De manière alternative par une liste de coureurs ou en choisissant automatiquement le plus proche

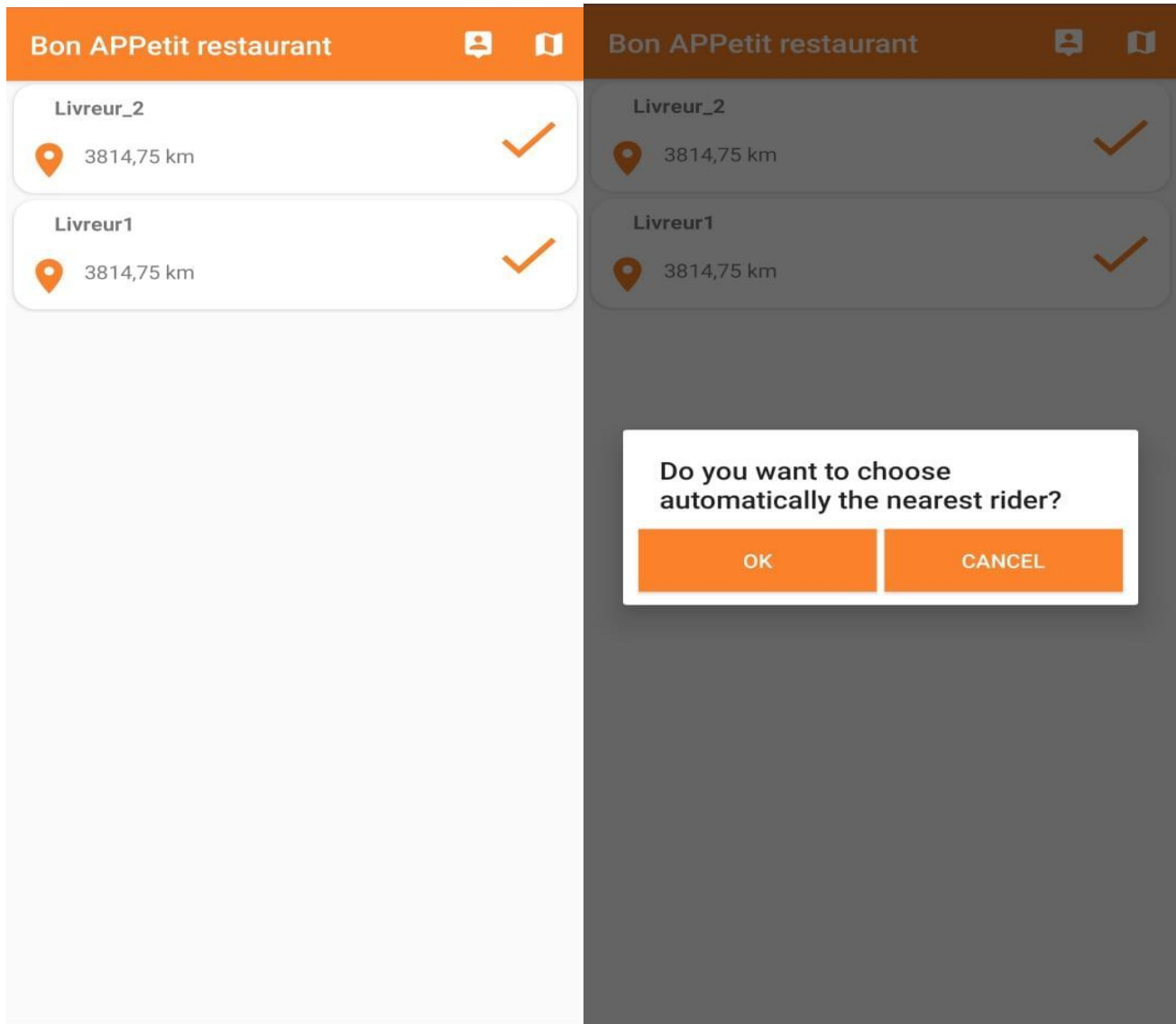


Figure 23 :affectation d'un livreur

### 3.2.4 Espace livreur

- Lorsqu'un ordre est attribué à un livreur, une notification s'affiche sur son application.
- Dans le fragment d'inversion, le livreur peut voir l'adresse du restaurant et du client.
- Le chemin est indiqué sur la carte. De plus, il est mis à jour lorsque le livreur atteint le restaurant et le parcours du client est affiché.
- Lorsque la commande est livrée, le livreur le notifie en cliquant sur le bouton correspondant.

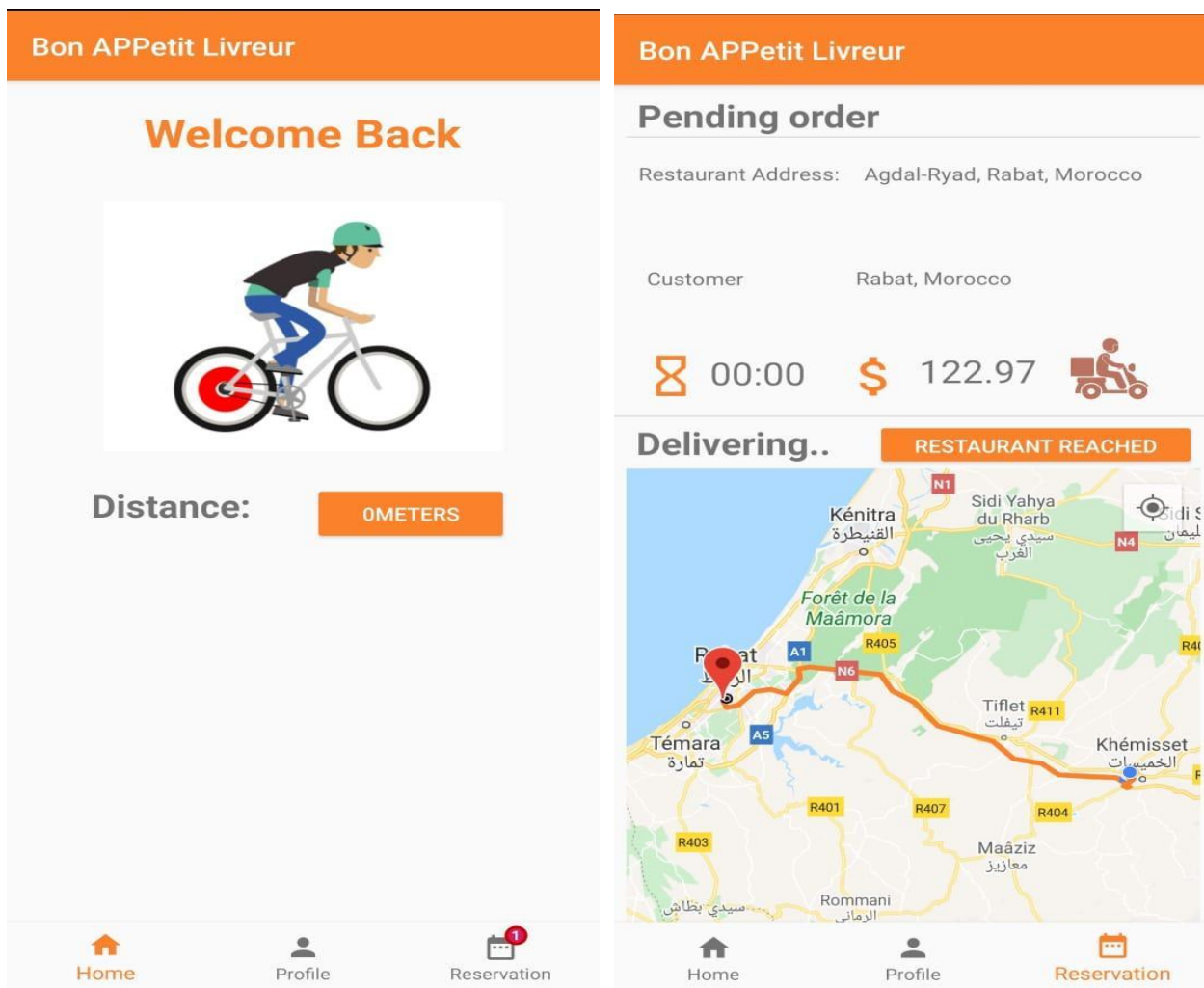


Figure 24 : espace livreur

### **Conclusion :**

Avec cette dernière étape, nous avons conclu la réalisation du projet en mettant en œuvre les études analytique et conceptuelle déjà présentées dans le premier chapitre. Nous avons présenté les outils déployés pour son développement ainsi que les différentes interfaces.



## Conclusion générale

En substance, nous vous avons présenté tout au long de ce rapport la démarche que nous avons suivie pour mettre au point cette application : en commençant par la présentation du sujet et du cahier de charge, pour ensuite aborder la partie conception de l'application et enfin présenter la réalisation finale de l'application.

L'objectif de notre travail était de concevoir et implémenter l'application de gestion des commandes au sein d'un restaurant qui permet à l'utilisateur d'échanger des données informatives et alternatives selon un ensemble de critères. Tout au long de ce projet nous avons pu apprendre à maîtriser java dans Android studio en se basant sur l'architecture Android.

Ce projet nous a permis d'appliquer les connaissances qui nous ont été inculquées au cours de cette année dès l'analyse et la conception d'une application jusqu'à son implémentation. Il nous a permis aussi de développer nos aptitudes de travail en équipe, de communication, de résolution des problèmes qui surgissent lors du déroulement d'un projet.

Cependant, nous pouvons dire que notre système de gestion d'un restaurant reste toujours en besoin d'amélioration et d'enrichissement par certaines techniques et fonctionnalités. En perspectives, nous comptons terminer la réalisation de la solution, en complétant les développements par :

- Prévoir un moyen pour récupérer le mot de passe en cas d'oubli.
- Sécuriser les données des clients.
- Ajouter d'autres fonctionnalités.

# Bibliographie

## □ Livres

- XML AND JAVA complete guide

## □ Logiciels

- android studio: <https://developer.android.com/studio>
- Google chrome

## □ Webographie

- android openclassroom: <https://openclassrooms.com/en/courses/626954-creez-votre-application-web-avec-java-ee>
- StackOverflow : <https://stackoverflow.com/>

