REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'enseignement supérieur et de la recherche scientifique
Université Mouloud Mammeri de Tizi-Ouzou
Faculté du génie électrique et d'informatique.
Le département d'informatique.

Mémoire de fin d'études

En vue de l'obtention du diplôme
De MASTER 2 en Ingénierie des systèmes d'information.

Thème

Conception et réalisation d'une application mobile multiplateforme pour la gestion des ventes

Proposé par :

MS CONTACT INFORMATIQUE Tizi-Ouzou

Dirigé par :

Mme F. BOUARAB

Réalisé par :

Mlle BENYOUCEF Lynda.

Mlle BOUAMARA Nedjma.

Membres de jury :

Mr HAMMACHE Arezki

Mme SEGHIRI

Promotion : 2019/2020

Remerciements

Nous tenons à témoigner notre reconnaissance à DIEU tout puissant, qui nous a aidé et bénit par sa volonté durant toute cette période.

Notre profonde gratitude et sincères remerciements vont à notre promoteur **Mme F.BOUARAB** pour sa présence continuelle, son encouragement tout au long de ce travail.

Le personnel de l'entreprise MS CONATCT qui nous ont été d'une très grande aide en répondant à chaque appel au secours.

Nous adressons nos remerciements aux membres du jury, devant qui, nous avons l'honneur d'exposer notre travail, et qui ont pris la peine de lire ce mémoire pour juger son contenu.

Nous réservons ici une place particulière pour remercier vivement tous ceux qui, d'une manière ou d'une autre, nous ont aidés et encouragés à la réalisation de ce modeste travail.

Nedjma et Lynda.

Dédicaces

Je dédie ce modeste travail à :

A ma mère la personne la plus chère au monde,
A mon adorable soeur,
A tous mes amis et mes collègues,
Et spécialement à Lounis, pour sa précieuse aide et son encouragement qui a contribué à la réussite de ce projet.

BENYOUCEF Lynda

Je dédie ce modeste travail à :

À mes chers parents qui m'ont fourni un soutien et une confiance sans faille. Que dieu vous protège et vous garde pour moi.
À mes précieuses sœurs.
À mes adorables amies, pour votre fidélité et votre soutien.
À tous mes amis avec lesquels j'ai partagé des moments de joie et de bonheur.

BOUAMARA Nedjma

Résumé

Le but de ce travail est de réaliser une application mobile fiable et aisée pour la gestion des ventes, qui est une extension du logiciel existant ILLUTRAE, munie des fonctionnalités nécessaires et adéquates aux besoins de la société de service MS CONTACT où s'est déroulé notre période de stage.

Pour mettre en œuvre notre solution, nous avons utilisé un processus de développement appelé Processus Unifié (UP), qui se base sur l'UML comme langage de modélisation conçu pour fournir une méthode normalisée pour la conception et la construction des documents nécessaires au bon développement. Cette conception est mise en œuvre sous l'environnement de développement (Visual studio code et Xampp, Xcode et Visual C++) , Ionic, AngularJs et PHP comme langages de programmation, un système de gestion de base de données MySQL et le serveur apache.

Abstract

The goal of this work is to achieve a reliable and easy mobile application for sales management , which is an extension of the existing ILUTRADE software , equipped with the necessary functionalities and adequate to the needs of the service company MSCONTACT where our period of internship took place .

To implement our solution, we used a development process called Unified Process (PU) , which is based on UML as a modeling language designed to provide a standardized method for designing and building the documents necessary for the proper development.
This design is implemented under the development environment (Visual studio code and Xampp , Xcode et Visual C++) , Ionic , AngularJs and PHP as programming languages , a MySQL database management system and apache server .

Sommaire

Chapitre II : L'étude de l'existant

Chapitre III : Analyse et Conception

Chapitre IV : Développement d'un prototype

Liste des tableaux

Liste des figures

Introduction générale

Les technologies de l'information et de la communication ont été la révolution la plus importante qui a marqué ces dernières années, grâce aux applications mobiles qui sont devenues une partie indispensable des smartphones et le nombre d'utilisateurs de ces derniers est en constante augmentation.

Les technologies mobiles offrent à l'homme la possibilité de contrôler et d'assurer ses gestions, sans être contraint de se présenter à son entreprise tout le temps. Mais la diversité des systèmes d'exploitation qui utilisent des technologies différentes ont poussé les développeurs de mettre en place des méthodes afin de parvenir à développer pour plusieurs plateformes à la fois, d'où le développement multiplateforme.

Le travail présenté dans ce mémoire concerne la mise en œuvre d'une application multiplateforme pour assurer la gestion des ventes qui est une extension mobile au logiciel existant de la société de service MS CONTACT de Tizi-Ouzou. Il s'agit donc de concevoir et de réaliser une solution permettant de :

- ✓ Gérer les ventes.
- ✓ Mise à jour des clients.
- ✓ Consulter la disponibilité des produits dans le stock ,etc.

Ce mémoire comporte quatre chapitres :

- ✓ Le premier chapitre expose les différents systèmes d'exploitation, les langages utilisés pour chaque système, ainsi que l'étude détaillée des systèmes multiplateformes.
- ✓ Le second est consacré à la présentation de l'organisme d'accueil et l'étude du logiciel existant.
- ✓ Le troisième présente l'analyse et la conception de notre application.
- ✓ Enfin le quatrième chapitre décrit l'implémentation et la mise en œuvre de notre solution.

Chapitre I

Développement multiplateforme

Introduction

Au fil des années l'environnement de travail décentralisé et mobile est devenu cruciale dans le monde en évolution rapide. Cela montre l'importance des applications mobiles. Ces derniers peuvent aider à opérer au-delà des frontières traditionnelles de l'environnement de bureau.

Ainsi, pour qu'une application mobile soit compatible avec les différents systèmes d'exploitation, il est nécessaire de faire appel à un développement hybride permettant la bonne prise en charge de l'application sur différents supports (smartphone, tablette …) mais aussi pour les différentes plateformes (Apple iOS, Android, Windows Mobile …).

De ce fait, ce chapitre aura pour objectif de présenter quelques notions des systèmes d'exploitation avec les langages propres à chaque plateforme (développement native), puis on se focalisera sur le développement multiplateforme, pour finir avec le choix du framework pour le développement de notre application.

I.1. Applications mobiles

Une application mobile est un logiciel applicatif transportable et autonome, développé pour être installé sur un appareil électronique mobile. Elle est identifiée par un ou plusieurs programmes téléchargeables de façon gratuite ou payante depuis un magasin d'applications "Application Store ", permettant d'accéder à un contenu homogène et exécutable à partir du système d'exploitation du Smartphone. Les applications mobiles permettent en général un accès plus pratique, rapide et efficace à des sites en version mobile ou web **[1].**

I.2. Les systèmes d'exploitation mobiles

Un système d'exploitation ou OS Operating System en anglais est un super logiciel qui permet de gérer toutes les autres applications (mise en marche, arrêt, allocation des ressources mémoires) et la communication avec le support physique **[2].**

Un système d'exploitation mobile est un système conçu pour fonctionner sur un appareil mobile ; c'est un ensemble de programmes, responsable de la liaison entre les ressources matérielles de l'appareil et ses applications logicielles.

Il se concentre sur la gestion de la connectivité sans fil et celles de différents types d'interface.

Les principaux systèmes sont : Android, iOS, BlackBerry OS, Symbian OS, Windows Mobile.

I.2.1. Android OS (Operating System)

C'est un OS fondé sur un noyau Linux, conçu par Googleet l'Alliance. Celui-ci équipe aujourd'hui des appareils très variés : Smartphones, tablettes, montres digitales, autoradios …. Android est gratuit pour les constructeurs d'appareils souhaitant l'utiliser et partiellement open-source. Il a intégré plusieurs services de Google pour accéder rapidement aux services d'internet comme Gmail, YouTube, Google Talk, Google Calendar et Google Maps **[3].**

Figure I.1: L'icône représentant le système Android.

I.2.2. iPhone (iOS)

Est le système d'exploitation mobile développé par Apple pour l'iPhone . Reconnu pour sa fluidité et son inituivité ; C'est le système le plus abouti à ce jour .

iOS se caractérise par une interface peu chargée, qui permet l'exécution de tâches rapidement pour les utilisateurs réguliers, mais qui laisse peu de place à la personnalisation. IOS est dérivé de Mac OS X et donc d'Unix **[4].**

Figure I.2: L'icône représentant le système iOS.

I.2.3. BlackBerry OS

Système du fabricant canadien RIM (Research In Motion), Il fut le premier à proposer la notification instantanée d'emails, en mode push. Il optimise également l'utilisation mobile en compressant les pages web, ainsi que les pièces jointes des mails. Le mail est donc le point fort des BlackBerry qui a fait son succès auprès des cadres et dirigeants **[4].**

Figure I.3: Exemplaire d'un téléphone utilisant BlackBerry OS.

I.2.4. Symbian OS

Il est créé par Nokia en 1998 en compagnie de Panosanic , Psiron , Ericsson et Motorola . Nokia fut ensuite le principal utilisateur de Symbian pendant de nombreuses années pour équiper ses téléphones mobiles et Smartphones et racheta tous les droits du consortium Symbian Ltd en 2008. Il

offre une plateforme flexible, néanmoins depuis quelques années , la part de marché de Symbian diminue à cause de concurrence avec d'autres plateformes **[4] .**

Figure I.4: Exemplaire d'un téléphone sous Symbian OS.

I.2.5. Windows Mobile

Anciennement appelé Windows Phone, est le système d'exploitation conçu par Microsoft en 2010. Il est présent sur les smartphones et tablettes tactiles de Windows. Proposant une interface simple et épurée ainsi que de multiples possibilités de personnalisation, il permet d'arriver rapidement à l'exécution d'une tâche ce qui a particulièrement séduit les utilisateurs. Il est basé sur un noyau Windows CE.

I.3. Les outils de développement mobile

Les applications mobiles sont développées sur des ordinateurs ; le langage utilisé dépend du système sous lequel l'application sera exécutée et chaque plateforme possède ses propres outils de développement.

Ci-après nous allons présenter les outils de développement pour les trois principales plateformes mobiles : Android, IOS et Windows Mobile.

a. **Android** : Pour développer une application mobile sur ce support, le langage de programmation utilisé sera Java. Le kit de développement proposé par Google :

SDK (Software Development Kit) autrement dit les outils de développement utilisés seront le SDK Android. On peut également utiliser le langage C++ avec le NDK (Native Development Kit)

b. **IOS** : C'est la plateforme Apple. Le langage utilisé sera l'Objective-c ou Swift. Pour développer des applications sur iOS il faut avoir un Mac, installer l'environnement de développement XCode et avoir le Framework CocoaTouch.

c. **Windows Mobile** : Le langage utilisé est généralement le C# accompagné des technologies web (HTML, CSS, JavaScript) via la technologie WinJS et l'environnement de développement utilisé est Microsoft Visual Studio.

I.4. Les solutions pour développer une application mobile

Actuellement, Il existe trois solutions possibles pour le développement des applications pour mobile. Le choix de ses trois solutions se fait en prenant compte du système d'exploitation sur lequel nous voulons utiliser l'application et des caractéristiques que doit avoir cette dernière.

Figure I.5: Les solutions pour le développement mobile .

Il est possible de regrouper ces solutions dans les cas suivants : Le développement natif, web et hybride.

I.4.1. Le développement natif

Le développement natif consiste à développer des applications avec les outils et les langages propres à chaque système d'exploitation. Le fait d'utiliser cette méthode et développer une application native permet de bénéficier des éléments et toutes les fonctionnalités de l'API (GPS, appareil photo, etc.) pour un développement plus rapide et plus propre.

Ces types d'applications sont téléchargés depuis un magasin d'applications mobiles (store), puis installées sur le périphérique, elles peuvent aussi fonctionner en mode hors ligne (sans connexion internet).

Le problème rencontré dans cette méthode est que nous devons développer un code pour chaque plateforme par exemple, pour les deux plateformes Android et iOS, il faut avoir deux codes sources complètement différents, par conséquent, le temps et le prix de développement augmentent et les mises à jour seront pénibles à faire.

I.4.2. Le développement web (HTML)

La deuxième solution consiste à développer des applications en utilisant les langages web classiques comme : HTML, CSS et JavaScript. Ces applications fonctionnent comme des sites web, et dans ce cas-là elles ne sont pas concernées par les problèmes de compatibilité, par contre nous ne pouvons pas profiter de toute la panoplie de fonctionnalités offertes par l'API du mobile.

I.4.3. Le développement hybride

Certaines communautés ont très bien compris le problème de compatibilité et proposent d'autres méthodes qui permettent, avec un seul code, de lancer des applications sur les différentes plateformes d'exécution, "Write once, run everyWhere". Une application hybride est développée à partir des langages web (HTML, JavaScript, CSS…), et qui s'appuie sur des technologies natives mobiles afin de pouvoir utiliser des fonctionnalités du smartphone telles que : caméra, GPS, etc…, ces types d'applications sont aussi téléchargées depuis les magasins d'applications en ligne et installées sur le mobile. La solution la plus connue étant Titanium Platform pour décrire une application en utilisant du JavaScript et leur API, une autre nouvelle solution qui vient d'apparaître est l'utilisation du framework Ionic.

L'avantage majeur de ces solutions est de coder avec un langage connu qui est le JavaScript pour avoir un code unique compatible avec plusieurs plateformes, par exemple, l'utilisation du framework Ionic permet d'avoir un code source d'applications mobiles exécutables dans trois systèmes d'exploitation : Android, iOS et Windows Mobile. (Voir figure **I-6**)

Figure I.6: Le principe de la solution hybride .

I.5. Choix de la solution

Le choix d'une solution est une phase très importante qui précède le développement de l'application. Pour développer une application qui doit utiliser le maximum de fonctionnalités, il est plus intéressant de se former dans le langage souhaité et de la développer en utilisant les outils propres au système choisi.

Par contre, le développement HTML permet de développer une application assez rapidement et qui fonctionne de la même manière dans les différentes plateformes d'exécution, dans ce cas, il faut faire attention aux caractéristiques des mobiles, tandis que la dernière solution "le développement hybride" est utilisable pour avoir un code multi- plateformes, c'est une solution très à la mode vu le nombre important des plateformes qui existent actuellement sur le marché.

Le tableau (I.1) représente les fonctionnalités de chaque solution :

Fonctionnalité \ Type d'application	Application Web	Application native	Application hybride
Accès aux fonctionnalités natives (GPS, etc.)	Non (Accès limité)	Oui	Oui
Téléchargement depuis les Mobile stores	Non	Oui	Oui
Codage 1 seule fois et exécution sur plusieurs plateformes	Oui	Non	Oui
Accès hors ligne	Non	Oui	Oui
Portabilité du code	Oui	Non	Oui
Coût de développement et de maintenance	Réduit	Élevé	Moyen

Tableau I.1: Fonctionnalités de chaque solution du développement mobile .

I.6. Les frameworks hybrides

Pour répondre à notre problématique, nous utilisons dans ce travail la solution “Hybride”, qui nous offre juste ce qu’il nous faut pour développer une application à la fois pas très lourde et assez riche en fonctionnalités et multi-plateformes.

La solution hybride garantit le multiplateforme ce qui permet un gain de temps non négligeable et une baisse des coûts importante. Généralement, ces solutions sont développées en JavaScript4.

Parmi les frameworks des applications mobiles hybrides : Apache Cordova, React-native et Ionic qu’on détaillera ci-dessous.

I.6.1. Apache Cordova

Cordova (Phone Gap) est un projet réalisé avec Adobe qui a fourni le code source à la fondation Apache, un framework open source, utilisé pour le développement des applications mobiles hybrides, il utilise les standards du web ou en d’autres termes, les technologies HTML, CSS et Javascript.

Figure I.7: Logo Apache CORDOVA (Phone Gap) .

Ce framework a connu une grande évolution, et actuellement, il supporte un nombre important de systèmes d'exploitation mobiles tels que : Android, iOS, Windows Mobile, BlackBerry ou encore Ubuntu. Nous pouvons penser à Cordova comme un conteneur pour connecter notre application Web aux fonctionnalités mobiles natives. L'idée est simple : compilez un shell à l'aide d'un module "WebView" chargé de l'application. C'est un peu comme regarder l'application dans un navigateur très léger.

Tous les points cités jusqu'à maintenant sont considérés comme des avantages pour Cordova, mais comme tous les frameworks, il a un inconvénient majeur, selon [5], l'inconvénient de Cordova est que toute la logique métier doit être développée en Javascript. Cordova est donc adaptée à des projets mobiles avec une faible logique métier. Malgré cet inconvénient et aussi la faille de sécurité annoncée par IBM Security X-Force Research en 2014, Cordova reste parmi les meilleures solutions de développement cross- Platform dans le marché.

I.6.2. React-native

React-native est un framework mobile hybride développé par Facebook en 2015 cible les applications natives [6]. Il est très récent sur le marché des outils multi-plateformes et basé sur la bibliothèque JavaScript React de Facebook. En utilisant JSX, il est possible d'écrire des composants React Native qui seront compilés en véritables composants natifs. Il s'appuie sur des connecteurs pour interagir avec des fonctionnalités natives telles que la caméra.

Dans un premier temps, react-native est utilisé pour créer des applications mobiles compatibles avec les deux plateformes Android et iOS, d'aujourd'hui il intègre la troisième plateforme qui est Windows Mobile.

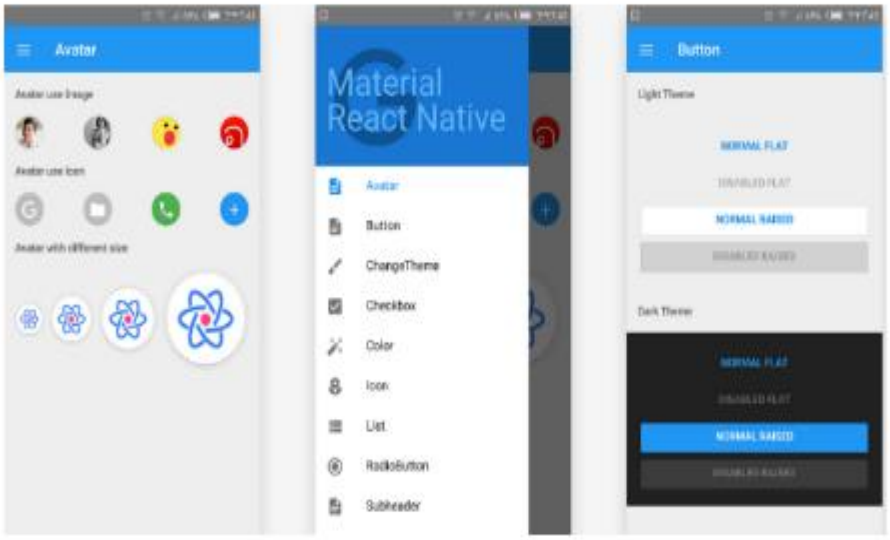


Figure I.8: Exemple d'applications avec les composants react-native .

I.6.3. Ionic

Ionic est un framework d'applications mobiles hybride populaire open-source, créé en 2013. Il se base sur AngularJS10 pour la partie application web du framework et sur Cordova pour la partie applications natives.

Figure I.9: Les bases du framework Ionic .

Ce framework open source est utilisé pour développer des applications mobiles hybrides rapidement et facilement et qui seront déployées sur des sites web ou des applications mobiles avec différentes plateformes telles que : Android, iOS et Windows Mobile.

Figure I.10: Ionic et le multi-plateforme.

Ionic fournit un ensemble de composants d'interface à utiliser dans des applications hybrides. Ces composants sont similaires à celles fournies par les SDK natif comme : des onglets et des boîtes de dialogue.

Une installation de système de plugin Cordova est nécessaire pour avoir des fonctionnalités matérielles supplémentaires. Ainsi il fournit une API utilisée par exemple pour demander des coordonnées GPS ou accéder à la caméra **[7].**

Les plugins se composent toujours de deux parties. La première est une partie JavaScript qui s'exécute dans le WebView qui expose une API agréable à l'application hybride. La deuxième partie est spécifique à la plate-forme et écrite dans la langue maternelle de la plate- forme, par exemple Java pour Android et Objective-C pour iOS. Les API natives sont contrôlées par cette deuxième partie.

La figure **(I.11)** illustre l'architecture de plugin de haut niveau.

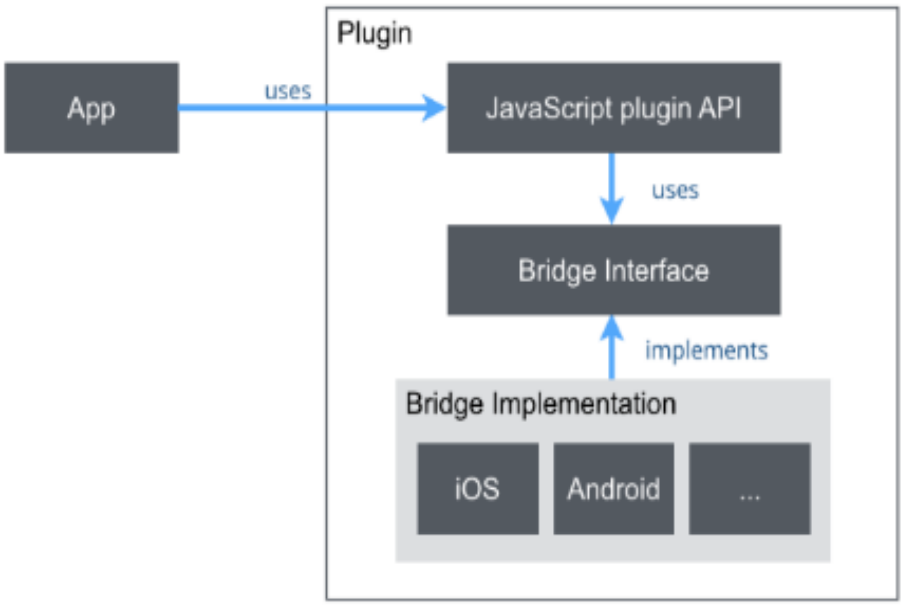


Figure I.11: Architecture de plugin Cordova de haut niveau.

Ionic n'est pas seulement un ensemble de composants d'interface utilisateur, mais aussi un ensemble d'outils de développement et un écosystème pour la construction rapide d'applications mobiles hybrides**.** Il en est à sa version 5.

I.7. Choix du framework

Il existe plusieurs solutions hybrides multi-plateformes, comme Cordova, Ionic et react-native, la question qui se pose toujours dans le développement mobile est : quel framework choisir pour notre future application ?

Pour répondre à cette question nous faisons une comparaison entre ces différents frameworks.

Tandis que React Native génère des composants natifs, Ionic génère des pages Web qui imitent uniquement le comportement natif. Cela signifie que Ionic est plus portable car le même code basé sur le Web s'exécute de manière cohérente sur les différentes plateformes web et mobiles. Avec React Native, les composants de l'interface utilisateur doivent être écrits spécifiquement pour les plateformes sur lesquelles ils seront exécutés. C'est pour cela que React Native ne peut pas se vanter de suivre le mantra «write once, use everywhere».

Ionic est basé sur des technologies populaires telles que Angular, CSS, HTLM, permettant d'être utilisé rapidement par les développeurs.

Ce framework fournit de nombreux composants et plugins. Il est également possible d'utiliser un kit de développement d'applications riche en fonctionnalités : bibliothèque de blocs front-end, des composants UI, plugins vous permettant d'accès à l'appareil photo, etc.

De plus, il offre un cadre multi-plateforme pour l'avenir qui dépasse l'api iOS et Android, car il couvrira selon les prévisions même les téléviseurs intelligents, consoles Xbox, périphériques Windows et la liste reste longue, il s'appuie sur une communauté active et il a une interface ergonomique [8].

Pour terminer, notre choix se tourne vers Ionic qui est une technologie très prometteuse avec un énorme potentiel, et qui a déjà été appréciée par des géants comme Baidu, Discovery, Instagram et d'autres.

Conclusion

Ce chapitre nous a permis d'aborder les différents systèmes d'exploitation conçus pour fonctionner sur les appareils mobiles et leurs outils, Ensuite on a cité les types d'applications mobiles, les différents frameworks hybrides et la différence entre eux.

Dans le chapitre qui suit, nous allons faire notre étude de l'existant qui consiste à étudier l'application existante sur bureau afin de pouvoir en extraire la problématique et trouver des solutions mobiles pour ce fait.

Chapitre II

Etude de l'existant

Introduction

De nos jours, chaque entreprise recherche des méthodes d'évaluation plus adaptées et plus efficaces pour gagner plus d'argent en moins de temps possible, et une bonne gestion d'entreprise passe indéniablement par une bonne gestion des ventes

Ainsi, dans une entreprise il est nécessaire de passer par un bon logiciel informatique. Il en existe beaucoup qui sont adaptés aux différents utilisateurs et aux différentes entreprises.

Dans ce chapitre, nous allons présenter l'application existante qui est un logiciel de gestion commerciale, où nous verrons la notion de gestion des ventes, tout en détaillant les fonctionnalités du logiciel pour parvenir à spécifier les besoins à traiter tout au long de ce travail.

II.1. La gestion de vente

La bonne gestion des ventes dans une entreprise se traduit par la bonne gestion des forces de vente et terrain, c'est un élément essentiel car c'est delà que l'entreprise tire sa principale source de rentabilité, tirer profits des nouvelles technologies pour l'optimisation des cycles de ventes et la réduction des coûts est actuellement l'objectif de toutes les entreprises commerciales.

II.1.1. Processus de vente

Pour effectuer une commande de vente, c'est-à-dire une vente, il faut avoir en données de bases : les articles, les clients et les fournisseurs. Ci-dessous est décrit le processus de commande de vente à connaître afin d'effectuer, dans l'ordre, une commande de vente. Nous avons schématisé le processus de vente comme le présente la figure (**II.1**).

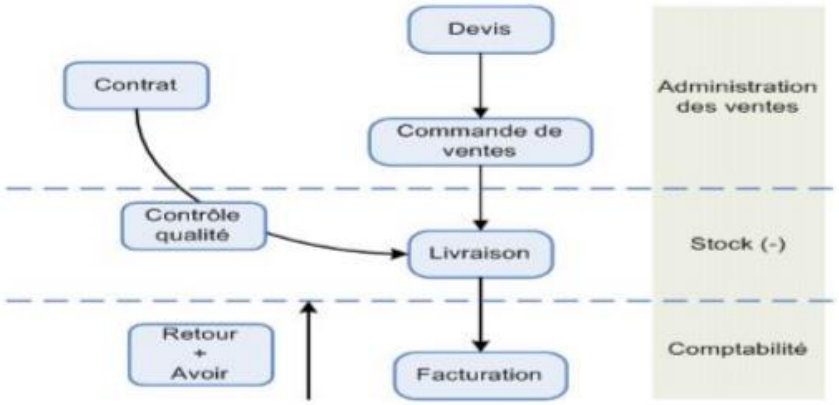


Figure II.1: Processus de vente .

II.1.2. Les fonctions de vente

- ✓ Gestion des commandes fermées
- ✓ Gestion des livraisons directes
- ✓ Gestion des transporteurs
- ✓ Gestion des factures et acompte
- ✓ Gestion des retours et avoirs
- ✓ Déclarations douanières
- ✓ Multi adresses (donneur d'ordre, livraison et facturation)
- ✓ Gestion des devis
- ✓ Gestion des commandes ouvertes…

II.1.3. Mission de la force de vente

La principale tâche pour la force de vente est de vendre. Vendre c'est un art, et posséder son art c'est maîtriser l'ensemble des techniques permettant l'accomplissement d'une œuvre, vendre requiert des connaissances approfondies comme dans tout autre métier **[9].**

Elle possède en tout quatre tâches qui sont expliqués ci-dessous.

- L'avant-vente (la prospection)

« Avant d'être un client effectif, un client est d'abord potentiel et il est appelé : PROSPECT ».

La prospection consiste à rechercher de nouveaux clients potentiels vers lesquels un effort de vente sera effectué par la suite.

- La vente :

« Toute conversation orale entreprise avec un ou plusieurs acheteurs potentiels, dans le but de présenter un produit, répondre à des objections et conclure une affaire » **[10] .**

- Le suivi des ventes :

Enfin, le vendeur a pour rôle de suivre son client, lors de cette étape, le vendeur va dresser le bilan de l'entretien, qui consiste à confronter le résultat obtenu aux objectifs préalablement fixés, ainsi qu'à identifier les causes d'un écart éventuel, en analysant ses points forts et faibles et tout cela pour assurer le suivi.

II.2. Présentation de l'entreprise d'accueil MS CONTACT

MS CONTACT est une société de Services, d'Ingénierie et de Distribution Informatique composée de professionnels en Informatique. Fondée en 1996, MS Contact a capitalisé un total expérience et savoir-faire qui lui permet d'accompagner ses clients dans l'appropriation des technologies de l'information au profit du développement et de la prospérité de leurs entreprises. Son siège se situe au sud-ouest de la ville de Tizi-Ouzou, au Lot Bouaziz Villa N°B1 **[11].**

MS CONTACT a pour mission de garantir à ses clients des interventions de qualité dans les différents domaines de l'informatique, dont on cite :

- Installation de réseaux informatiques, téléphoniques
- Développement WEB.
- Maintenance.
- Distribution d'équipement informatique et bureautique.
- Développement de logiciels spécifiques.

II.2.1.1. Organigramme général

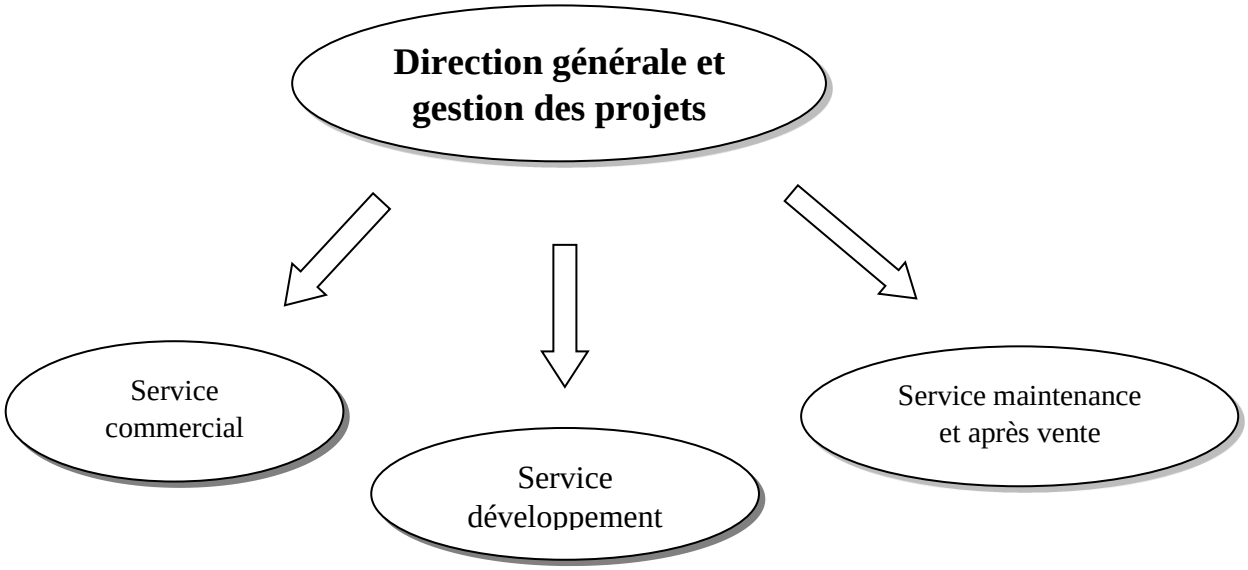


Figure II.2: Organisation générale de MS CONTACT INFORMATIQUE.

II.2.1.2. Description de l'organisation de l'entreprise d'accueil

a. Direction générale et gestion des projets : Le directeur général est le premier responsable de l'entreprise, il a pour taches de coordonner toutes les opérations de différents services, assurer la réalisation des objectifs fixés par la société et coordonner les travaux des employés.

b. Service commercial : chargé de la prise en charge de la gestion commerciale et tout ce qui concerne la vente de matériel et le logiciel

c. Service développement : responsable du développement des applications, étude des projets …..

d. Service maintenance et réseaux : chargé de la réparation et maintenance matériel, installation réseau et de solution de contrôle d'accès ….

II.2.2. Présentation du Logiciel ILUTRADE

L'étude de l'existant est une phase essentielle, qui permet d'étudier le système existant afin de voir les interactions du système et de le comprendre, elle doit être menée d'une façon claire et efficace pour parvenir à déterminer les besoins de l'organisation.

Le logiciel **ILUTRADE** est un logiciel de gestion commerciale et gestion de stocks, il fonctionne sous l'environnement Windows (7, 8, 10, Server 2003, 2008,2012) en monoposte et en client/serveur, intégrant une base de données de type FIREBIRD en langage SQL **[11].**
Ilutrade permet la gestion de plusieurs Sociétés, en ayant une seule Société active à la fois.

Les domaines d'activités du logiciel sont :

- **Achat-Revente en l'état** (Détail, gros et import) quel que soit le produit
- **Production** : gestion des matières premières et des produits finis, ainsi que l'assemblage de produits semi-finis.
- **Point de vente** : Superette, chaussure, habillement. Pour ce type d'utilisation, des fonctions supplémentaires sont offertes notamment :
  - Utilisation de lecteur code barre, de tiroir-caisse, d'imprimante ticket de caisse et d'imprimante code barre
  - Possibilité de générer des codes-barres pour les produits n'en disposant pas.
  - Fidélisation des clients par la gestion des cartes de fidélité (points accumulés lors des achats, remise fidélité, chèque cadeau,…)
  - Clôture quotidienne de caisse
  - Bilans journaliers (CA, bénéfice, recette…) [11] **.**

Les principales fonctions prises en charge par le logiciel ILUTRADE sont :

i. Gestion de clients

- Fiche client complète (code, raison sociale, adresse, tel, e-mail, coordonnées fiscales…).
- Catégorisation des clients.
- Suivi d'un client (créances, chiffre d'affaire, facture sur bon livraison…).

- Historique d'un client (vente, bon livraison, devis, règlement,..).
- Edition du chiffre d'affaire d'un client….

ii. Gestion des produits

- Fiche article complète.
- Définition de plusieurs références par produit : (référence , référence équivalente, référence fournisseur).

- Désignation produit illimitée.
- Gestion des produits composés (ensemble de produits) pour les montages et les assemblages.
- Classification des produits par famille, par modèle par rayon.
- Insertion d'une photo pour chaque produit.
- Impression d'étiquettes d'articles code-barres.
- Gestion des produits garantis à l'achat et à la vente (par N° de série) .

iii. Gestion des achats

- Gestion de tout le cycle des achats : Commande Fournisseur, Facture d'achat, Bon de Réception, Avoir fournisseur, Retour fournisseur.
- Possibilité de régler automatiquement plusieurs achats avec un même effet de paiement
- Transformation de facture d'achat en Bon d'entrée , de Commande en achat,…
- Suivi des restes à réceptionner sur une commande.
- Calcul du prix de revient à l'achat (Local ou Importation) en intégrant les frais, les taxes douanières…

iv. Gestion des fournisseurs

- La fiche fournisseur complète (code, raison sociale, adresse, tel, e-mail, coordonnées fiscales…)
- Suivi d'un fournisseur (dettes, chiffre d'affaire…)
- Historique fournisseur (achat, règlement...)

v. Gestion de ventes

- Gestion de cycle de vente : Devis, commande, bon de livraison, facture, avoir, bon de retour.
- Transformation de facture en Bon de Livraison.
- Transformation de pro forma en Commande.
- Suivi des restes à livrer sur une commande.
- Gestion des remises, ristournes.
- Gestion distincte des Bons de Livraison facturés et des Bons de Livraison en instance de facturation.

vi. Gestion des stocks

- Suivi et gestion de plusieurs magasins ou dépôts
- Calcul du Prix Moyen Unitaire Pondéré (PMUP) à chaque nouvelle Réception.
- La tenue réelle du Stock à chaque achat et à chaque vente.
- La gestion des sorties internes en PMUP.
- La gestion des Transferts inter-magasins.
- Seuil d'alerte et seuil de réapprovisionnement.
- Calcul des besoins de réapprovisionnement en fonction des seuils et des commandes en instance.
- Stock centralisé.
- La reconstitution du stock à n'importe quelle période.
- L'interrogation du stock.
- L'état du stock par produit, par fournisseur, par famille et par magasin.
- L'état de la disponibilité en stock valorisé, des ruptures de stock…

vii. Gestion et Suivi des Règlements

- Suivi des règlements ventes sur facture ou/et sur bon de livraison.
- Gestion des avances.
- Suivi des règlements des achats.
- Situation suivant période choisie des créances et des dettes.
- Balance des règlements.

A côté des gestions citées précédemment, ILUTRADE propose des fonctions supplémentaires telles que :

- Sauvegarde et restauration de données manuelles ou automatiques.
- Transfert de données inter-sociétés.
- Paramétrage de la formule du timbre.
- Paramétrage de l'interdiction de modifier un document validé par un simple utilisateur.
- Paramétrage de l'interdiction de modifier le prix de vente au moment de l'établissement d'un document de vente par un simple utilisateur.
- Documentation électronique au format PDF.

Présentation de quelques interfaces :

1. Création du dossier de travail

A tout lancement de ILUTRADE, une fenêtre qui regroupe l'ensemble des sociétés définis au préalable pour le choix du dossier de travail est présentée.
Chaque société se distingue des autres par ses propres paramètres de gestion et ses propres données, voir **(figure II.3).**

Figure II.3: Fenêtre de démarrage et de définitions des dossiers des sociétés.

2. Gestion des produits

La gestion des produits englobe 3 niveaux de gestion qui sont : les familles de produits, les modèles et les produits eux-mêmes, parmi elles, nous verrons la partie produit. En ce qui concerne les produits ILUTRADE met à la disposition des utilisateurs une fenêtre permettant de consulter tous les produits existants.

Figure II.4: catalogue des produits.

3. Gestion des clients

La gestion des clients consiste à l'ajout et la manipulation des clients existants dans l'entreprise, une interface est mise à la disposition des utilisateurs qui leur permet de les gérer.

Figure II.5: Fiche client.

4. Gestion des ventes

ILUTRADE met à la disposition des utilisateurs une interface qui leur permet le traitement des procédures de vente telles que : Bon de commande, Devis, Bon de livraison, Facture .

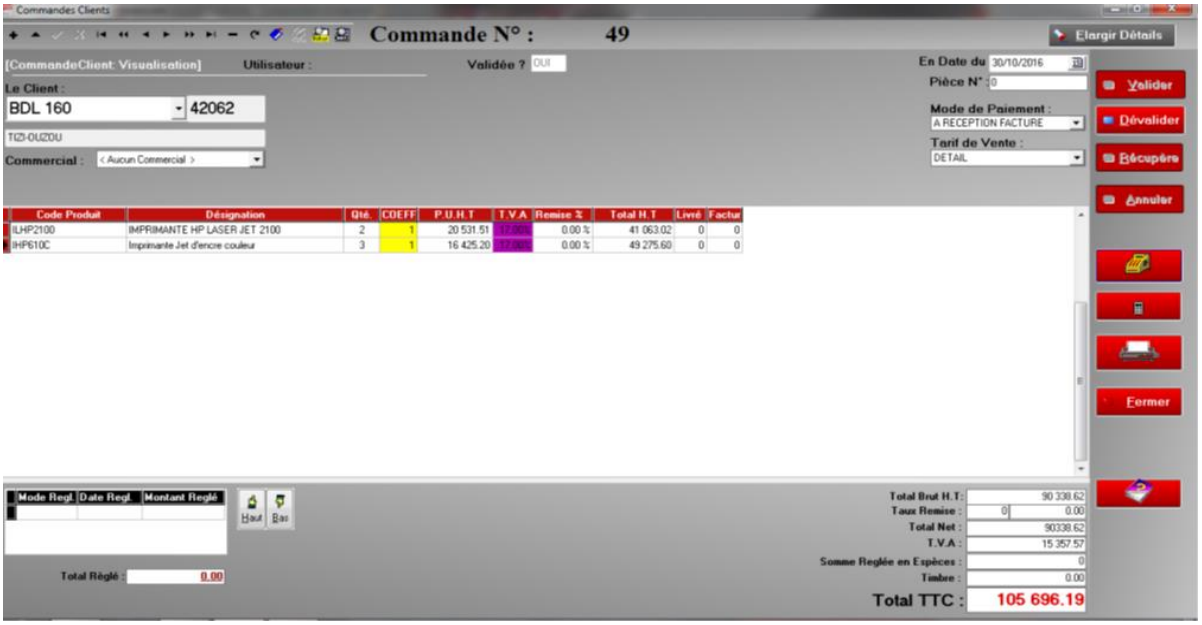


Figure II.6 : Interface d'édition d'un bon de commande client.

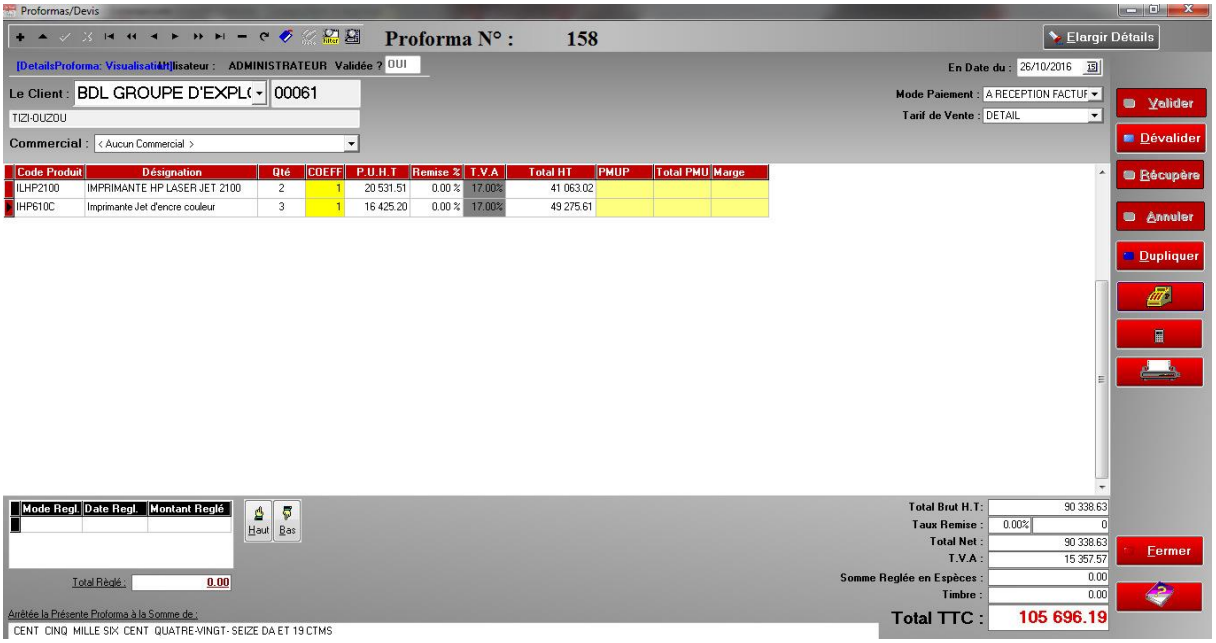


Figure II.7: Interface d'édition d'un devis client.

II.3. Problématique et objectifs

Les applications mobiles multiplateformes sont actuellement en pleine croissance grâce aux succès des smartphones et des nouveaux systèmes d'exploitation qui sont en cours d'apparition, cette tendance offre aux entreprises des formidables opportunités pour créer des nouveaux services ou élargir ces supports de communication.

Malgré la performance et la puissance du logiciel ILUTRADE dans la gestion commerciale des entreprises, il en manque la fonctionnalité de mobilité, il n'offre pas aux employées la possibilité d'utiliser le logiciel sur n'importe quel appareil et sur plusieurs plateformes, plus précisément pour la gestion de vente qui est la fonction principale de tout système commerciale, ce qui apporte plusieurs problèmes aux agents de vente notamment :

- Indisponibilité des informations hors entreprise
- Gestion manuelle des commandes et des livraisons
- Impossibilité de s'occuper de la moindre gestion hors entreprise, car les utilisateurs sont appelés à faire des déplacements et ne peuvent être présent tout le temps en entreprise.

Pour essayer de remédier aux insuffisances relevées et répondre aux contraintes et difficultés énoncées dans la problématique, nous avons opté pour la réalisation d'une application mobile multiplateforme pour la gestion des ventes qui répondra aux attentes suivantes :

- Délimitation de l'utilisation du logiciel.
- Délocalisation de certaines fonctions tels que :
  - Edition des bons de commandes et de devis.
  - Edition des bons de livraison et de facture.
  - mise à jour des clients (prospects).

Conclusion

Dans ce chapitre, nous avons définit la notion de gestion de ventes et présenter l'organisme d'accueil.

Par la suite, nous avons mené une étude sur l'existant qui nous a aidé à spécifier la problématique et les besoins de l'entreprise, ainsi déterminer les objectifs à atteindre.

Le chapitre qui suit, nous allons le consacrer à l'analyse et la conception de notre application.

Chapitre III

Analyse et conception

Introduction

Dans ce chapitre, nous commençons par présenter notre projet, puis on entamera la phase **« Analyse »** qui mettra en évidence les différents acteurs intervenants dans l'application et leurs besoins. Suivi d'une phase **« conception »,** s'appuyant sur la phase précédente, qui donnera la modélisation des objectifs à atteindre.

Notre travail sera consacré au développement d'une application multiplateforme, qui est une réécriture d'une partie de l'application desktop (ILUTRADE) qui permettra de délocaliser la gestion de ventes. Parmi les taches à réaliser :

- Mise à jour des prospects, clients
- Mise à jour des devis, bons de livraison….
- Paiement de la facture.

De ce fait, notre démarche se basera sur le langage UML, qui permettra de bien représenter les aspects fonctionnels, statiques et dynamiques de notre projet par la série des diagrammes qu'il offre **(cf. Annexe A).**

III.1. Analyse

III.1.1. Identification des acteurs

III.1.1.1. Définition d'un acteur

Un acteur représente un rôle joué par une entité externe (utilisateur humain, dispositif matériel ou autre système) qui interagit directement avec le système, il peut consulter et/ou modifier directement l'état du système, en émettant et/ou recevant des messages susceptibles d'être porteurs de données **[12].**

III.1.1.2. Les acteurs de notre système

Les acteurs qui interviendront dans notre système sont : **l'Agent de vente** qui est une personne inscrite et validée par l'administrateur, chargée des ventes et de la mise à jour des clients.

III.1.1.3. Diagramme de contexte

Le diagramme de contexte est un modèle conceptuel de flux, Il permet de bien délimiter le champ d'étude et de spécifier le nombre d'instances d'acteurs connectés au système.

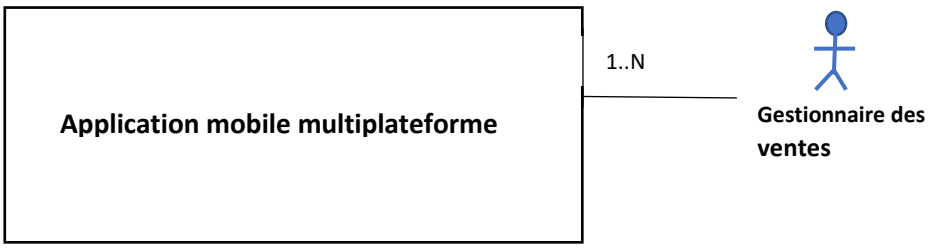


Figure III.1 : Diagramme de contexte.

III.1.2. Spécification des besoins

On trouve deux types de besoin : les besoins fonctionnels et les besoins non fonctionnels.

- ✓ **Les besoins fonctionnels :**

Un besoin est l'ensemble logique d'opérations qu'un programme ou une partie d'un programme doit exécuter pour répondre aux attentes d'un acteur.

Le tableau (**III.1)** résume les besoins fonctionnels de notre application :

Acteur	***Besoins***
Gestionnaire des ventes	**B1:** S'authentifier. **B2:** Consulter la liste des produits. **B3:** Ajouter un Client. **B4:** Modifier un Client. **B5:** Rechercher un Client. **B6:** Consulter la liste des Clients. **B7:** Etablir un Devis Client. **B8:** Modifier un Devis Client. **B9:** Télécharger un Devis Client. **B10:** Etablir une commande Client. **B11:** Modifier une commande Client. **B12:** Télécharger Bon de commande Client. **B13:** Etablir un Bon de Livraison Client. **B14** : Paiement bon de livraison. **B15:** Télécharger un Bon de Livraison Client. **B16** : Modifier Bon de Livraison Client. **B17:** Etablir une facture Client. **B18:** Télécharger une facture Client **B19:** Paiement facture Client. **B20:** Consulter stock.

Tableau III.1: Tableau de spécification des besoins.

✓ Les besoins non fonctionnels :

Il s'agit des besoins qui caractérisent le système. Ce sont des besoins en matière de performance, de type de matériel ou le type de conception.

Les besoins non fonctionnels de notre application se résument à :

- Architecture client/serveur.
- SGBD de type : MySQL
- Systèmes d'exploitation : tous
- Langages de programmation : AngularJS, Ionic, PHP
- La disponibilité : le système constitue le cœur de l'activité, il est nécessaire qu'il soit toujours en service.
- Sécurité : c'est primordial pour l'application, étant donné qu'il gère des données confidentielles, l'authentification est indispensable.
- Performance : fluidité du logiciel et la minimisation du temps de réponse.
- Doit être connectée au serveur de l'entreprise qui, lui-même est connecté à Internet.

III.1.3. Spécification des cas d'utilisation

III.1.3.1. Définition

Un cas d'utilisation est une manière spécifique d'utiliser un système où les acteurs sont à l'extérieur ; Ils modélisent tout ce qui interagit avec lui.

Un cas d'utilisation réalise un service de bout en bout, avec un déclenchement, un déroulement et une fin, pour l'acteur qui l'initie **(cf. Annexe A).**

III.1.3.2. Relations entre cas d'utilisation

Pour clarifier un diagramme, UML permet d'établir des relations entre les cas d'utilisation.

Il existe principalement :

- ✓ **La relation d'inclusion :** Un cas A est inclus dans un cas B si le comportement décrit par le cas A est inclus dans le comportement du cas B : on dit alors que le cas B dépend de A.

- ✓ **La relation d'extension** : Si le comportement de B peut être étendu par le comportement de A, on dit alors que A étend B. Une extension est souvent soumise à condition.

- ✓ **La relation de généralisation** : Un cas A est une généralisation d'un cas B si B est un cas particulier de A [12].

III.1.3.3. Diagrammes de cas d'utilisation

Nous allons décrire les diagrammes de cas d'utilisation selon les acteurs :

- ✓ Diagramme de cas d'utilisation « acteur : gestionnaire des ventes » (Voir figure **III.2**).

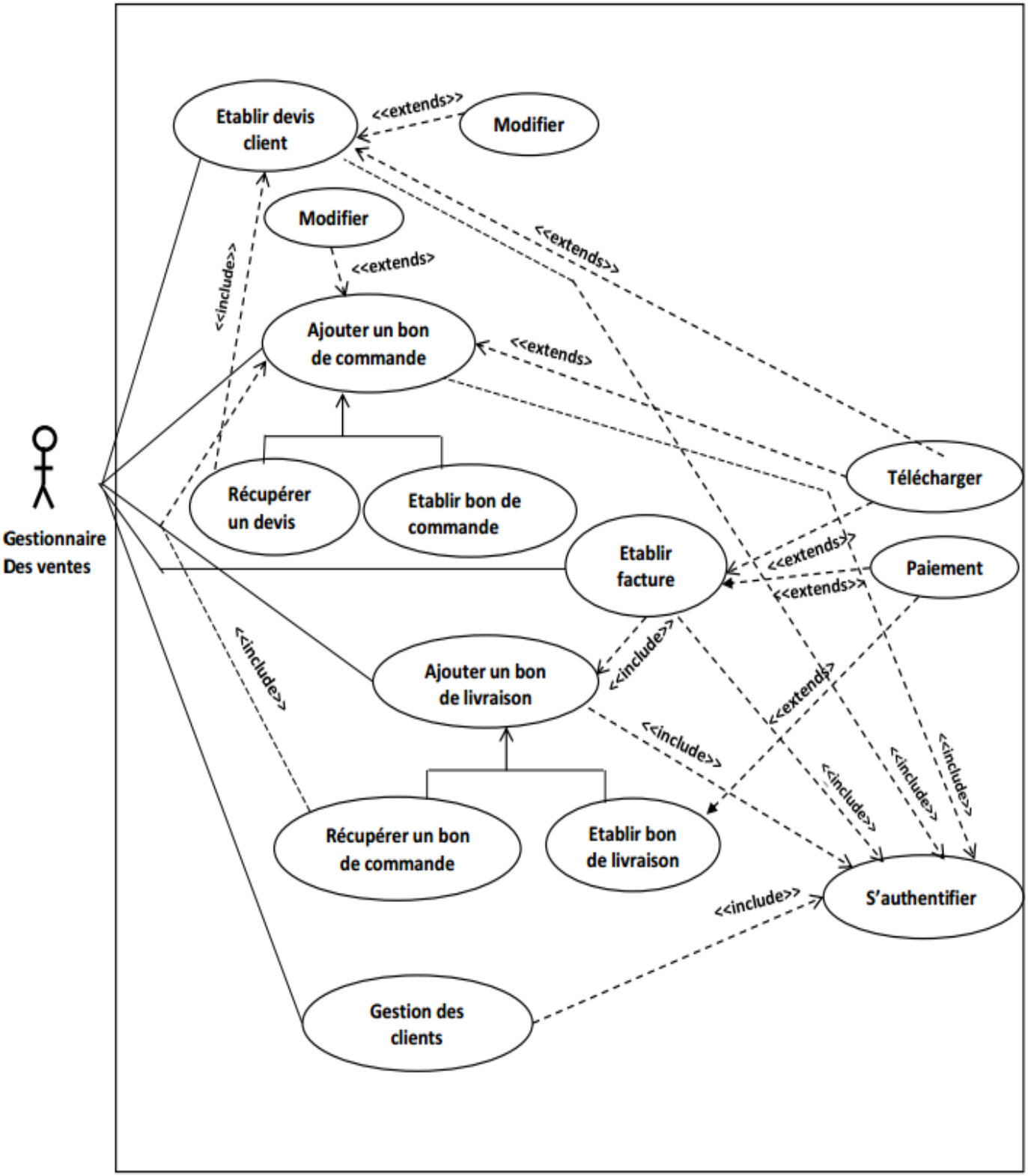


Figure III.2: Diagramme détaillé du cas d'utilisation « Gérer les ventes » .

III.1.3.4. Description textuelle des cas d'utilisation

Bien que de nombreux diagrammes d'UML permettent de décrire un cas, il est recommandé de rédiger une description textuelle car c'est une forme souple qui convient dans bien des situations. Une description textuelle couramment utilisée se compose de deux parties :

✓ La première partie permet d'identifier le cas. Elle doit contenir :

- Le nom du cas.
- Un résumé de son objectif.
- Les acteurs impliqués (principaux et secondaires).

✓ La deuxième partie contient la description du fonctionnement du cas sous la forme d'une séquence de messages échangés entre les acteurs et le système. Elle contient **une séquence nominale** qui correspond au fonctionnement nominal du cas.

Cette séquence se développe en trois points :

- **Les pré-conditions :** Elles indiquent dans quel état est le système avant que se déroule la séquence.
- L'enchaînement des messages.
- **Les post-conditions :** Elles indiquent dans quel état se trouve le système après le déroulement de la séquence nominale.

Comme on peut y avoir :

- **Une séquence alternative** : diverge de la séquence nominale (c'est un embranchement dans une séquence nominale).

- **Une séquence d'exception :** intervient quand une erreur se produit (l'enchainement nominal s'interrompt, sans retour à la séquence nominale).

Nous décrivons les cas suivants :

- ✓ S'authentifier.
- ✓ Ajouter un client.
- ✓ Etablir Devis Client.
- ✓ Etablir Bon de commande Client.
- ✓ Etablir Bon de livraison Client.
- ✓ Etablir une Facture.

Description du cas d'utilisation : « S'authentifier »
Identification Nom du cas : « S'authentifier ». Résumé : Permettre à l'utilisateur d'accéder à son espace. Acteur principal : Gestionnaire des ventes.
Pré-conditions L'utilisateur possède un compte créé préalablement. **Enchaînement nominal** 1. Le système demande à l'utilisateur de saisir le « Identifiant » et d'introduire « le mot de passe ». 2. L'utilisateur remplit les champs « Identifiant » et « Mot de passe » puis Valide. 3. Le système affiche à l'utilisateur son espace. **Enchaînement alternatif** **A1 : Login ou mot de passe saisit incorrect (informations incorrectes).** L'enchaînement peut démarrer après le point 2 de l'enchainement nominal : 3. Le système affiche une erreur d'identification. L'enchainement nominal reprend au point 1. **Post- conditions :** L'utilisateur accède à son espace.

Tableau III.2: Description du cas d'utilisation « S'authentifier ».

Description du cas d'utilisation : « Ajouter client »
Identification Nom du cas : « Ajouter Client ». Résumé : Permettre à l'utilisateur d'ajouter un client Acteur principal : Gestionnaire des ventes.
Pré-conditions : L'utilisateur doit s'authentifier. L'utilisateur accède au volet « client ». L'utilisateur accède à l'espace « Ajouter un client ». **Enchaînement nominal :** 1. Le système affiche un formulaire pour saisir les informations du client. 2. L'utilisateur saisit les informations puis les valide. 3. Le système affiche un message de confirmation de l'opération. **Enchaînement Alternatif** **A1 Un champ obligatoire non rempli.** L'enchaînement peut démarrer après le point 2 de l'enchainement nominal : 3. Le système affiche un message d'erreur. L'enchainement nominal reprend au point 1. **Post- conditions :** Le système enregistre le nouveau client dans la base de données.

Tableau III.3: Description du cas d'utilisation « Ajouter Client ».

Description du cas d'utilisation : « Etablir Devis Client »
Identification Nom du cas : « Etablir Devis Client ». Résumé : Permettre à l'utilisateur d'établir un proforma pour un client. Acteur principal : Gestionnaire des ventes.
Pré-conditions L'utilisateur doit s'authentifier. L'utilisateur accède au volet "Devis". **Enchaînement nominal** 1. L'utilisateur clique sur le bouton d'ajout d'un devis. 2. Le système affiche le formulaire d'ajout de devis. 3. L'utilisateur remplit le formulaire concernant le devis. 4. L'utilisateur ajoute les produits au devis et valide. 5. Le système affiche le devis. **Enchainement alternatif :** **A1 Champ non rempli.** L'enchaînement peut démarrer après le point 3 de l'enchainement nominal : 4. Le système affiche un message d'erreur. L'enchainement nominal reprend au point 2. **Post- conditions :** Le système enregistre le devis proforma dans la base de données.

Tableau III.4: Description du cas d'utilisation « Etablir Devis Client ».

Description du cas d'utilisation : « Etablir Bon de commande Client »

Identification

Nom du cas : « Etablir bon de commande client ».

But : Permettre à l'utilisateur d'établir un bon de commande pour un client.

Acteur principal : Gestionnaire des ventes.

Pré-conditions

L'utilisateur doit s'authentifier.

L'utilisateur accède au volet "Commande".

Enchaînement nominal

1. L'utilisateur clique sur le bouton d'ajout de commande.
2. Le système affiche le formulaire d'ajout de commande.
3. L'utilisateur remplit les champs concernant la commande.
4. L'utilisateur ajoute les produits au bon de commande et valide.
5. Le système affiche le bon de commande ajouté.

Enchainement alternatif :

A1 Récupérer un Devis.

L'enchaînement peut démarrer après le point 2 de l'enchainement nominal :

3. L'utilisateur clique sur le bouton "Importer".
4. Le système affiche la liste des devis.
5. L'utilisateur choisit le devis à importer.
6. Le système charge les données du Devis.
7. L'utilisateur valide la commande.
8. Le système affiche la commande ajoutée.

A2 Champ non rempli.

L'enchaînement peut démarrer après le point 3 de l'enchainement nominal :

4. Le système affiche un message d'erreur.

L'enchainement nominal reprend au point 2. **Post- conditions :** Le système enregistre la commande dans la base de données.

Tableau III.5: Description du cas d'utilisation « Etablir Bon commande Client ».

Description du cas d'utilisation : « Etablir Bon de livraison Client »
Identification Nom du cas : « Etablir bon de livraison client ». Résumé : Permettre à l'utilisateur d'établir un bon de livraison pour un client. Acteur principal : Gestionnaire des ventes.
Pré-conditions L'utilisateur doit s'authentifier. L'utilisateur accède au volet "Bon de livraison". **Enchaînement nominal** 1. L'utilisateur clique sur le bouton d'ajout. 2. Le système affiche le formulaire d'ajout. 3. L'utilisateur remplit les champs concernant le bon de livraison. 4. L'utilisateur ajoute les produits au bon de livraison et valide. 5. Le système affiche le bon de livraison ajouté. **Enchainement alternatif :** **A1 Récupérer de commande.** L'enchaînement peut démarrer après le point 2 de l'enchainement nominal : 3. L'utilisateur clique sur le bouton "Importer". 4. Le système affiche la liste des bons de commande.

5. L'utilisateur choisit le bon de commande à importer.

6. Le système charge les données du bon.

7. L'utilisateur valide le bon de livraison.

8. Le système affiche le bon de livraison ajouté.

A2 Champs obligatoire non remplit.

L'enchaînement peut démarrer après le point 5 de l'enchainement alternatif A1.

6. Le système affiche un message d'erreur.

L'enchainement nominal reprend au point 4 de l'enchainement alternatif A1.

Post- conditions :

Le système enregistre le bon de livraison dans la base de données.

Tableau III.6: Description du cas d'utilisation « Etablir Bon de livraison Client ».

Description du cas d'utilisation : «Etablir Facture Client»
Identification Nom du cas : «Etablir facture client». Résumé : Permettre à l'utilisateur d'établir une facture pour un client. Acteur principal : Gestionnaire des ventes.
Pré-conditions L'utilisateur doit s'authentifier. L'utilisateur accède au volet "Facture". **Enchaînement nominal** **A1 Récupérer bon de livraison.** 1. L'utilisateur clique sur le bouton d'ajout. 2. Le système affiche le formulaire d'ajout. 3. L'utilisateur clique sur le bouton "Importer". 4. Le système affiche la liste des bons de livraisons.

5. L'utilisateur choisit les bons de livraisons à importer en une seule facture. 6. Le système charge les données du bon. 7. L'utilisateur valide la facture 8. Le système affiche la facture ajoutée. **A2 Champs obligatoire non remplit.** L'enchaînement peut démarrer après le point 5 de l'enchainement A1. 6. Le système affiche un message d'erreur. L'enchainement nominal reprend au point 4. **Post- conditions :** Le système enregistre la facture dans la base de données.

Tableau III.7: Description du cas d'utilisation « Etablir Facture Client ».

Description du cas d'utilisation : « Consulter stock »
Identification Nom du cas : « **Consulter stock** ». But : Permettre à l'utilisateur de consulter la liste des produits et leurs disponibilités dans le stock. Acteur principal : Gestionnaire des ventes.
Pré-conditions L'utilisateur doit s'authentifier. L'utilisateur accède au volet "Consulter stock". **Enchaînement nominal** 1. Le système affiche la liste des produits.

Tableau III.8: Description du cas d'utilisation « Consulter stock ».

III.2. Conception

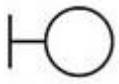
Dans cette phase une nouvelle vue du modèle fait son apparition. Cette vue exprime les modules et les exécutables physiques sans aller à la réalisation concrète du système. Elle est basée sur :

- Le diagramme de séquence.
- Le diagramme de classe.

III.2.1. Diagramme de séquence

L'objectif de ce type de diagramme offert par UML est de représenter les interactions entre les objets en mettant l'accent sur le classement chronologique des messages échangés. Les scénarios sont des instances des cas d'utilisation et sont traduits en diagrammes de séquences. Les classes d'analyses peuvent être scindées en trois catégories d'objet :

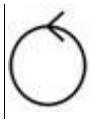
- ✓ **Les objets de type interface** : ils représentent l'interface entre l'acteur et le système.

L'icone : 

- ✓ **Les objets de type entité** : ils représentent principalement les bases de données.

L'icône : 

- ✓ **Les objets de contrôle** : ils représentent les processus du système.

L'icône : 

Vu le nombre élevé de cas d'utilisation recensés, et afin d'éviter qu'ils s'étalent sur plusieurs pages, nous avons décidé d'en étudier que quelques diagrammes :

- Diagramme de séquence du cas " S'authentifier".
- Diagramme de séquence du cas "Etablir Commande Client".
- Diagramme de séquence du cas "Etablir Devis Client".
- Diagramme de séquence du cas "Etablir la facture client".
- Diagramme de séquence du cas "Bon de Livraison client".

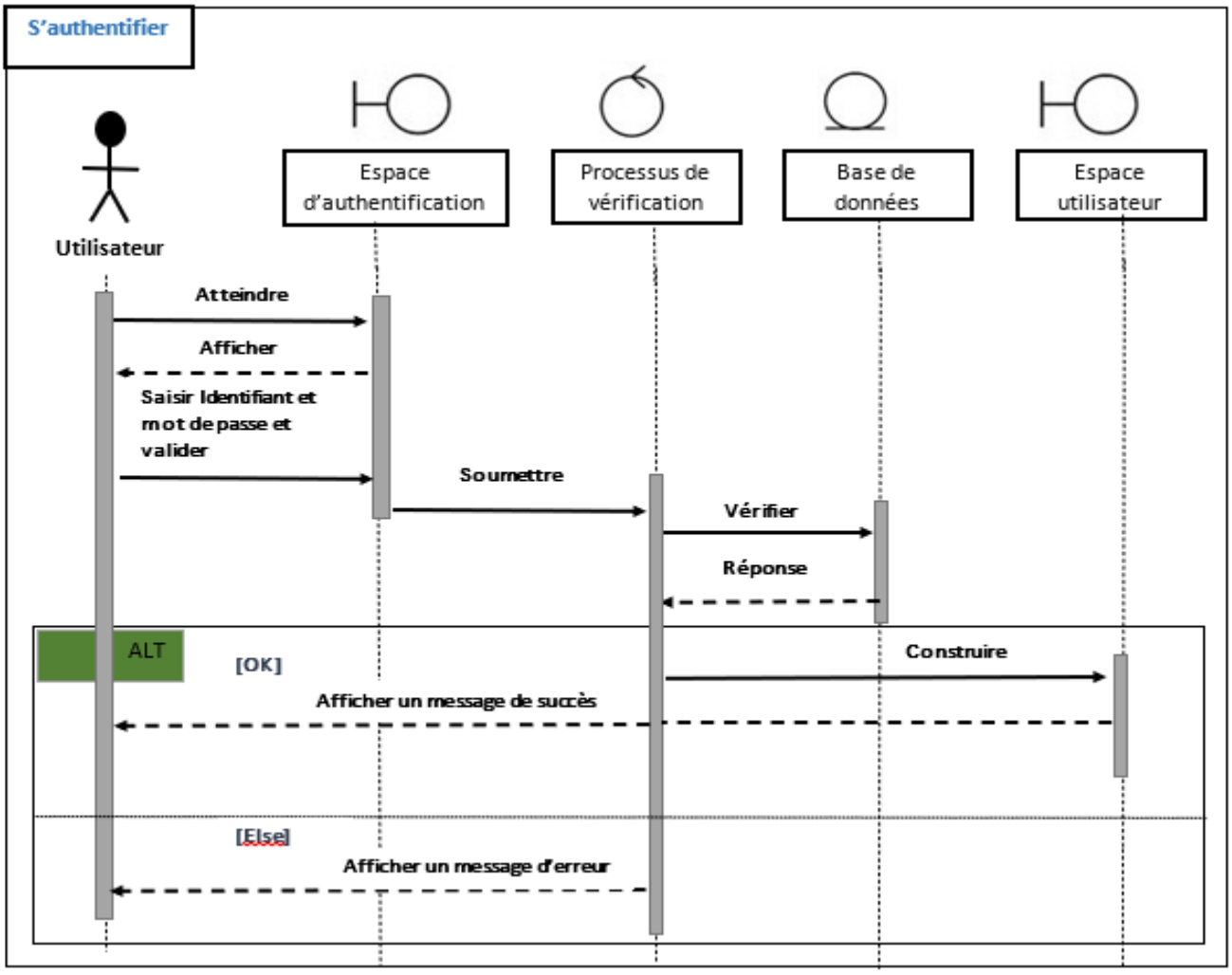


Figure III.3: Diagramme de séquence du cas " S'authentifier".

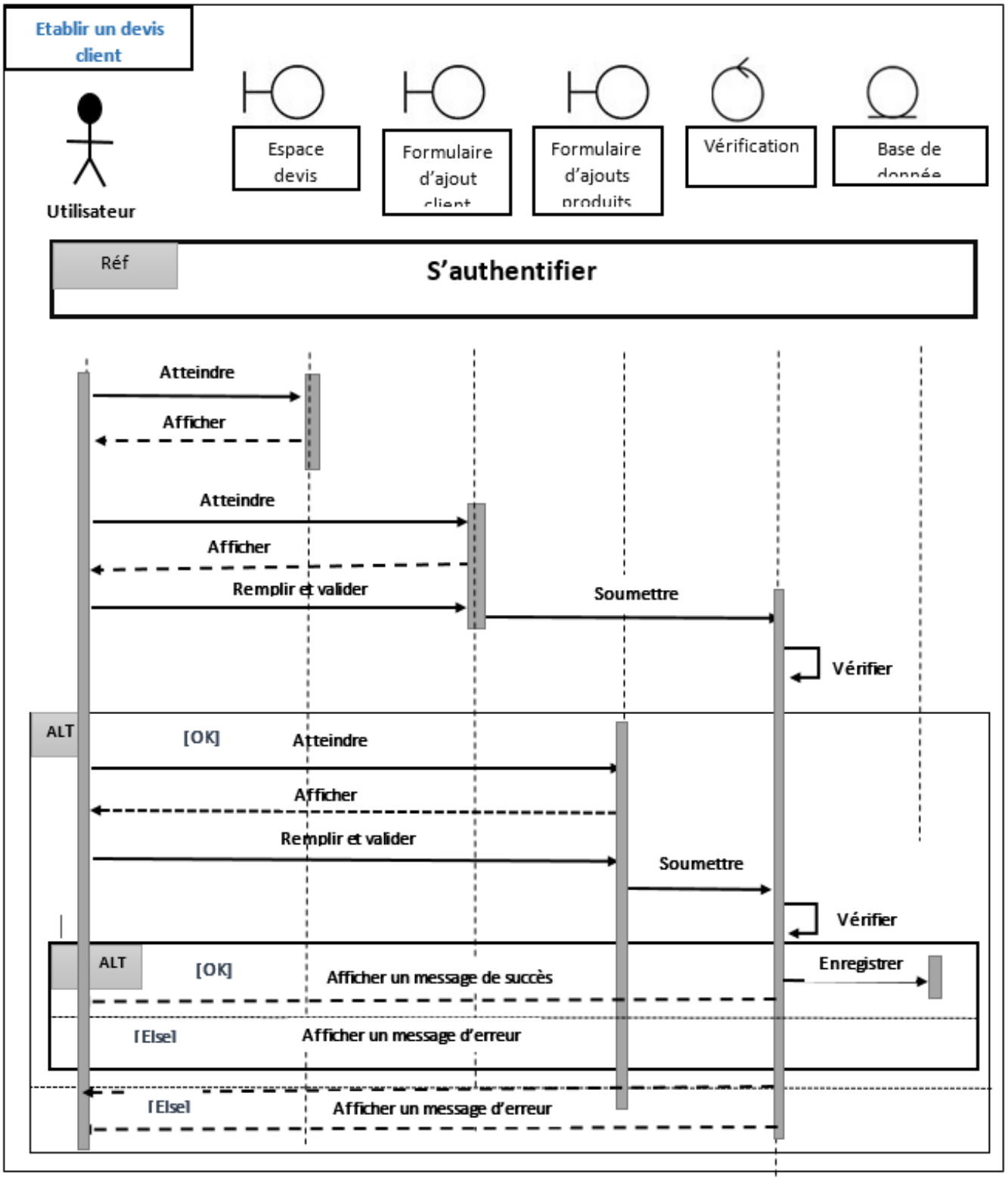


Figure III.4: Diagramme de séquence du cas "Etablir devis Client".

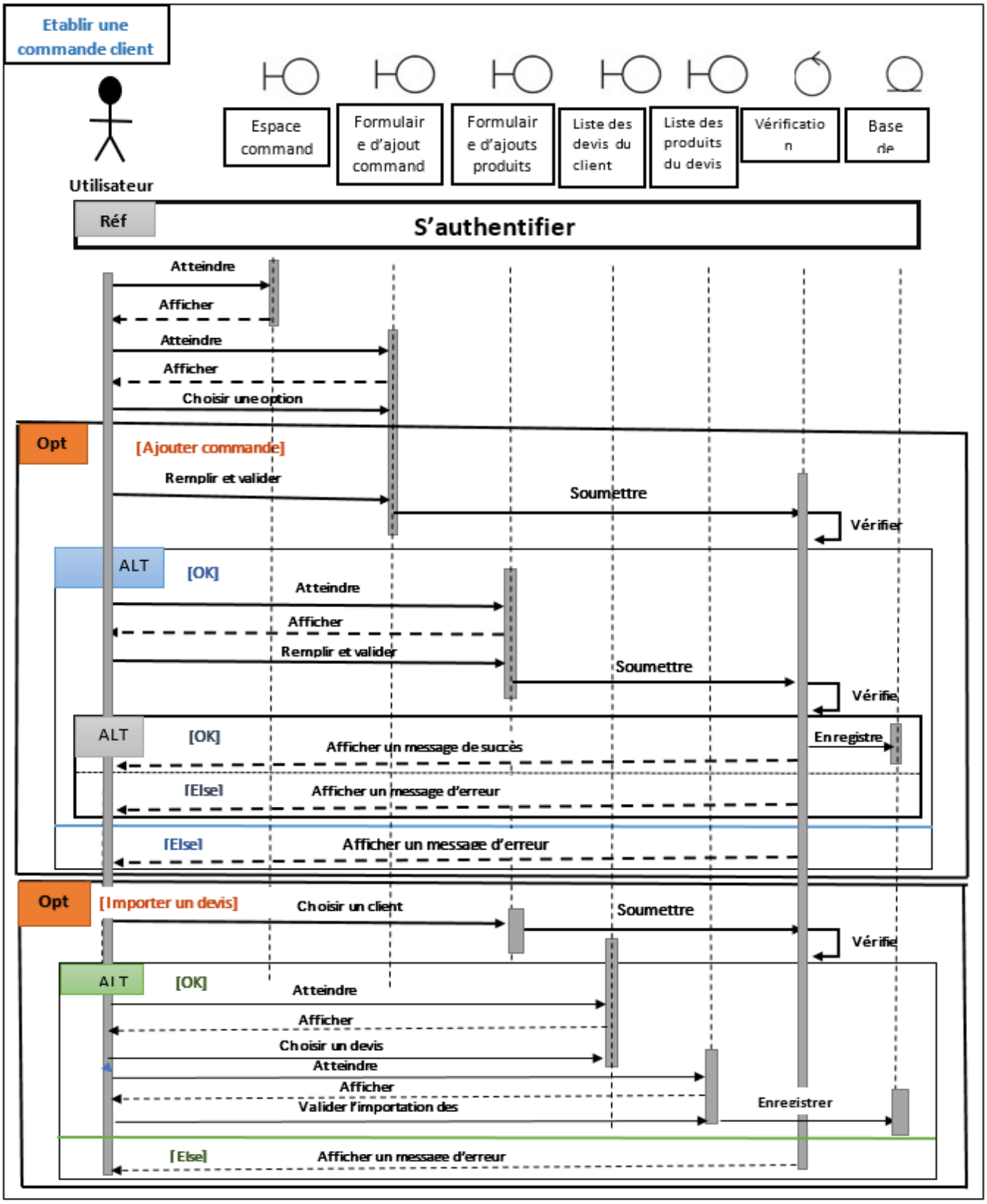


Figure III.5: Diagramme de séquence du cas "Etablir commande Client".

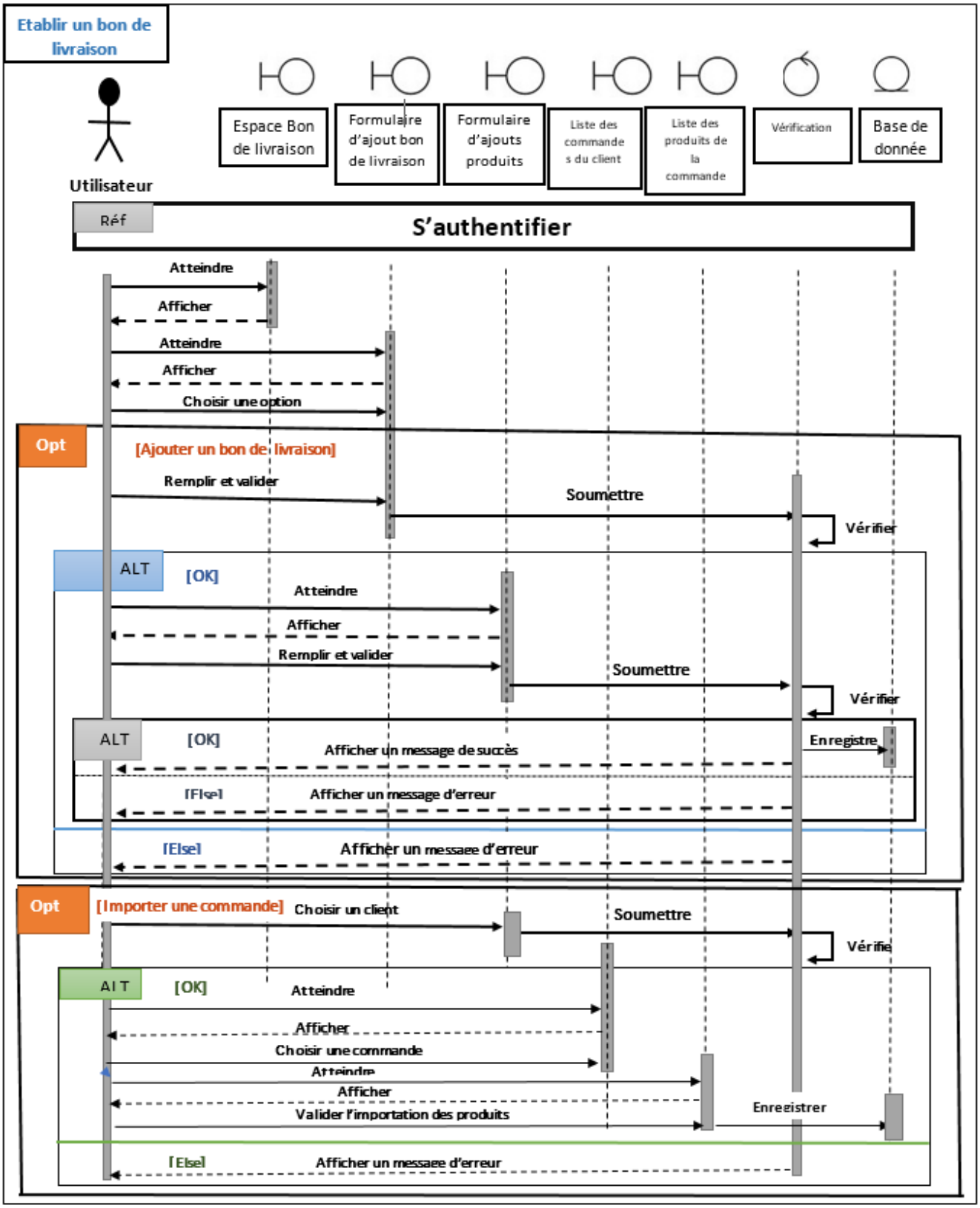


Figure III.6: Diagramme de séquence du cas " Etablir bon de livraison".

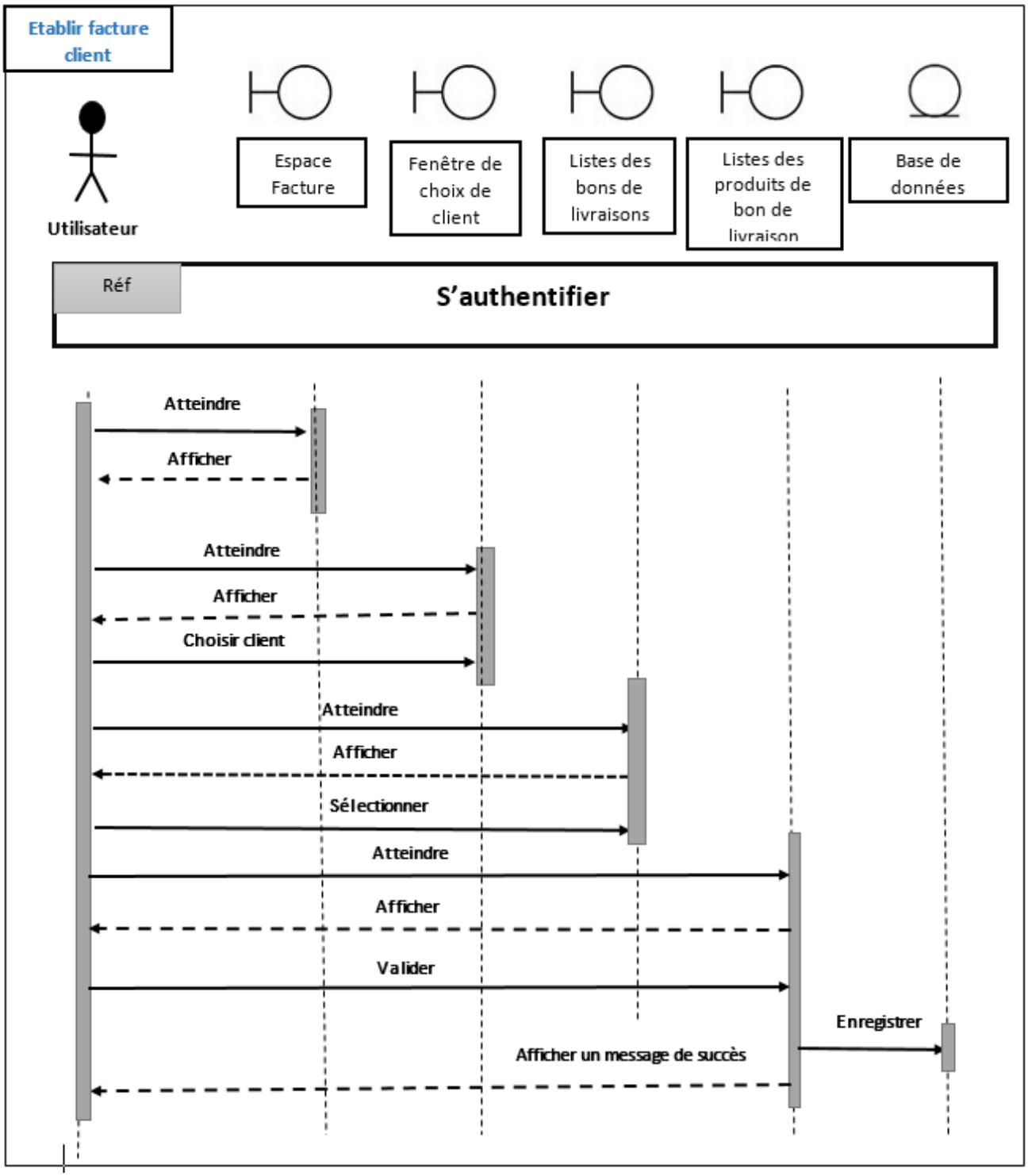


Figure III.7: Diagramme de séquence du cas "Etablir facture Client".

III.2.2. Diagramme de classe

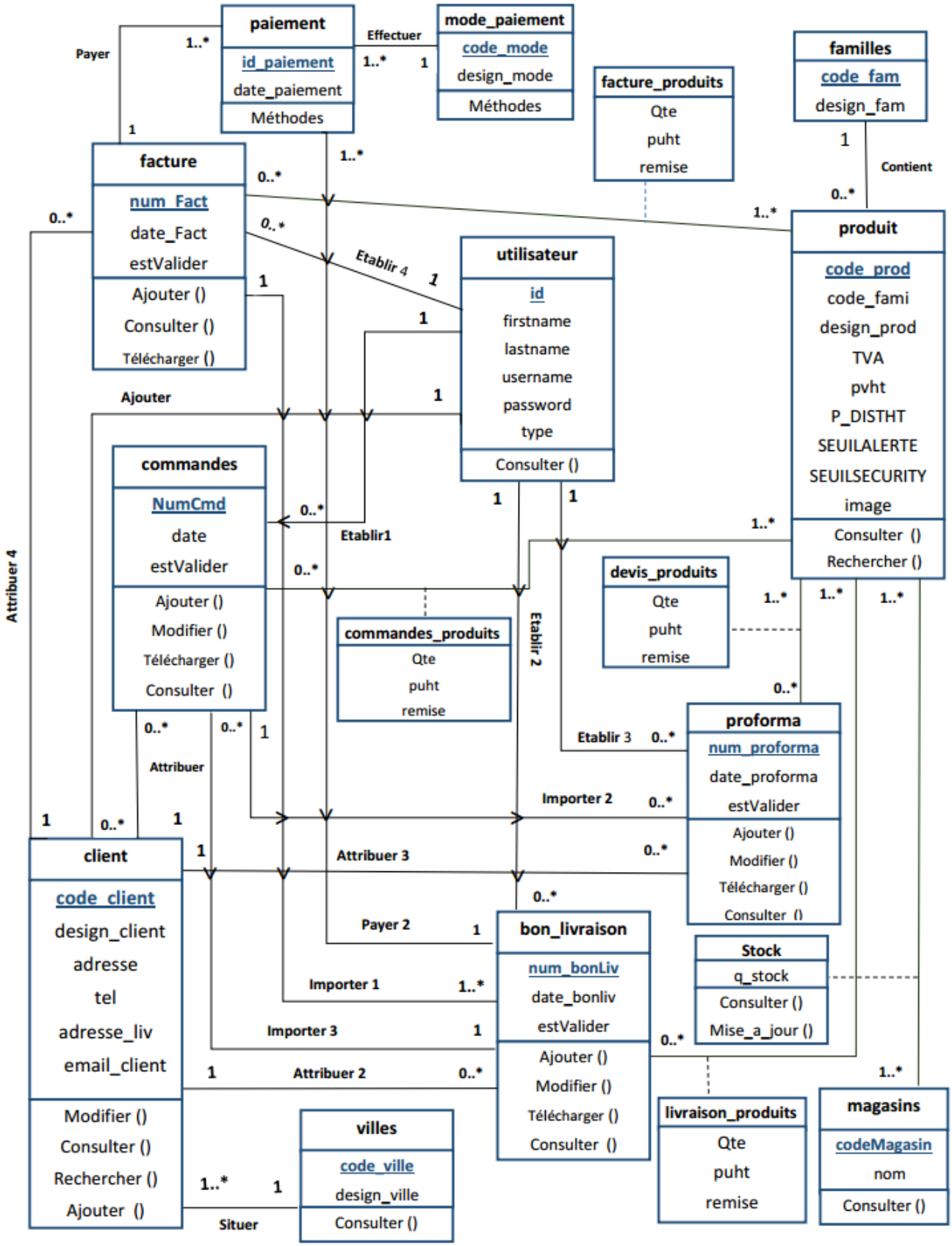
Le diagramme de classes est considéré comme le plus important de la modélisation orientée objet, il montre la structure interne du système.

Il contient principalement des classes, ces dernières contiennent des attributs et des opérations, aussi il démontre les liens entre ces classes ainsi que leurs cardinalités.

Classe	***Attribut***	***Code***	***Type***	***Méthode***
Produit	Code produit Désignation produit Taxe sur valeur ajouté Prix distribution hors taxe Prix de vente hors taxe Seuil d'alerte Seuil de sécurité image	code_prod design_prod TVA P_DISTHT pvht SEUILALERTE SEUILSECUTE image	AN AN N N N N N AN	Consulter() Rechercher()
Famille	Code famille Désignation famille	code_fam design_fam	AN AN	Consulter()
Client	Code client Désignation client Adresse client Téléphone Adresse de livraison Email	code_client design_client adresse tel adresse_liv email_client	AN AN AN AN AN AN	Consulter () Ajouter() Modifier () Rechercher()
Villes	Code ville Désignation ville	code_ville design_ville	AN AN	Consulter()
Proforma	Numéro bon Date bon	num_proforma date_proforma	N D	Consulter() Ajouter() Modifier() Importer () Télécharger()

Commande	Numéro bon Date bon	NumCmd date	N D	Ajouter() Modifier () Consulter() Importer() Télécharger()
Bon_livraison	Numéro bon Date bon	num_bonLiv date_bonLiv	N D	Ajouter() Consulter() Importer() Télécharger()
Facture	Numéro de facture Date facture	num_Fact date_Fact	N D	Ajouter() Consulter() Télécharger()
Paiement	Date du paiement	date_paiement	D	Consulter()
Mode_paiement	Code mode paiement Désignation mode paiement	code_mode design_mode	AN AN	Consulter()
Magasins	Code du magasin Désignation du magasin	codeMagasin magasin	AN AN	Consulter()
Utilisateur	Numéro d'utilisateur Nom d'utilisateur Mot de passe Nom Prénom	numUtil username password nom prenom	N AN AN AN AN AN	Consulter()

Tableau III.9: Dictionnaire de donnée.


paiement
id_paiement
date_paiement
Méthodes
Payer
Effectuer
mode_paiement
code_mode
design_mode
Méthodes
facture_produits
Qte
puht
remise
familles
code_fam
design_fam
Contient
facture
num_Fact
date_Fact
estValider
Ajouter ()
Consulter ()
Télécharger ()
Etablir 4
utilisateur
id
firstname
lastname
username
password
type
Consulter ()
produit
code_prod
code_fami
design_prod
TVA
pvht
P_DISTHT
SEUILALERTE
SEUILSECURITY
image
Consulter ()
Rechercher ()
Ajouter
commandes
NumCmd
date
estValider
Ajouter ()
Modifier ()
Télécharger ()
Consulter ()
Etablir1
commandes_produits
Qte
puht
remise
devis_produits
Qte
puht
remise
Etablir 2
Attribuer 4
Attribuer
Etablir 3
proforma
num_proforma
date_proforma
estValider
Ajouter ()
Modifier ()
Télécharger ()
Consulter ()
Importer 2
client
code_client
design_client
adresse
tel
adresse_liv
email_client
Modifier ()
Consulter ()
Rechercher ()
Ajouter ()
Attribuer 3
Payer 2
bon_livraison
num_bonLiv
date_bonliv
estValider
Ajouter ()
Modifier ()
Télécharger ()
Consulter ()
Stock
q_stock
Consulter ()
Mise_a_jour ()
Importer 1
Importer 3
Attribuer 2
livraison_produits
Qte
puht
remise
magasins
codeMagasin
nom
Consulter ()
villes
code_ville
design_ville
Consulter ()
Situer

III.2.3. Conception de la base de données

Une base de données est une collection de données cohérente et structurée.

Pour concevoir la base de données du système, nous avons commencé par recenser les différentes entités qui peuvent intervenir dans l'application.

En se basant sur ces entités, et en respectant les différentes règles du modèle relationnel nous avons déduit les tables de la base de données.

III.2.3.1. Passage du diagramme de classe au modèle relationnel de données

-Chaque classe devient une table dont la clé primaire correspond généralement à l'identifiant de la classe.

-L'association de plusieurs à plusieurs devient une table dont la clé primaire est composée des clés primaires des tables correspondant aux classes associées.

-L'association d'un à plusieurs (père-fils) provoque la migration de la clé primaire du père vers le fils et qui devient clé étrangère.

-Les classes spécifiques deviennent des tables dont la clé primaire est celle de la table correspondant à la classe générale.

III.2.3.2. Description des liens du diagramme de classes

Relation	*Collection*	*Cardinalité*	*Propriété*
devis_produits	✓ produit ✓ proforma	0..* 1..*	qte puht remise
commandes_produits	✓ produit ✓ commande	0..* 1..*	qte puht remise
livraison_produits	✓ produit ✓ bon_livraison	0..* 1..*	qte puht remise
facture_produits	✓ produit ✓ facture	0..* 1..*	qte puht remise
stock	✓ produit ✓ magasins	1..* 1..*	q_stock

Tableau III.10 : Description des liens du diagramme de classes.

III.2.3.3. Le modèle relationnel

Remarque :

Le mot en **Gras :** clé primaire.

Le mot souligné : clé secondaire.

produit (**code_prod**, code_fami , design_prod, TVA, pvht, P_DISTHT, SEUILALERTE, SEUILSECURITE, image);

famille (**code_fam**, design_fam);

client (**code_client**, id_ville , design_client, adresse, tel, adresse_liv, email_client) ;

villes (**code_ville** , design_ville) ;

proforma (**num_proforma** , id_client , user_id, date_proforma , estValider) ;

commande (**NumCmd**, user_id , id_client, date, estValider);

bon_livraison (**num_bonLiv**,id_client,num_Fact,num_bonCmd,user_id ,date_bonLiv, estValider , reste_du);

facture (**num_Fact,** id_client , user_id, date_Fact , reste_du, estValider);

mode_paiement (**code_mode,** design_mode);

paiement (**id_paiement,** id_modePaiement, id_Fact, date_paiement, idbon_Liv);

magasins (**codeMagasin,** magasin);

utilisateur (**id** , nom, prenom , username , password , type)

devis_produits (id_prod, id_devis, qte, puht, remise);

commandes_produits (id_produit, Num_cmd, qte, puht, remise);

livraison_produits (id_prod, id_bonLiv, qte, puht, remise);

facture_produits (id_prod, id_Fact, qte, puht, remise);

stock (id_prod, id_magasin, q_stock);

III.2.3.4. Architecture de l'application

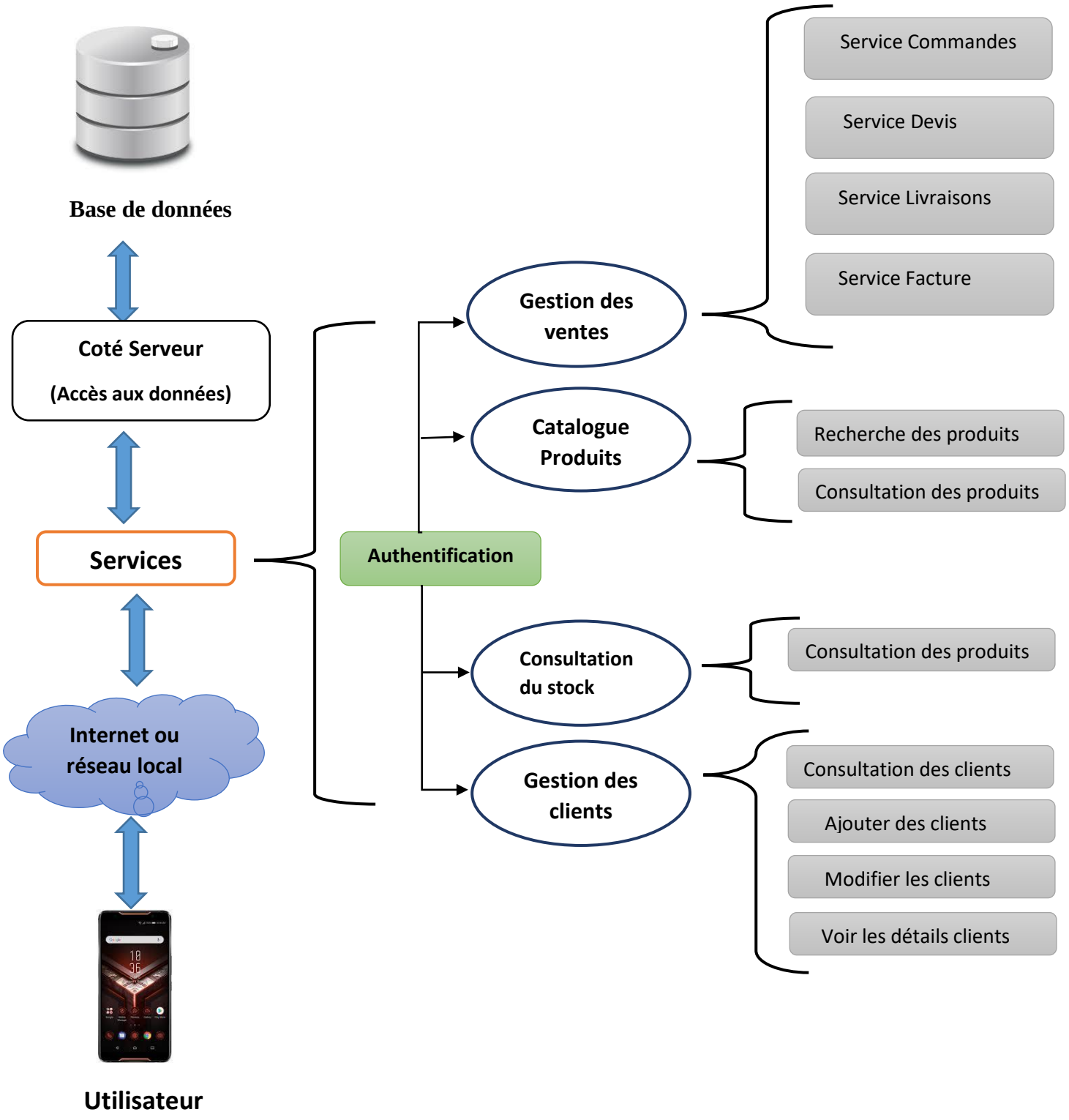


Figure III.9: Architecture trois tiers de l'application.

Conclusion

Durant ce chapitre nous avons présenté l'architecture conceptuelle de notre application en utilisant le formalisme UML.

Nous avons défini les acteurs de notre application et les tâches qu'ils assurent, puis élaboré le diagramme de contexte, le diagramme de cas d'utilisation, les diagrammes de séquence et le diagramme de classes. Enfin, nous avons conçu le modèle relationnel.

Dans le chapitre suivant nous entamerons l'étape d'implémentation de notre application.

Chapitre IV

Développement d'un prototype

Introduction

Dans ce chapitre nous allons commencer tout d'abord par la description de notre environnement de travail, les différents langages ainsi que les outils utilisés pour l'implémentation.

Par la suite nous allons présenter le fonctionnement général de notre application ainsi que ses différentes interfaces.

IV.1. Environnement de développement

✓ Visual Studio Code

C'est un éditeur du code source développé par Microsoft pour Windows, Linus et MacOs, qui prend immédiatement en charge presque tous les principaux langages de programmation. Plusieurs d'entre eux sont inclus par défaut, par exemple JavaScript, TypeScript, CSS et HTML, mais d'autres extensions de langage peuvent être trouvées et téléchargées gratuitement à partir de VS Code Marketplace.

Il a été présenté lors de la conférence des développeurs Build d'avril 2015 comme un éditeur de code multiplateforme **[12].**

✓ L'invite de commandes

C'est un logiciel d'interprétation des commandes DOS, Windows et OS/2 qui affiche une interface utilisateur en ligne de commande de type Win32.

Cette dernière c'est une infrastructure permettant la création et gestion d'invites de commandes ou d'applications consoles sous Windows **[13].**

✓ Xampp

Il est constitué d'un ensemble de logiciels permettant de mettre en place un serveur Web local, c'est une distribution Apache entièrement gratuite et facile à installer contenant MySQL, PHP et Perl. Le paquetage open source XAMPP a été mis au point pour être incroyablement facile à installer et à utiliser **[14].**

✓ PhpMyAdmin

PhpMyAdmin est un outil logiciel gratuit écrit en PHP, destiné à gérer l'administration de MySQL sur le Web.

Prend en charge une large gamme d'opérations sur MySQL tel que la gestion des bases de données, des tableaux, des colonnes, des relations, des index, des utilisateurs, des autorisations. Etc.

Ses opérations peuvent être effectuées via l'interface utilisateur, alors il offre aussi la possibilité d'exécuter directement une instruction SQL **[15].**

✓ Xcode

Est un éditeur de texte ultra rapide pour la création des applications pour Mac et IPhone, dispose d'une interface qui unifie le design, la programmation, le test et le débogage **[16].**

✓ Visual C++

Est un environnement de développement intégré pour Windows , conçu par Microsoft pour langages de programmation C et C++ et intégrant différents outils pour développer, compiler, déboguer un programme en C++ s'exécutant sur Windows, ainsi que des bibliothèques comme les MFC **[16].**

✓ Apache

Le logiciel libre Apache HTTP est un serveur HTTP créé et maintenu au sein de la fondation Apache. C'est le serveur HTTP le plus populaire du World Wide Web. Il est distribué selon les termes de la licence Apache.

IV.2. Langages de développement

❖ PHP

PHP (Hypertext Preprocessor) est un langage de scripts généraliste et Open Source, spécialement conçu pour le développement d'applications web. Il peut être intégré facilement au HTML **[17].**

❖ SQL

SQL est un langage de manipulation de bases de données mis au point dans les années 70 par IBM, il permet d'effectuer trois types de manipulations :

1. La manipulation des tables : Création, suppression, modification de la structure.

2. Les manipulations des données de la base : Sélection, modification, suppression d'enregistrement.

3. La gestion des droits d'accès aux tables : Contrôle des données, droit d'accès, validation des modifications **[18].**

❖ Ionic

C'est un Framework basé initialement sur AngularJs et Apache Cordova.

Ionic permet de créer un code multisupport en utilisant des outils Web comme HTML, CSS, JavaScript, afin de générer des applications IOS, Android, Chrome, Windows Phone et bien d'autres**[19].**

❖ Angular

C'est un Framework open source écrit en JavaScript. Il permet la création d'applications Web et plus particulièrement des applications web accessibles via une page web unique qui permet de fluidifier l'expérience utilisateur **[20].**

❖ TypeScript

Un langage de programmation libre et open source développé par Microsoft qui a pour but d'améliorer et de sécuriser la production de code JavaScript.

Le code TypeScript est transcompilé en JavaScript, et peut ainsi être interprété par n'importe quel navigateur web ou moteur JavaScript **[21].**

❖ NodeJs

Est un langage interpréter coté serveur écrit en JavaScript, il mit en disposition des bibliothèques JavaScript selon le besoin grâce au gestionnaire de baquet NPM.

❖ HTML

Un langage de balisage : il utilise une structure du code sous forme de balises.

Il permet la représentation des pages Web statiques, c'est un langage permettant d'écrire de l'hypertexte **[13].**

❖ CSS

Un langage informatique qui décrit la présentation des documents HTML et XML.

CSS devient couramment utilisé dans le développement Web et bien pris en charge par les navigateurs web. Les standards définissant CSS sont publiés par le World Wide Web Consortium (W3C) **[19].**

❖ XML

Extensible Markup Language est un langage informatique de balisage générique.

Il sert essentiellement à stocker/transférer des données de type texte Unicode structurées en champs arborescents. Ce langage a été standardisé par le W3C en février 1998 et est maintenant très populaire. IL est similaire à l'HTML de par son système de balisage, permet de faciliter l'échange d'information sur Internet. Il offre ainsi plus de souplesse de développement, facilite les modifications du code et assure la séparation entre la présentation et le comportement des objets **[22].**

IV.3. Protocole et format de données

1. Protocole de communication

Dans notre projet, nous avons utilisé le protocole HTTP, afin de communiquer les données entre le client Mobile et le serveur web. En effet, Le HTTP est un protocole qui définit la communication entre un serveur et un client.

2. Format de données communiquées : (JSON)

JSON (JavaScript Object Notation – Notation Objet issue de JavaScript) est un format léger d'échange de données. Il est facile à lire ou à écrire pour des humains. Il est aisément analysable ou générale par des machines. Il est basé sur un sous-ensemble du langage de programmation JavaScript. JSON est un format texte complètement indépendant de tout langage, mais les conventions qu'il utilise seront familières à tout programmeur habitué aux langages descendant du C, comme par exemple : C lui-même, C++, C#, Java, JavaScript, Perl, Python et bien d'autres. Ces propriétés font de JSON un langage d'échange de données idéal.

JSON se base sur deux structures : Une collection de couples nom/valeur et une liste de valeurs ordonnées **[23].**

IV.4. Présentation des interfaces de notre application

L'application prend en charge 4 principales fonctionnalités : gestion des ventes, mise à jour des clients, catalogue des produits et consultation du stock.

Nous allons présenter dans ce qui suit les principales interfaces illustrant le fonctionnement de l'application :

- Interface d'authentification.
- Interface « Configuration serveur »
- Interface Accueil (Espace utilisateur)
- Interface « Clients » de l'espace "Gestion des clients"
- Interface « Ajouter un Client » et « Modifier un Client »
- Volet « Devis ».
- Interface « Ajout devis »
- Interface «Ajout d'u produit dans un devis»
- Interface « Détails devis »
- Volet « commandes » .
- Interface « Importer un devis »
- Volet « Bon de livraison »
- Interface « nouveau bon de livraison »
- Interface d'établissement d'une facture
- Interface « Paiement Facture »
- Interface « Produits » dans l'espace « catalogue produits »
- Interface « Stock » dans l'espace « Consultation du stock »

Interface d'authentification :

Au démarrage de l'application, nous avons l'interface d'authentification qui s'affiche. L'utilisateur doit saisir son nom d'utilisateur et son mot de passe pour s'authentifier.

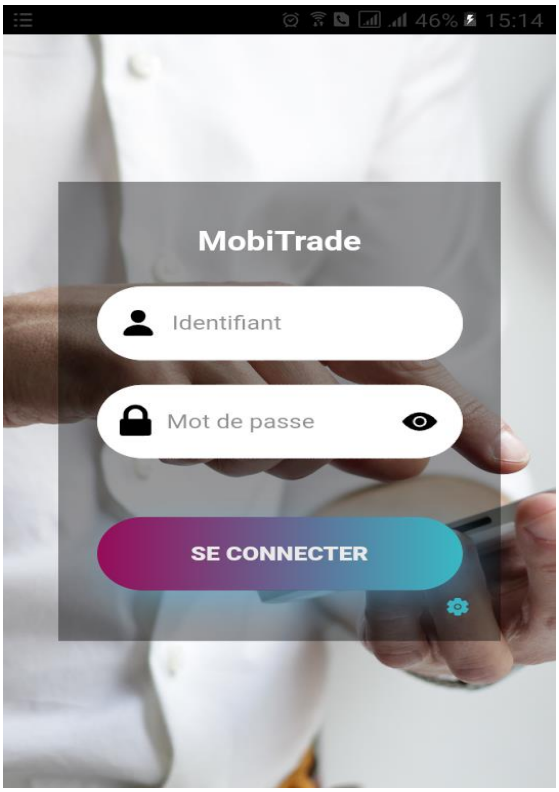


Figure IV.1: Interface "Authentification" **(Android)**

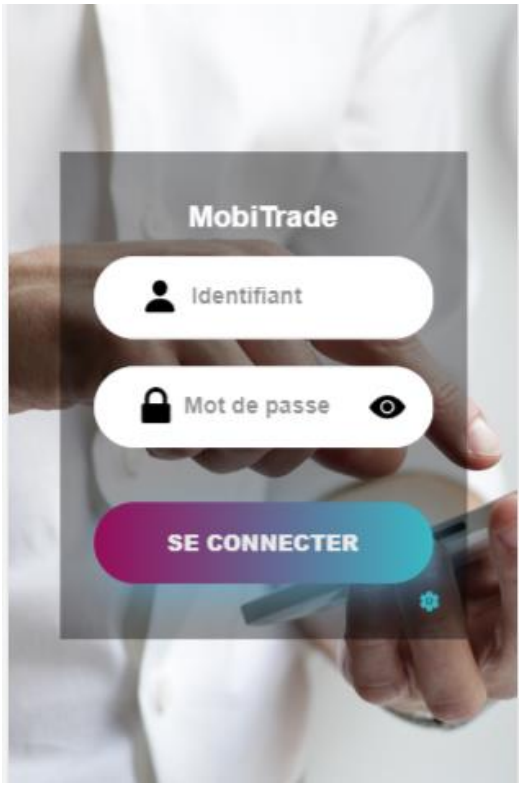


Figure IV.2: Interface "Authentification" **(IOS)**

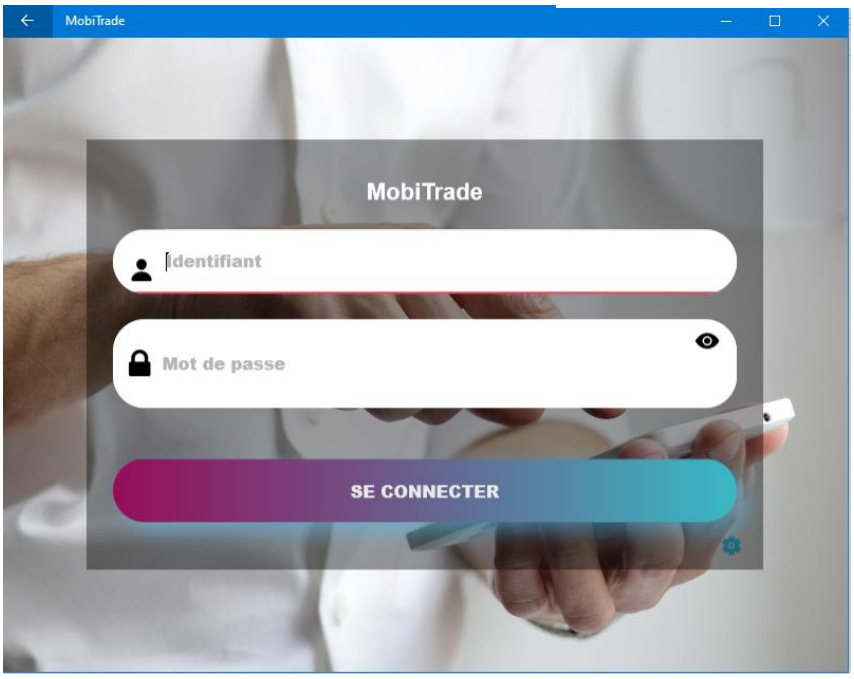


Figure IV.3: Interface "Authentification" **(Windows)**

Interface « Configuration serveur » :

On atteint cette page en cliquant sur l'icône du paramétrage dans la page d'authentification.

Dans cette interface l'utilisateur peut changer l'adresse du serveur soit vers une adresse IP local ou bien vers l'adresse du site hébergeant le coté serveur de l'application.

Figure IV.4: Interface "configuration serveur" **(Android)**

Figure IV.5: Interface "configuration serveur" **(IOS)**

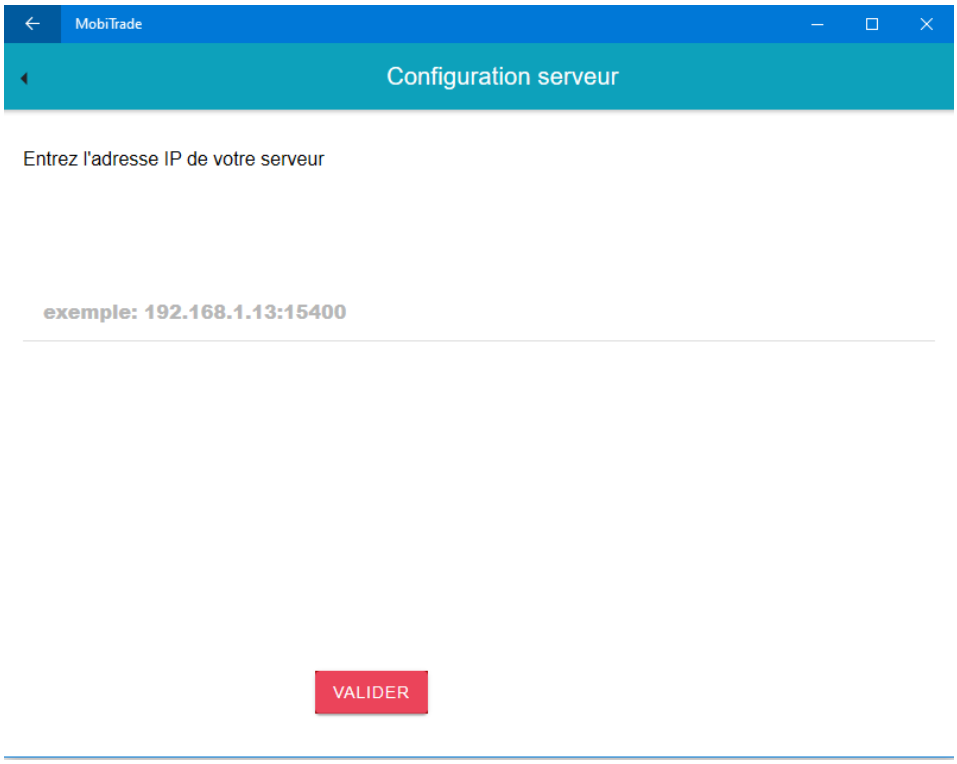


Figure IV.6: Interface "configuration serveur" **(Windows)**

Après avoir authentifié, l'utilisateur accède à son espace du travail à partir duquel il peut accéder aux différentes fonctionnalités en cliquant sur le lien correspondant.

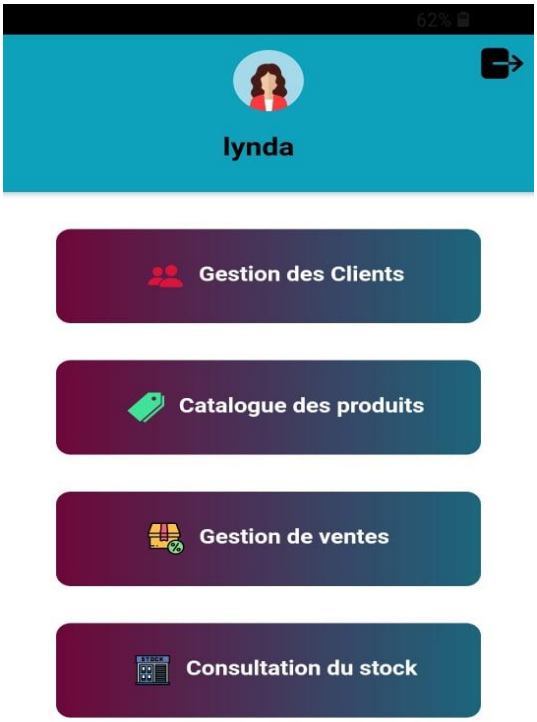


Figure IV.7: Interface "Accueil" **(Android)**

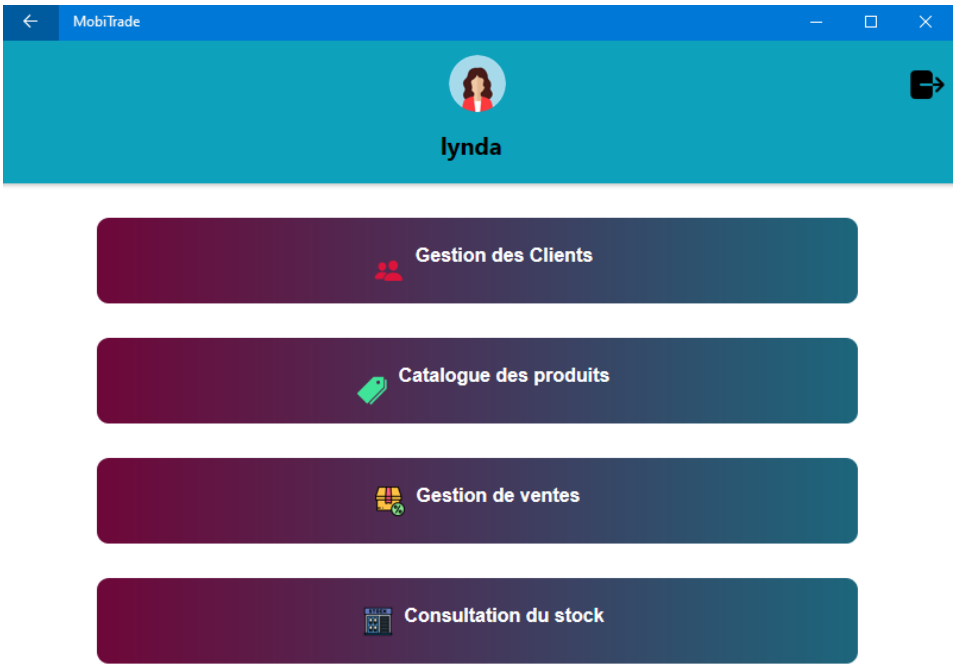


Figure IV.8: Interface "Accueil" **(Windows)**

Interface « Clients » de l'espace "Gestion des clients" :

Dans ce volet, on affiche la liste des clients de la société. La modification des informations d'un client se fait en cliquant sur l'icône de paramétrage.

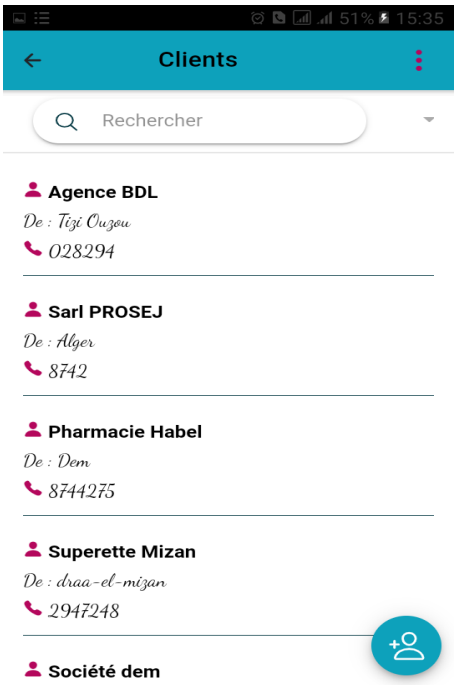


Figure IV.9: Volet "Client" **(Android)**

Figure IV.10: Volet "Client" **(Windows)**

Interface « Ajouter un Client » et « Modifier un Client »

- Une fois cliqué sur le bouton d'ajout, l'interface « **Ajouter un client** » s'affiche. En remplissant tous les champs du formulaire et cliquant sur l'icône de validation, le client sera ajouté .

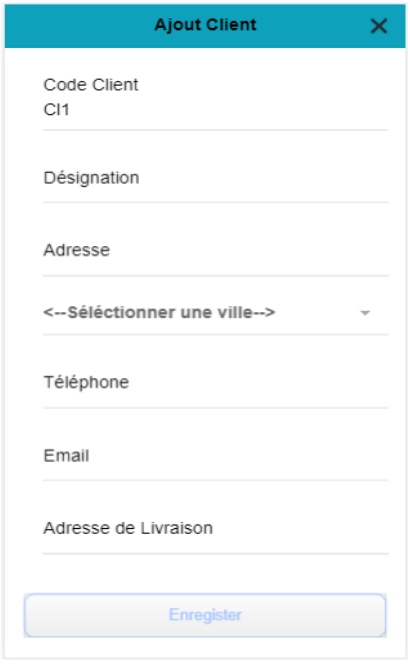


Figure IV.11: Interface "ajout client" **(IOS)**

- Une fois cliqué sur l'icône de modification, L'interface « **Modifier un client** » s'affiche , permet à l'utilisateur la mise à jour et la modification des informations déjà enregistrées sur le client.

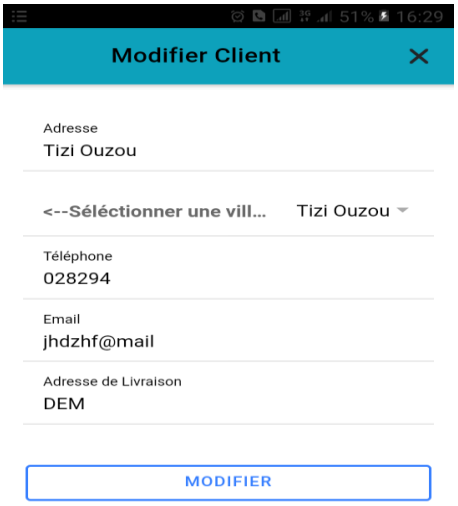


Figure IV.12: Interface "modifier client "**(Android)**

Volet "Devis "

Ce volet présente au gestionnaire des ventes la liste des devis.

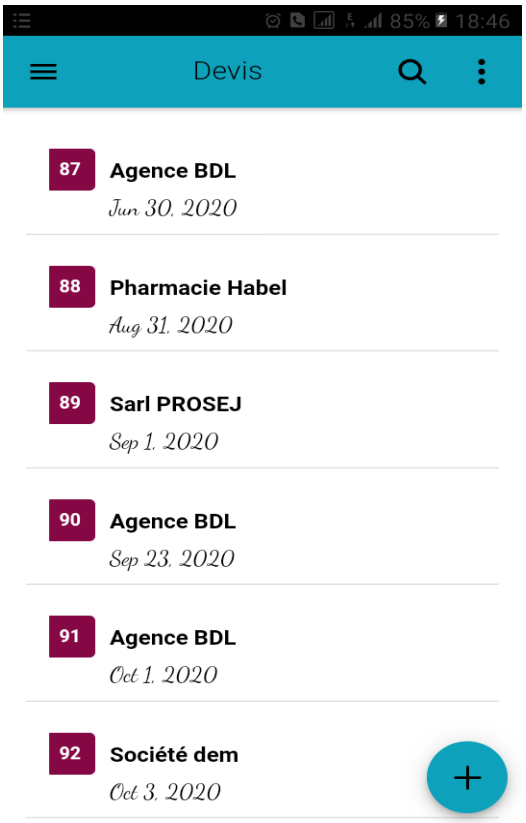


Figure IV.13: Volet "Devis" **(Android)**

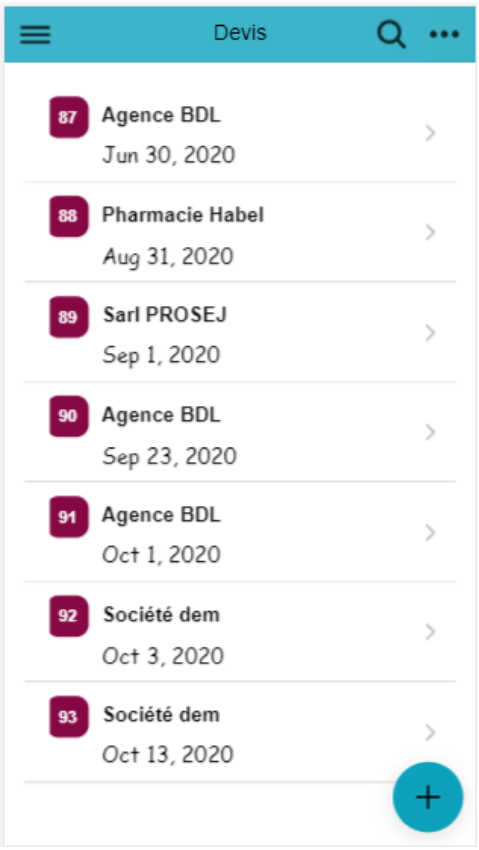


Figure IV.14: Volet "Devis" **(IOS)**

Interface « Ajout devis » :

Une fois cliqué sur le bouton d'ajout, le gestionnaire des ventes aura le formulaire d'ajout du devis qui s'affiche. En remplissant tous les champs du formulaire et en cliquant sur le bouton Enregistrer, le devis sera ajouté.

Figure IV.15: Interface "Ajout Devis" **(Android)**

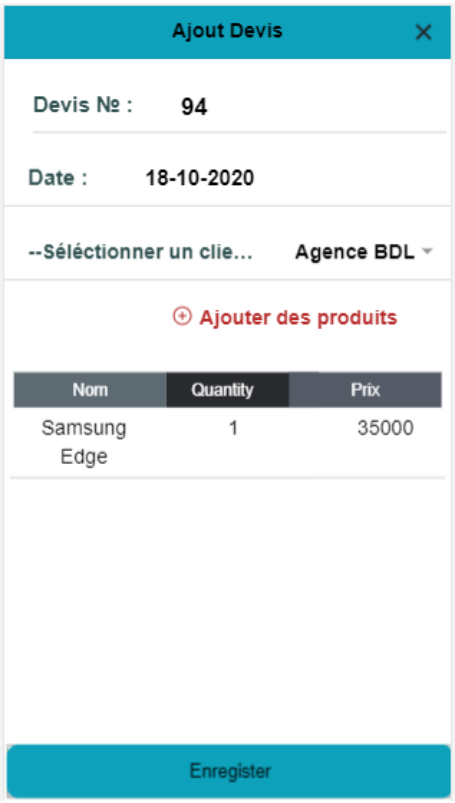


Figure IV.16: Interface "Ajout Devis" **(IOS)**

Figure IV.17: Interface "Ajout Devis" **(Windows)**

Interface « Ajout d'un produit dans un devis » :

Si le devis n'a pas encore été validé, le gestionnaire des ventes peut toujours ajouter des produits en remplissant les champs de l'interface et cliquant sur le bouton "Valider".

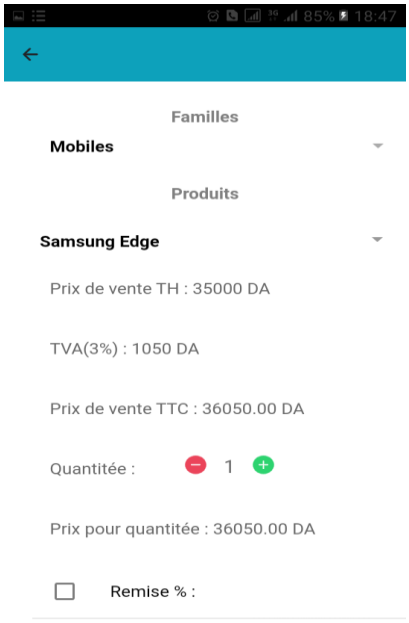


Figure IV.18: Interface "Ajout Produit" **(Android)**

Figure IV.19: Interface "Ajout Produit" **(iOS)**

Interface « Détails devis » :

Cette interface permet au gestionnaire des ventes de consulter les détails d'un devis établit préalablement, avec possibilité de modification si ce dernier n'est pas encore validé.

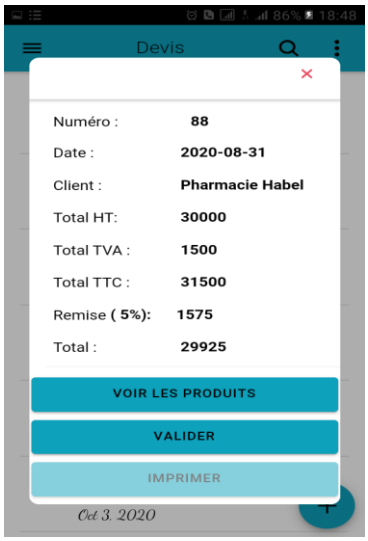


Figure IV.20: Interface "détails devis" **(Android)**

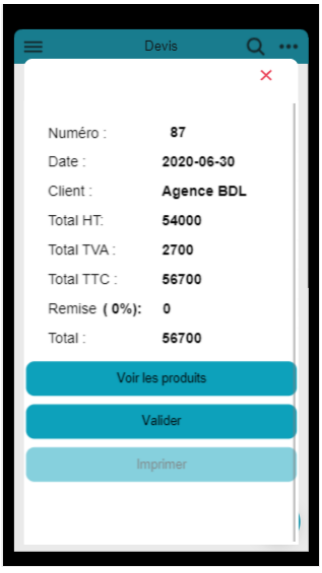


Figure IV.21: Interface "détails devis" **(IOS)**

Volet "Commandes ":

Ce volet affiche la liste des bons de commandes établis Préalablement par le gestionnaire des ventes.

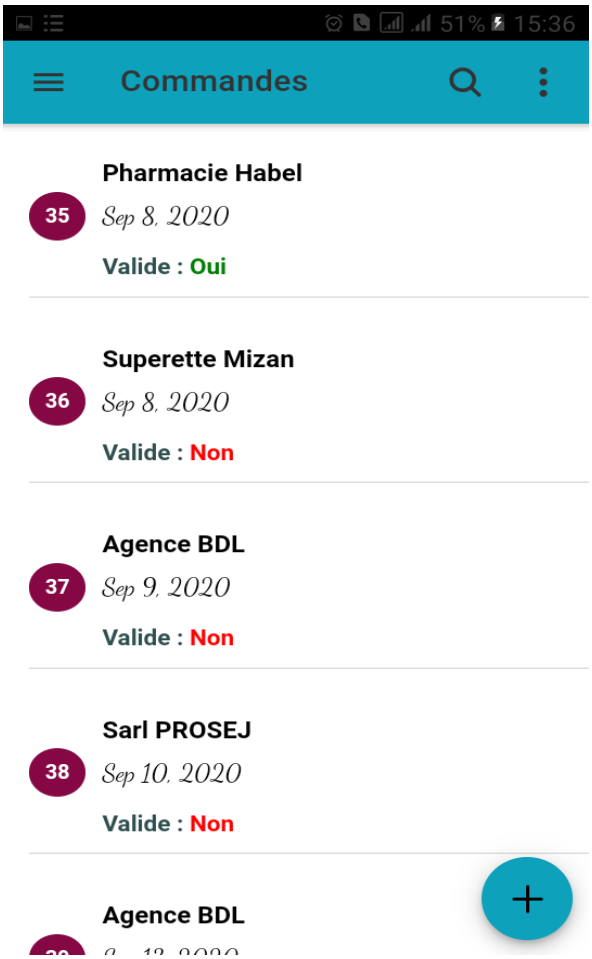


Figure IV.22: Volet "Commandes" **(Android)**

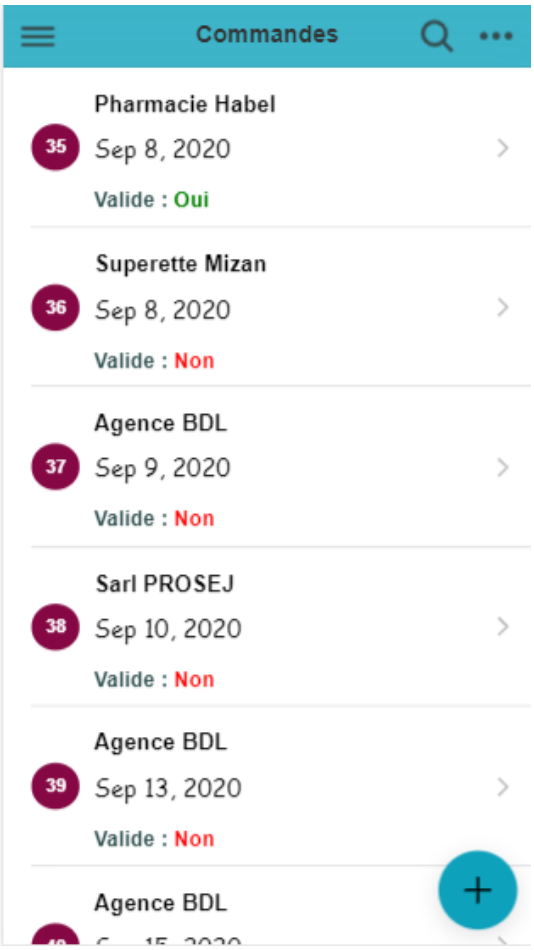


Figure IV.23: Volet "Commandes" **(IOS)**

Interface « Importer un devis » :

Cette interface permet lors d'établissement d'un bon de commande d'importer directement un devis. En sélectionnant un client et cliquant sur le bouton importer dans l'interface « **ajouter commande** » , une liste des devis faite par ce client est affichée et le gestionnaire des ventes peut en choisir.

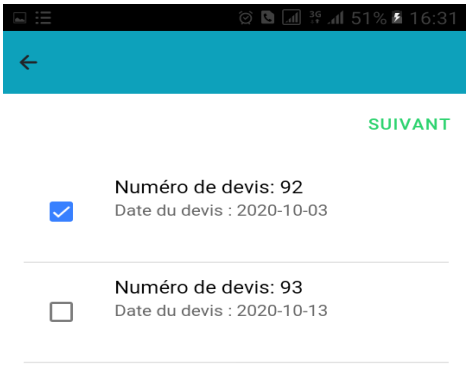


Figure IV.24: Importer devis **(Android)**

Volet "Bon de Livraison" :

Ce volet présente au gestionnaire des ventes la liste des bons de livraison, en cliquant sur le bouton d'ajout, il aura la possibilité d'ajouter un bon de livraison.

Figure IV.25: Volet "Bon de livraison" **(Android)**

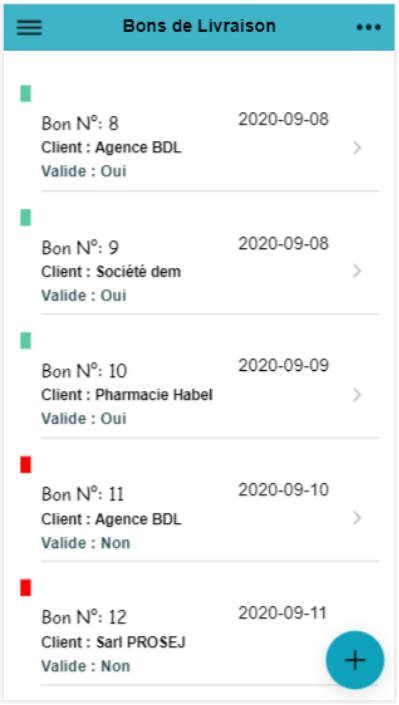


Figure IV.26: Volet "Bon de livraison" **(IOS)**

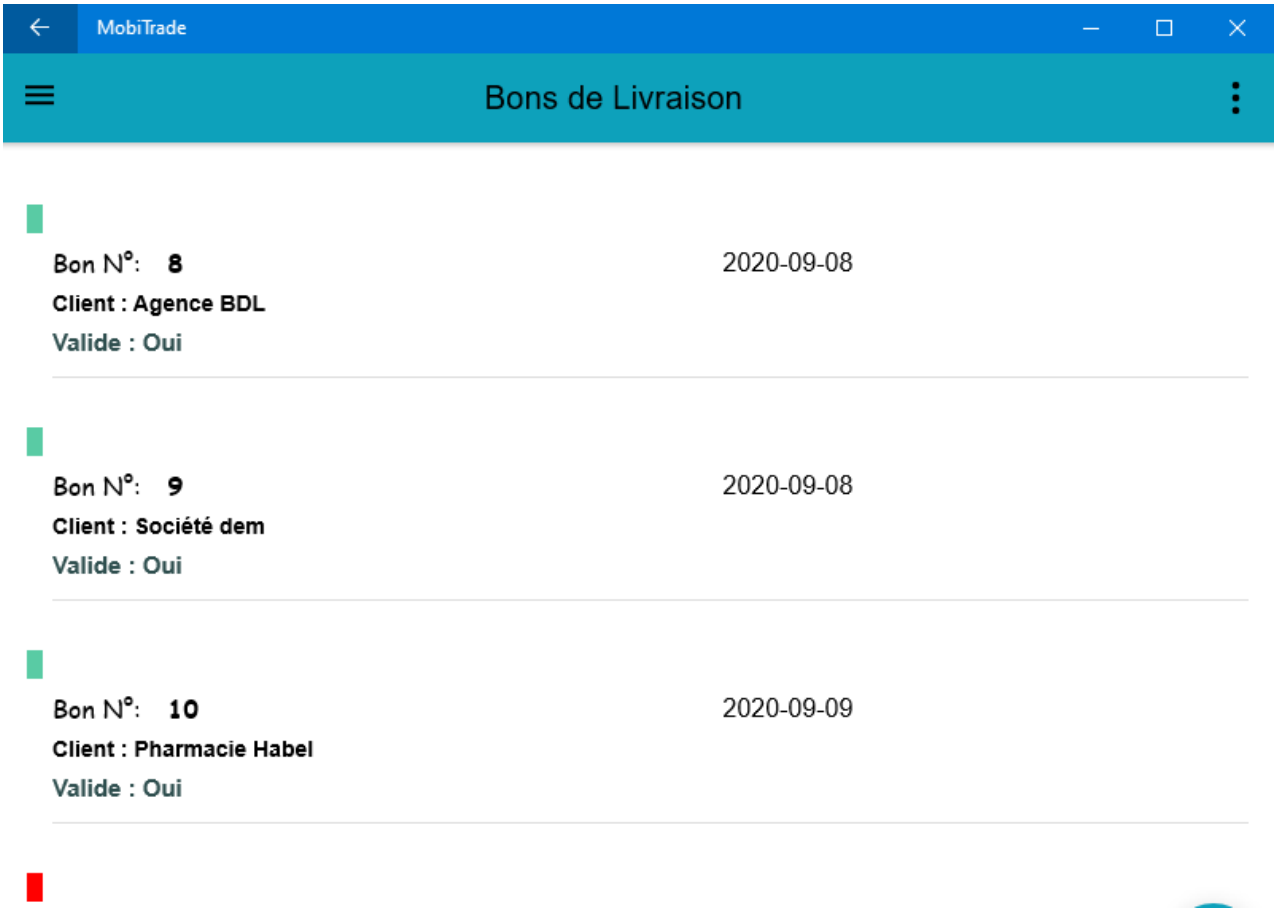


Figure IV.27: Volet "Bon de livraison" **(Windows)**

Interface « nouveau bon de livraison » :

Cette interface permet au gestionnaire des ventes d'éditer un nouveau bon, soit en important un bon de commande, pour ce fait, il clique sur "Importer", ou en ajoutant manuellement un bon de livraison et cela en cliquant sur "Nouveau".

Figure IV.28: Nouveau Bon de livraison **(Android)**

Figure IV.29: Nouveau Bon de livraison **(IOS)**

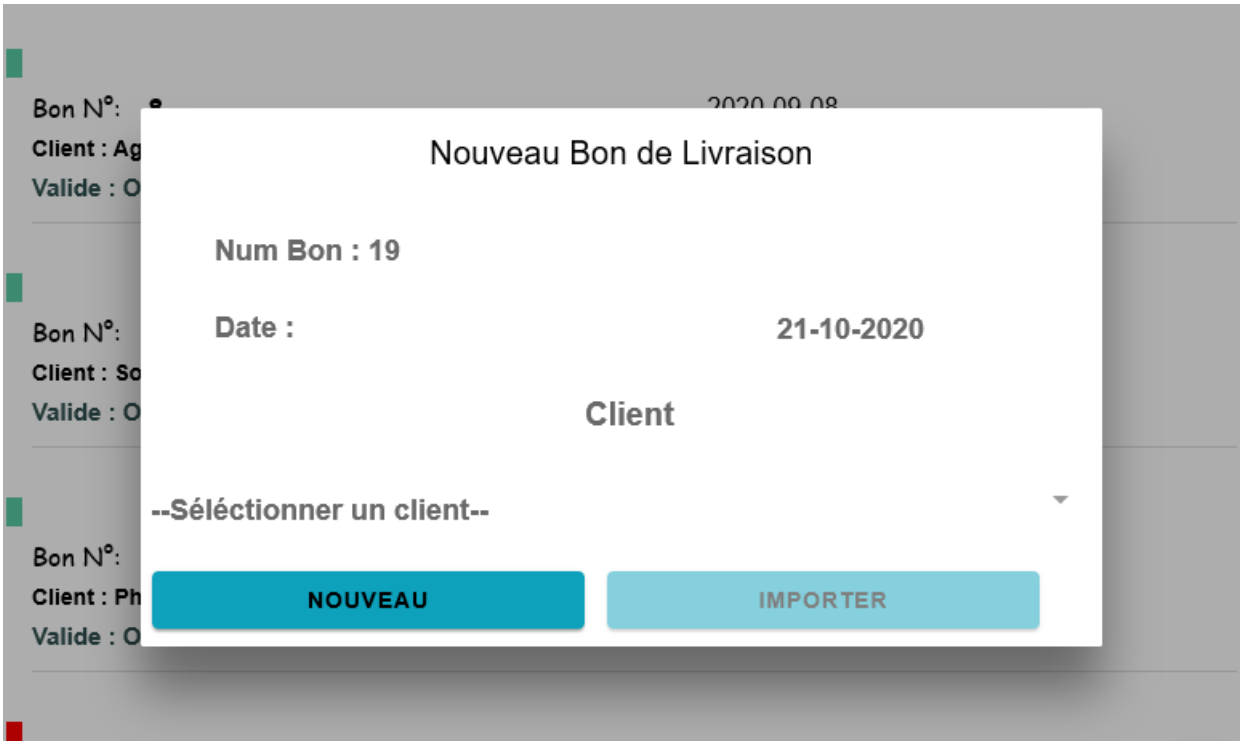


Figure IV.30: Nouveau Bon de livraison **(Windows)**

Interface d'établissement d'une facture :

Après l'atteinte du volet "Facture", l'interface "nouvelle facture " permet au gestionnaire des ventes d'établir une nouvelle facture en cliquant sur le bouton d'ajout, en choisissant le client à facturer et en cliquant sur le bouton "Importer".

Figure IV.31: Interface Nouvelle Facture **(IOS)**

L'interface nous donnera accès à tous les bons de livraison non facturés du client choisit, qu'on peut sélectionner et regrouper en une seule facture (Voir figure **IV.32) .**

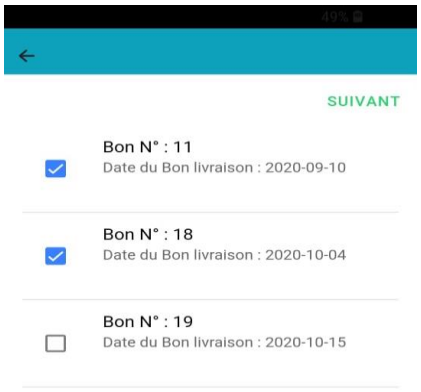


Figure IV.32 : Interface de sélection des bons de livraison à facturer.

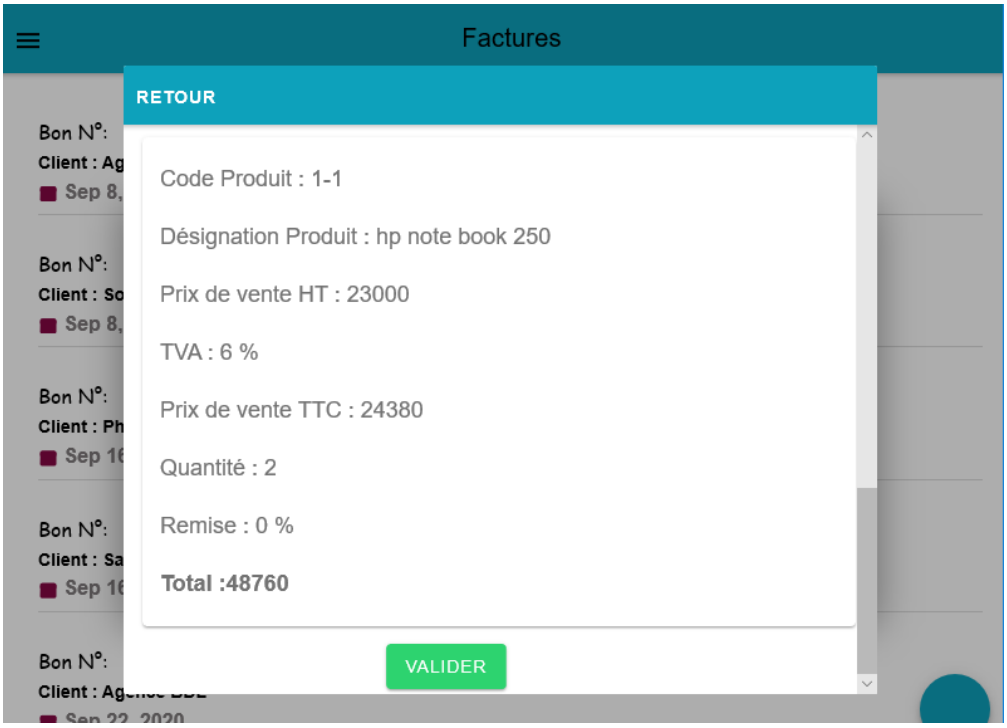


Figure IV.33 : Affichage de la facture. **(Windows)**

Interface « Paiement Facture » :

En cliquant sur une facture, une interface de détails de facture est affiché , le bouton Paiement permet d'enregistrer un nouveau paiement en remplissant les champs du formulaire de l'interface « Paiement Facture » .

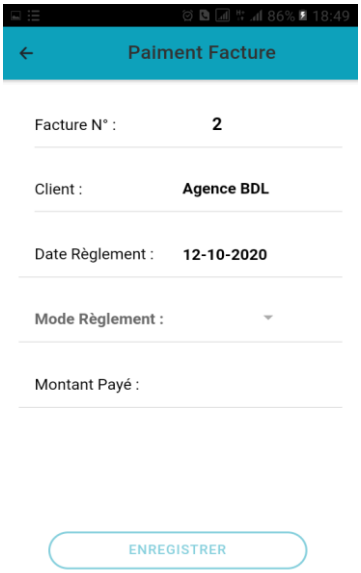


Figure IV.34: Interface Paiement Facture **(Android)**

Interface « Produits » dans l'espace « catalogue produits » :

L'utilisateur peut consulter la liste des produits qui existent et peut rechercher un produit en cliquant sur la barre de recherche.

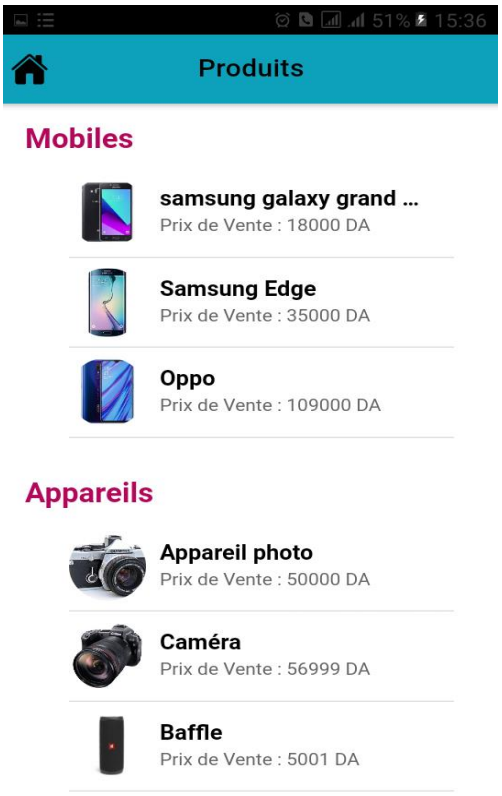


Figure IV.35: Interface Produits **(Android)**

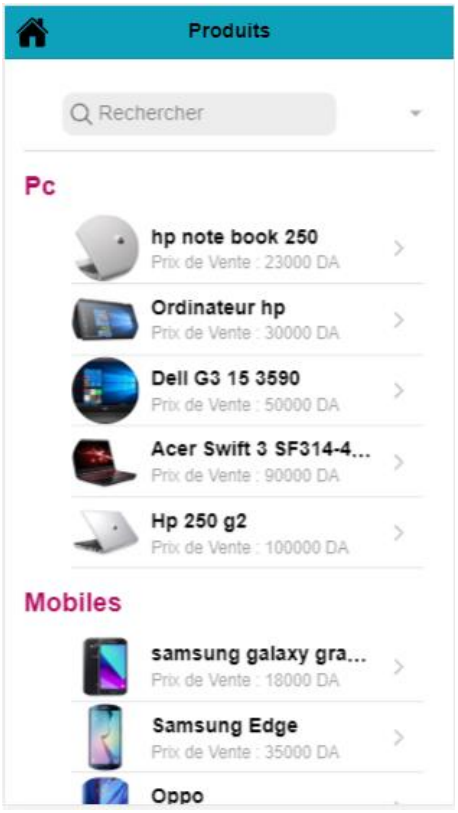


Figure IV.36: Interface Produits **(IOS)**

Volet "Stock " :

Ce volet permet au gestionnaire des ventes de consulter la disponibilité des produits en stock en terme de quantité.

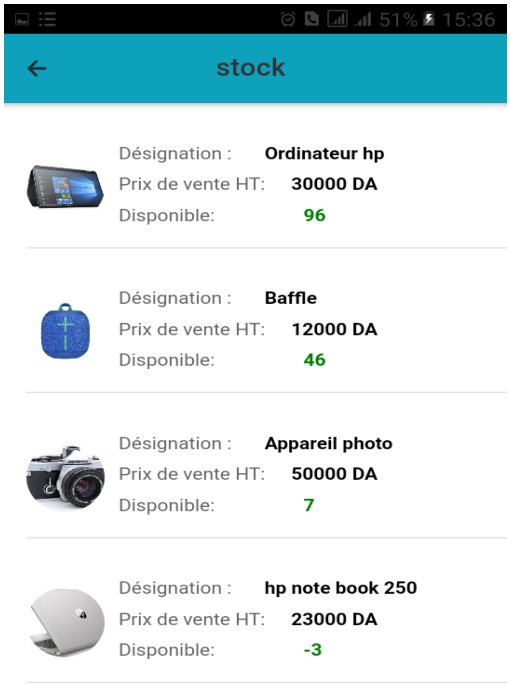


Figure IV.37: Interface Stock **(Android)**

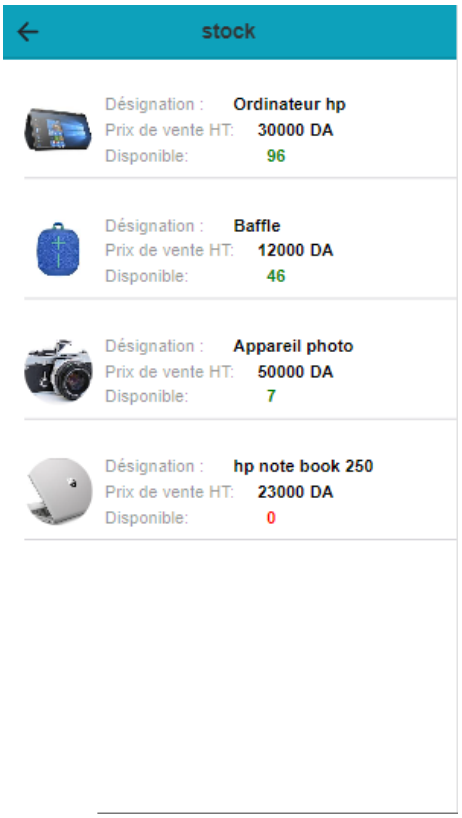


Figure IV.38: Interface Stock **(IOS)**

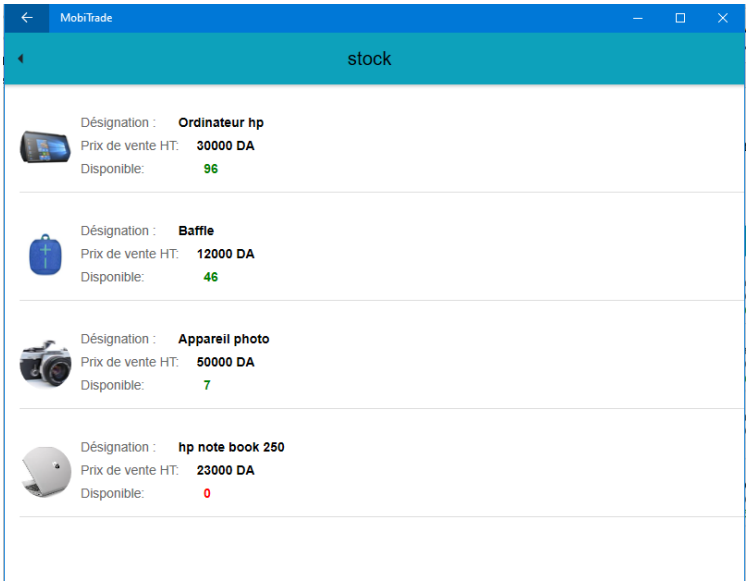


Figure IV.39 : Interface Stock **(Windows)**

IV.5. Conclusion

Ce dernier chapitre a été consacré à la présentation de l'étape réalisation de notre application ainsi, nous avons présenté les outils logiciels qui nous ont permis la réalisation de notre travail à savoir l'environnement de développement et les langages de programmation.

Puis, nous sommes passés aux présentations de notre application en décrivant ses fonctionnalités et présentant plusieurs interfaces.

Conclusion générale

L'objectif de notre travail était de concevoir et réaliser une application mobile multiplateforme pour la gestion de ventes, qui nous permettra de décentraliser certaines fonctionnalités du logiciel existant ILUTRADE conçu par la société de service MS CONTACT de Tizi-Ouzou.

Pour mener à terme notre projet, nous avons donné un aperçu général sur les systèmes d'exploitation mobiles et les technologies qui permettent de réaliser des applications multiplateformes en citant quelques exemples de ces applications .

Ensuite, nous avons présenté brièvement la société de service MS CONTACT de Tizi-Ouzou qui est notre organisme d'accueil, ainsi que le logiciel ILUTRADE afin de cerner la problématique, ce qui nous a conduits à en tirer les objectifs à atteindre dans notre application.

L'Application à laquelle nous avons aboutie, répond à plusieurs fonctionnalités, telles que : l'établissement des devis, bons de commande, bons de livraison et les factures des clients. L'ajout et la modification d'un client, la consultation de stock et le paiement de la facture.

Pour cette réalisation, nous avons pris connaissance de plusieurs domaines tels que : les réseaux, les accès distants aux bases de données ainsi que leurs manipulations et le modèle client/serveur. Il nous a aussi permis de nous familiariser avec un certain nombre d'outils informatiques et de développement, tels que:

Ionic avec le langage Angular pour la création des interfaces et le design, le JavaScript pour les traitements et les interactions, PhpMyAdmin avec le langage PHP pour la communication des données et la connexion à la base de données. Ainsi que IonicServe et PostMan pour les différents tests.

Les perspectives

- ✓ Evaluer le prototype avec les clients.
- ✓ Intégrer l'application au logiciel existant (Ilutrade) .
- ✓ Ajouter un module de géolocalisation pour localiser les clients.

Références Bibliographique

[1] http://www.uky.edu/~jclark/mas490apps/History%20of%20Mobile%20Apps.pdf.

[2] http://www.universalis.fr/encyclopedie/systemes-d-exploitation-informatique/

[3] https://fr.wikipedia.org/wiki/Android

[4] http://www.techno-science.net/?onglet=news&news=11005.

[5]https://www.geeek.org/apache-cordova-la-future-generation-d-applications-mobiles 545.html

[6] Antoine C.D., « React Native : le framework JavaScript de Facebook au crible », article scientifique sur le framework react-native, 14/04/2016. http://www.journaldunet.com/web-tech/developpeur/1176787-react-native-la-declinaison-de-react-pour-developper-des-apps-natives/

[7] https://blog.codecentric.de/en/2014/11/ionic-angularjs-framework-on-the-rise/

[8] http://dspace.univ-tlemcen.dz/bitstream/112/12126/1/Developpement-dun-atelier-graphique-pour-la-generation-dapplications-mobiles-multi-plateforme.pdf

[9]http://www.univbejaia.dz/jspui/bitstream/123456789/11800/1/L%E2%80%99analyse%20de%20l%27efficacie/gestion-de-vente.pdf

[10] https://d1n7iqsz6ob2ad.cloudfront.net/document/pdf/5385ae030040d.pdf

[11] Guide de ILUTRADE, tirer du site: https://www.mscontact.com

[12]"Visual Studio Code - Code Editing. Redefined." https://code.visualstudio.com/ (accessed Oct. 15, 2020).

[13] "Wikipédia, l'encyclopédie libre." https://fr.wikipedia.org/wiki/Wikipédia:Accueil_principal (accessed Oct. 14, 2020).

[14] "XAMPP Installers and Downloads for Apache Friends." https://www.apachefriends.org/fr/index.html (accessed Oct. 15, 2020).

[15] "https://www.phpmyadmin.net/." consulté le 10 octobre 2020.

[16] http://dspace.univ-tlemcen.dz/bitstream/112/13466/1/MarketPlace-Developpement-dune

application-mobile..pdf

[17] Eric Despet, Cyril Pierre de Gever, "Php5 avancé" 4e édition.

[18] C. Soutou. *Apprendre SQL avec MySQL*. Paris, 2006.

[19] "Ionic - Cross-Platform Mobile App Development." https://ionicframework.com/ (accessed Oct. 14, 2020).

[20] "Angular - Introduction to the Angular Docs." https://angular.io/docs (accessed Oct. 14, 2020).

[21] "TypeScript: Typed JavaScript at Any Scale." https://www.typescriptlang.org/ (accessed Oct. 14, 2020).

[22] *XML*. Date de consultation 06 2013. http://www.futura-sciences.com/fr/definition/t/internet-2/d/xml_3997/

[23] Z. Moussouni and M. Ramdani, "Conception et réalisation d'une application mobile pour le service de tourisme , cas d'étude " Wilaya de Bejaia "," Béjaia, 2017.

[24] http://www.maxicours.com/se/fiche/5/4/230354.html

[A] Marwa B., « développement mobile hybride Ionic», présentation de mémoire master II, université Hassan II Maroc, 27/03/2017. https://fr.slideshare.net/marwabaich1/ionic-angularjscordovanodejssass

[B] Olfa-Hamrouni, « généralité sur le développement d'applications mobiles ». Présentation cours, 2006. http://www.isetjb.rnu.tn/jwmu2017/docs/Developpement-Mobie-OLFA-HAMROUNI.pdf

[C] Vincent, « Cordova : Applications mobiles hybrides », article sur cordova, ionic, phonegap, hybride, . https://vincent-g.fr/2016/02/25/ionic-apache-cordova-developpement-mobile-hybride/

[D] Cihad h., « Hybrid Apps with Angular & Ionic Framework».Présentation cours,Istanbul Coders, 2014. https://www.slideshare.net/cihadhoruzoglu/hybrid-apps-with-angular-ionic-framework

[E] Ripkens B., «Ionic: An AngularJS based framework on the rise». Article scientifique, 28/11/2014. https://blog.codecentric.de/en/2014/11/ionic-angularjs-framework-on-the-rise/

[F] http://www.zdnet.fr/actualites/chiffres-cles-les-os-pour-smartphones-39790245.html

Annexes

I. **Annexe A**: Présentation d'UML

Unified Modeling Language est un langage unifié de modélisation objets. Ce n'est pas une méthode . C'est un ensemble d'outils pour aide la modélisation de la future des applications informatique. UML c'est une méthode utilisant des graphismes pour la création de modèles orientés objet vers de la conception et de modélisation de logiciels orientés objet.

A quoi sert UML ?

Pour bon développement d'un logiciel orienté objet l'UML offre les moyens pour spécifier, visualiser, modifier et construire les documents nécessaires du développement à partir d'un standard de modélisation.
Les différents éléments sont:

- Acteurs.
- Composants logiciels.
- Réutilisation de composants.
- Processus.
- Schéma de base de données.

UML utilise l'approche objet en présentant un langage de description universel. Il permet grâce à un ensemble de diagrammes très explicites, de représenter l'architecture et le fonctionnement des systèmes informatiques complexes en tenant compte des relations entre les concepts utilisés et l'implémentation qui en découle.

Caractéristiques du langage de modélisation UML

- **UML comme une conception complète d'un système**

UML offre déférente représentation d'un système selon différentes vues grâce aux diagrammes. [9]

- **UML moyen l'analyse objet**

UML propose différentes vues d'un système, pour guider son utilisation ainsi que plusieurs niveaux d'abstractions qui aide contrôler la complexité dans l'expression des solutions orienté.

- **UML est un soutien de communication**

La notation graphique d'UML pour d'exprimer une solution objet, tandis que l'aspect formel de ses notations limite les ambiguïtés et les incompréhensions. [17]

Que ce qu'un diagramme UML ?

Un diagramme UML est une représentation graphique, qui s'intéresse à un aspect précis du modèle ; c'est une perspective du modèle.

Chaque type de diagramme UML possède une structure (les types des éléments de modélisation qui le composent sont prédéfinis) et véhicule une sémantique précise (il offre toujours la même vue d'un système).

Les différents types de diagrammes UML offrent une vue complète des aspects statiques et dynamiques d'un système. Les diagrammes permettent donc d'inspecter un modèle selon différentes perspectives et guident l'utilisation des éléments de modélisation (les concepts objet), car ils possèdent une structure.

Les différents diagrammes de l'UML

Il existe treize types de digrammes UML dans l'UML 2.0 ses diagrammes fait des concepts particuliers du système d'information, Ils se répartissent en trois catégories :

Diagrammes structurels

- **Diagramme de classes**

Le diagramme de classe représente les entités manipulées par les utilisateurs c'est le diagramme le point central dans le développement orienté objet et le plus utilisé il présenter les types d'objets et les relations entre aux **[13].**

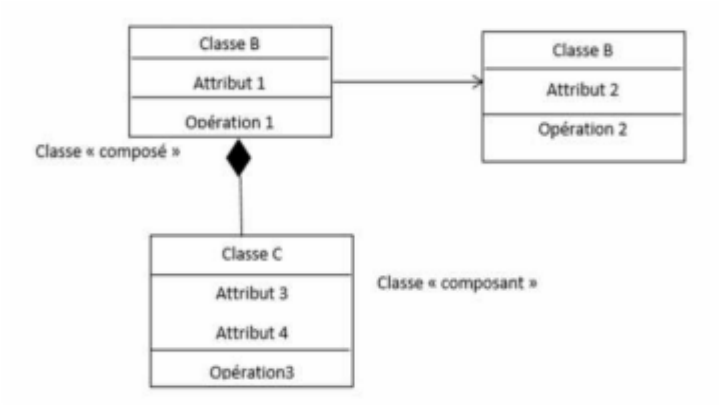


Figure A.1 : Diagramme de classe.

- **Diagramme d'objets**

Permet de représenter les objets et les relations entre eux à partir les lient nécessaire.

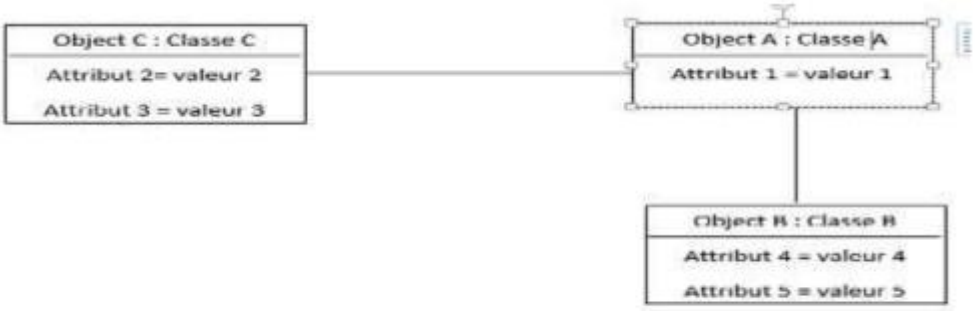


Figure A.2 : Diagramme de composant

- **Diagramme de composants**

Collection des classes ou des composants pour décomposer le système en parties de logiciel, donc Il montre des structures complexes avec leurs interfaces fournies et requises **[14].**

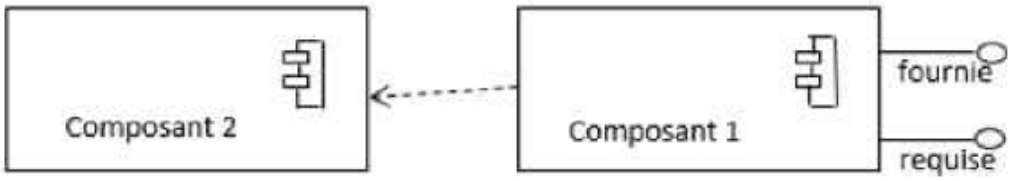


Figure A.3 : Diagramme de composant

➢ **Diagramme de déploiement**

Ce diagramme décrit l'architecture technique d'un système avec une vue centrée sur la répartition des composants dans la configuration d'exploitation. **[11].** Donc montre le déploiement physique des artefacts sur les ressources matérielles **[8].**

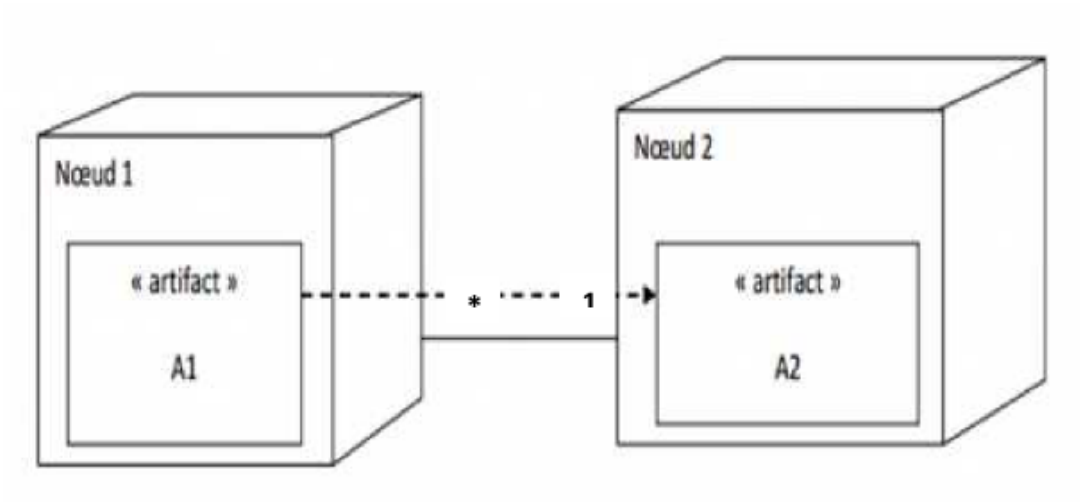


Figure A.4 : Diagramme de déploiement

Diagrammes comportementaux ou diagrammes dynamiques

➢ **Diagramme de cas d'utilisation**

Permet la représentation des fonctionnalités nécessaires aux utilisateurs. On peut faire un diagramme de cas d'utilisation pour le logiciel entier ou pour chaque package, ce diagramme clarifié comment les utilisateurs externes (acteur), dialoguer avec ces cas d'utilisation **[11].**

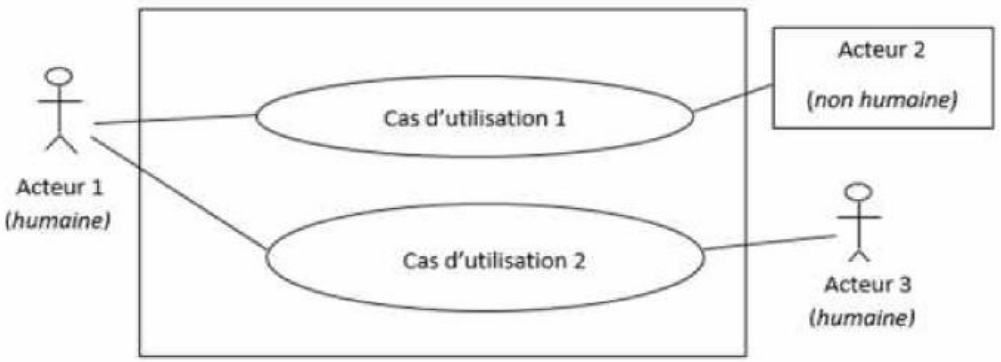


Figure A.5 : Diagramme de cas d'utilisation .

➢ **Diagramme d'états-transitions**

Le diagramme d'états transition permet de décrire le passage d'états d'un objet à un autre état s'appelle transition, il permet aussi décrire le comportement d'un objet. **[16]**

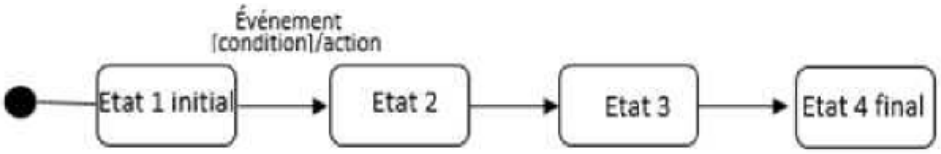


Figure A.6 : Diagramme d'état transition.

➢ **Diagramme d'activités**

Le diagramme d'activité utilisé pour approfondir sur le comportement interne d'une méthode, ou pour concaténer les activités ou le déroulement des cas d'utilisation, et permet la spécification des actions de bases, les structures de contrôle et les instructions interagissent à la programmation orienté objet.

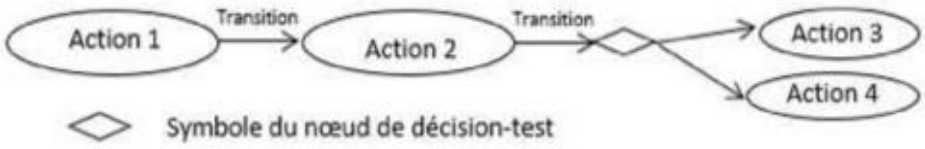


Figure A.7 : Diagramme d'activité.

Diagrammes d'interaction

➢ **Diagramme de séquence**

Permet décrire les interactions entre les objets d'un système selon un ordonnancement temporel, cette interaction fait par l'envoi de messages (message synchrone ou message asynchrone), qui appelle une méthode.

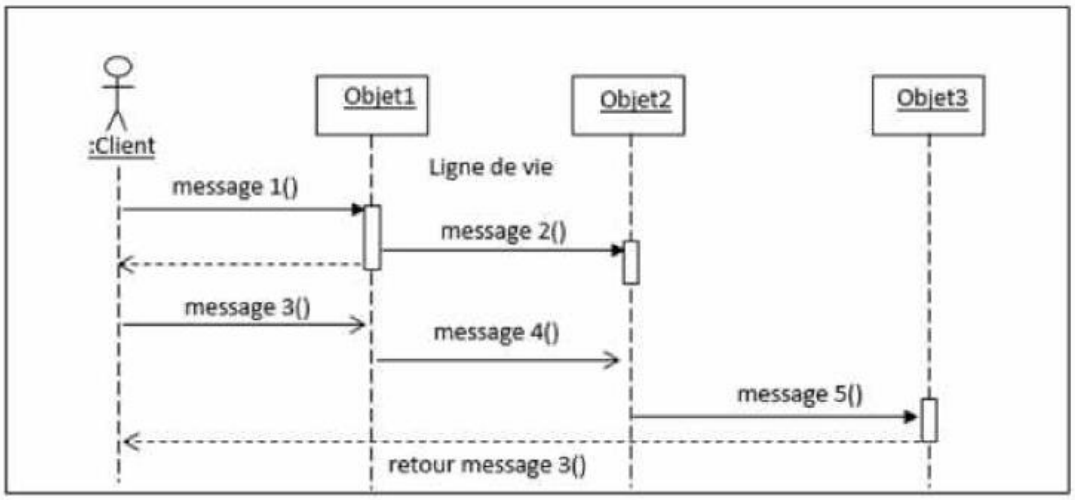


Figure A.7 : Formalisme d'un diagramme de séquence.

- **Diagramme de communication**

Le diagramme de communication est une représentation d'objet en relation avec déférente objets et l'échange des messages dans le diagramme de séquence se représente comme un rôle dans le diagramme communication décrit comme suit : <nom de rôle> : <nom du type> **[11] .**

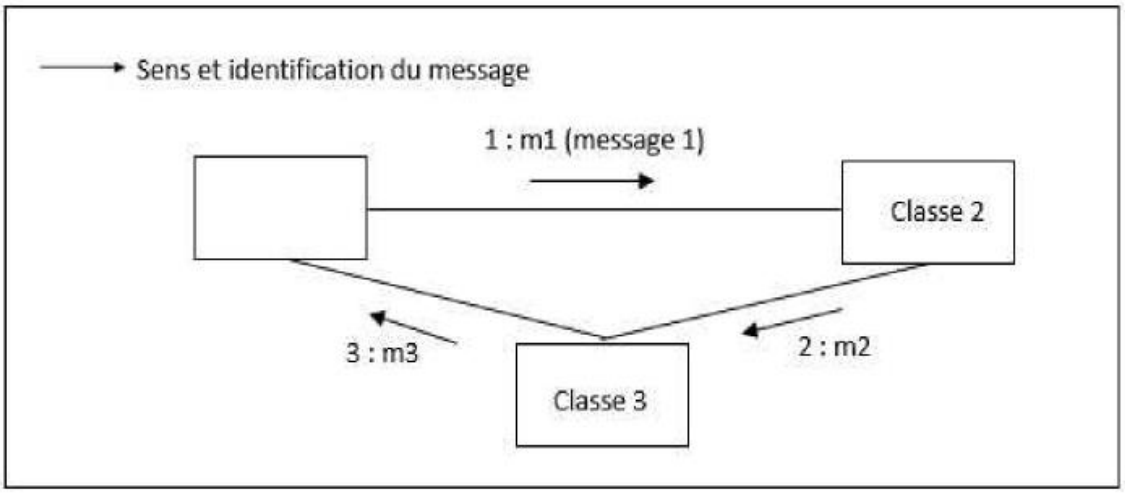


Figure A.8 : Diagramme de communication.

Avantages et inconvénients d'UML

- **Les avantages**

- ✓ UML est un langage formel et normalisé.
- ✓ Offre une Exactitude rigoureuse.
- ✓ Facilité l'utilisation d'outils.
- ✓ Garantie un aide pour la communication performant.
- ✓ Aisé à comprendre la personnification abstraite composite et complexes.
- ✓ Offre un milieu d'analyse les besoins.
- ✓ UML c'est un langage universel.

- **Les inconvénients**

- ✓ La mise en pratique d'UML nécessite un apprentissage et passe par une période d'adaptation.

- ✓ Le processus (non couvert par UML) est une autre clé de la réussite d'un projet. Or, l'intégration d'UML dans un processus n'est pas triviale et améliorer un processus est une tâche complexe et longue. Les auteurs d'UML sont tout à fait conscients de l'importance du processus, mais l'acceptabilité industrielle de la modélisation objet passe d'abord par la disponibilité d'un langage d'analyse.

II. **Annexe B** : Les Réseaux

En informatique, un réseau désigne une série de machines ou nœuds Interconnectés par des chemins de communication. Les réseaux peuvent eux-mêmes s'interconnectés à d'autres réseaux et contenir des sous-réseaux.

Les topologies de réseau, ou configurations générales, les plus courantes sont le bus, en étoile, Token Ring et le maillage.

Les réseaux peuvent également être décrits en termes d'étendue géographique : réseau local (LAN, Local Area Network), urbain (MAN, Metropolitan Area Network) ou étendu (WAN, Wide Area Network).

Un réseau se caractérise en outre par le type de technologie ou de protocole de transmission des données qu'il utilise, par exemple, un réseau TCP/IP ou SNA (Systems Network Architecture) ; par les types de signaux qu'il transporte (voix, données, les deux, etc.) ; par l'usage qui en est fait (réseau public ou privé) ; par la nature habituelle de ses connexions (réseau commuté, spécialisé ou à connexions virtuelles) ; et enfin par les types de liaisons physiques (fibre optique, câble coaxial, paire torsadée non blindée, réseau hertzien - ou sans-fil).

- ## Architecture client/serveur

1.Définition

L'architecture client-serveur est un mécanisme de communication entre deux ou plusieurs ordinateurs via des protocoles. Cette architecture est basée sur l'utilisation de deux types de logiciels à savoir un logiciel serveur et un logiciel client s'exécutant sur deux machines différentes. [24]

Le dialogue entre eux peut se résumer par :

- ✓ Le client demande un service au serveur.
- ✓ Le serveur réalise ce service et renvoie le résultat au client.

Comme le montre le schéma suivant :

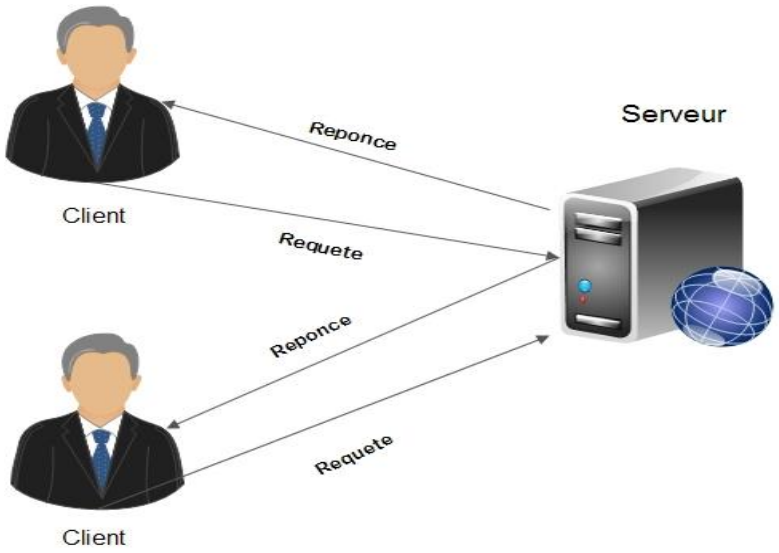


Figure B.1 : schéma de fonctionnement du système client/serveur.

On peut trouver plusieurs types d'architecture Client/serveur, comme :

2. Architecture à 2 niveaux (2-tiers)

L'architecture à 2-tiers (tiers signifiant rangée en anglais) est l'architecture la plus classique, elle décrit les systèmes Client/serveur dans lesquels un serveur exécute la requête du client et fournit directement le service, sans faire appel à d'autres intermédiaires, en utilisant ses propres ressources, le dialogue entre client et serveur se résume donc à l'envoi de requêtes et au retour de données correspondants à celles-ci.

L'architecture à deux niveaux est schématisée dans la figure (B.2)

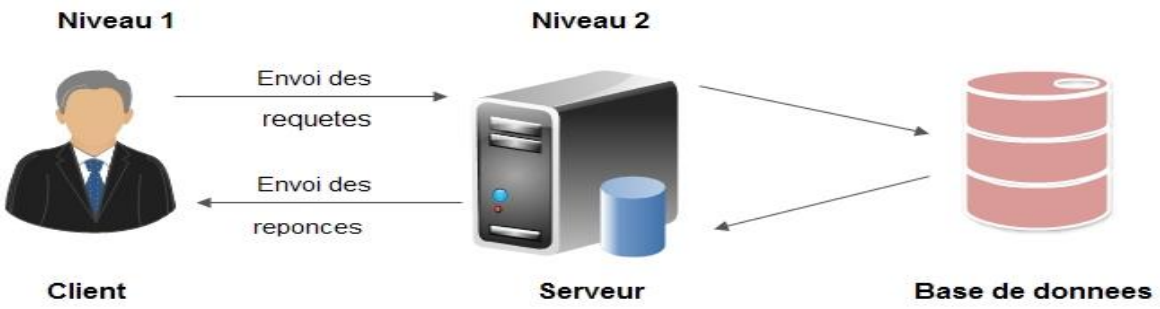


Figure B.2 : Architecture Client / Serveur à deux niveaux.

3. Architecture à 3 niveaux (3-tiers)

Elle est également appelée client/serveur de deuxième génération, dans ce type d'architecture existe un niveau intermédiaire, c'est-à-dire que l'on a généralement une architecture partagée entre :

- ✓ **Premier niveau :** c'est le poste client qui est l'ordinateur demandeur de ressources, équipé d'une interface utilisateur (un navigateur web) chargée de la présentation (contrôle de saisie, mise en forme de données…).
- ✓ **Deuxième niveau :** c'est le serveur d'application (appelé également middleware), chargé de fournir la ressource mais faisant appel à un autre serveur.
- ✓ **Troisième niveau :** c'est le serveur de base de données (le serveur secondaire), fournissant au serveur d'application les données dont il a besoin pour retourner directement la réponse vers le demandeur.

L'architecture à trois niveaux est schématisée dans la figure (B.3) .

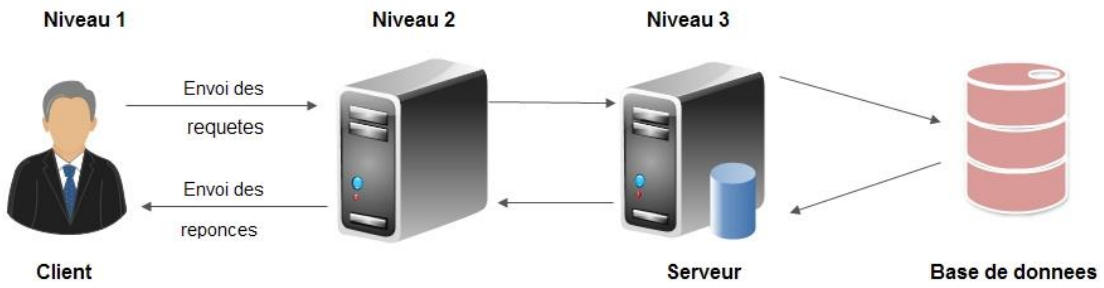


Figure B.3: Architecture Client/serveur à trois niveaux

4. Architecture multi niveaux (n-tiers) :

L'architecture n-tiers appelée aussi architecture distribuée ou architecture multi-tiers, elle permet de concevoir des applications puissantes et simples à maintenir.

Ce type d'architecture facilite la répartition de la charge entre tous les niveaux, cela signifie qu'un serveur peut utiliser les services d'un ou plusieurs autres serveurs afin de fournir son propre service [24].

▪ Architecture matérielle

L'architecture de notre application est à 3 niveaux (architecture 3-tiers), elle est partagée entre:

- **Le client mobile** : Conteneur d'application et demandeur de ressources.
- **Le serveur Web** : Vu que les données seront communiquées entre deux environnements hétérogènes, le rôle principal du serveur web est de gérer la communication entre le client mobile et le serveur de base de données,
- **Le serveur de l'entreprise** : qui est le serveur de base de données qui fournit les données au serveur web.

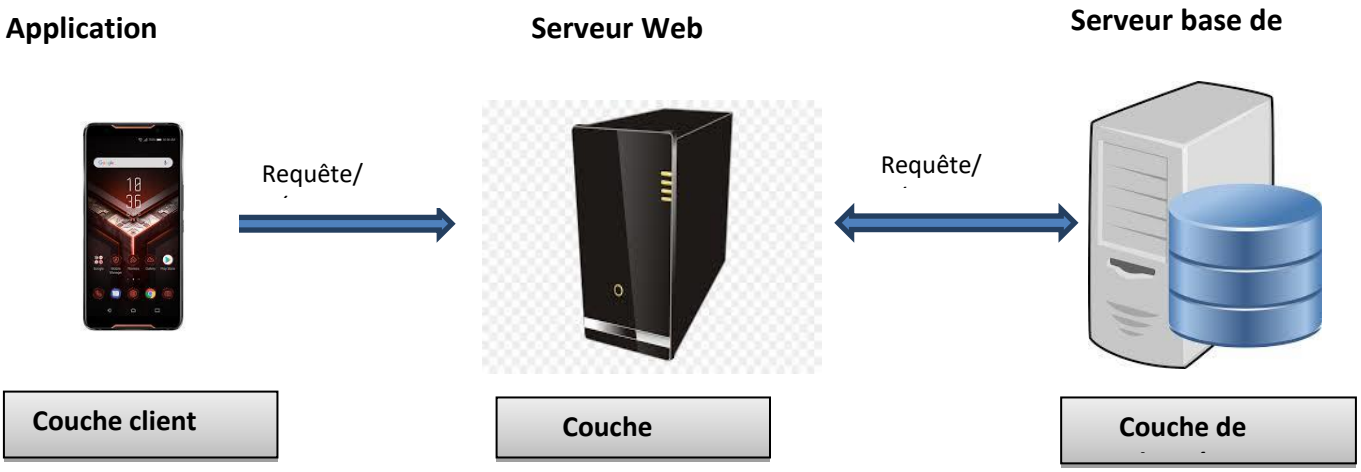


Figure B.4: Architecture matériel du système.