MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE TUNIS EL MANAR

INSTITUT SUPERIEUR D'INFORMATIQUE

RAPPORT DE STAGE DE FIN D'ETUDES

Présenté en vue de l'obtention du

Diplôme National de Licence Fondamentale en Sciences et Technologies

Mention : Sciences de l'Informatique

Spécialité : Sciences de l'informatique

Par

Nader SOMRANI & Wajdi KHATTEL

Développement d'une application mobile de gestion bancaire

Encadrant professionnel **Monsieur Amir SOMRANI**
Encadrant académique **Monsieur Sami OUALI**

Réalisé au sein de BFI

Année Universitaire 2016/2017

Dédicace

Je dédie ce travail à toute ma famille.
Mes parents qui m'ont soutenu toute ma vie, je vous remercie de tout mon cœur pour les sacrifices et les efforts que vous faites.
Mes frères et ma sœur je vous suis très reconnaissant pour tous les conseils que vous me donnez.

J'espère toujours vous rendre heureux et fiers de moi.

Je dédie aussi ce travail à tous mes amis, en souvenir des beaux moments qu'on a passé ensemble. Je vous remercie pour vos encouragements et de votre aide précieuse.

Nader Somrani

Dédicace

A mes parents pour tous les sacrifices innombrables qu'ils ont faits et pour tout le soutien qu'ils ont offert tout au long de mes études.

A ma sœur pour l'encouragements et le soutien qu'elle m'a apporté.

A mes amis pour leur encouragement et pour tous les bons moments qu'on a vécus ensemble, je vous souhaite un avenir radieux et plein de bonnes promesses.

Wajdi Khattel

Remerciement

Au terme de ce projet nous tenons à remercier infiniment tous les enseignants et administrateurs de l'institut supérieur d'informatique.

Nos vifs remerciements à Monsieur **Sami Ouali,** maître assistant à l'institut supérieur d'informatique, qui a gardé un œil attentif sur le déroulement du projet en donnant des remarques constructives. Nous le remercions pour sa disponibilité et son aide précieuse et cela a été un plaisir de travailler sous sa directive.

Nous exprimons aussi notre gratitude à Monsieur **Amir Somrani,** ingénieur informatique à **BFI** pour son aide, ses conseils et l'accueil chaleureux au sein de **BFI**.

Nous exprimons également notre reconnaissance à tous les membres du jury pour avoir accepté de juger ce travail.

Table des matières

La liste des figures

La liste des tableaux

La liste des abréviations

- **BFI** (**B**anque **F**inance **I**nternational) : c'est notre entreprise d'accueil.
- **SOAP** (**S**imple **O**bject **A**ccess **P**rotocol) : c'est un type de service web.
- **WSDL** (**W**eb **S**ervices **D**escription **L**anguage) : c'est un document XML qui décrit le service web.
- **UDDI** : (**U**niversal **D**escription **D**iscovery and **I**ntegration) c'est un document XML qui contient un annuaire de services.
- **REST** : (**Re**presentational **S**tate **T**ransfer) c'est un type de service web.
- **URI** : (**U**niform **R**esource **I**dentifier) c'est une chaine de caractère identifiant une ressource sur un réseau.
- **HTTP** : (**H**yper**T**ext **T**ransfer **P**rotocol) c'est un protocole de communication client-serveur développé pour le World Wide Web.
- **JSON** : (**J**ava**S**cript **O**bject **N**otation) c'est un format de données textuelles.
- **GPS** (**G**lobal **P**ositioning **S**ystem) : c'est un système de géolocalisation.
- **RIB** (**R**elevé d'**I**dentité **B**ancaire) : c'est un identifiant de coordonnées bancaires.
- **IBAN** (**I**nternational **B**ank **A**ccount **N**umber) : c'est le numéro international de compte bancaire.
- **BIC** (**B**ank **I**dentifier **C**ode) : c'est le RIB mais en anglais.
- **CRM** (**C**ustomer **R**elationship **M**anagement) : c'est la gestion de relation clientèles.
- **J2EE** (**J**ava **2** **E**nterprise **E**dition) : c'est la version Java d'entreprise.
- **MVC** (**M**odel-**V**ue-**C**ontroller) : c'est une architecture de programmation web.
- **QR Code** (**Q**uick-**R**esponse **Code**) : c'est un code à barre en deux dimensions (ou code à matrice).
- **API** (**A**pplication **P**rogramming **I**nterface) : c'est une interface de programmation applicative.
- **JSP** (**J**ava **S**erver **P**ages) : ce sont des pages html avec un code Java dedans.
- **VPS** (**V**irtual **P**rivate **S**erver) : c'est une machine virtuelle à distance.
- **JAX-RS** (**J**ava **A**PI e**X**tension for **R**ESTful Web **S**ervices) : c'est une extension d'interface de programmation Java pour le service web REST.
- **JVM** (**J**ava **V**irtual **M**achine) : c'est une machine virtuelle Java.
- **JDBC** (**J**ava **D**ata**B**ase **C**onnector) : c'est un moyen de communication entre une application Java et la base de données.
- **Zxing** (**Z**ebra cross**ing**) : c'est un projet open-source de code à barre multi format.
- **SSH** (**S**ecure **Sh**ell) : c'est un protocole de communication sécurisé.

Introduction générale

Le marché des technologies a connu une énorme évolution ces dernières années et avec l'intégration de l'internet il y a eu des changements sur la façon et la rapidité d'accès aux informations. Plusieurs secteurs se sont adaptés à ce changement et encore plus ont vu le jour pour satisfaire ces nouveaux clients.

Les banques représentent l'un des domaines qui connaissent un nombre énorme de client et qui souffrent pour les satisfaire. Ils doivent alors proposer leurs services d'une façon plus rapide et sécurisé mais aussi à distance afin de pouvoir éviter la congestion dans les agences.

Avec l'apparition des appareils mobiles, comme les smartphones et les tablettes qui ont connu une révolution technologique, il est devenu facile pour une banque de gérer ce conflit par proposer une application mobile capable de procurer des services indispensables rapidement et à tout moment.

Notre projet consiste à développer une application bancaire mobile sous Android qui propose de nombreuses fonctionnalités à l'utilisateur ainsi que sa partie administrative qui consiste à gérer les services clientèle.

Ce rapport décrit les étapes de développement de notre projet. Il contient 5 Chapitres.

Le premier chapitre « Présentation Générale » présente le contexte du travail.

Le deuxième chapitre « Etude théorique et choix techniques » traite l'étude de quelque solution déjà présente dans le marché, voir ce qu'ils apportent et déterminer ce qui manque pour proposer la meilleure solution.

Le troisième chapitre intitulé « Analyse des besoins et spécification »

Ensuite le quatrième chapitre « Conception » détaille l'étude conceptuelle du projet.

Le dernier chapitre « Réalisation » présente les démarches techniques pour la réalisation de l'application ainsi que l'environnement et les outils utilisés.

Finalement on termine avec une conclusion générale

Chapitre 1 : Présentation générale

1. Introduction

Ce chapitre présente l'entreprise d'accueil ainsi que la présentation de l'application et le cadre du projet.

2. Organisme d'accueil

BFI est une entreprise tunisienne crée en 1994, spécialisé dans l'édition et l'intégration de solutions logicielles destinées aux banques et institutions financières.

Cette entreprise se repose essentiellement sur le marché africain de la gestion bancaire dont elle occupe une place importante avec une réputation de très haut niveau en possédant enivrent 180 clients répartis dans des différentes nations. ***[1]***

3. Cadre du travail

Ce stage a été effectué dans le cadre d'un projet de fin d'études pour l'obtention du diplôme « *Licence fondamentale en science d'informatique* » de l'institut supérieur d'informatique. Le stage est réalisé au sein de l'entreprise « BFI » qui nous a proposés de concevoir une application mobile qui simule les fonctionnalités bancaires.

4. Présentation du projet

Une banque offre de nombreux services à ses clients qui doivent se présenter chaque fois à l'agence pour effectuer les moindres opérations.

Etant donné le nombre de clientèle bancaire, nous devrons trouver une solution pour le confort du client afin qu'il puisse faire ses besoins à distance et ainsi les agences ne souffrent pas d'encombrement humain.

C'est pourquoi il est nécessaire aujourd'hui pour une banque d'offrir à ses clients une application mobile permettant de réaliser des opérations en toute sécurité et à tout moment, de même elle doit proposer une partie administrative à ses employés.

4.1. Objectif du projet

Comme nous présente la figure 1.1, l'idée de ce projet est de réaliser une application mobile qui satisfait les besoins du client et lui fait gagner du temps pour ne pas se présenter aux agences.

Par conséquent, l'objectif consiste à développer une application mobile conviviale et moderne qui propose plusieurs fonctionnalités bancaires ainsi qu'une simple application web pour la partie administrative de la gestion des services proposés par la partie mobile.

Figure 1.1 : Idée du projet

4.2. Méthode de développement de l'application

Pour le développement de notre application nous avons choisi le schéma de développement « SCRUM ». Ce cadre méthodologique permet de garantir la transparence entre les membres de l'équipe, il permet aussi de faire des inspections pour détecter tout problème, et enfin pendant les inspections la méthode « Scrum » permet de s'adapter facilement et de faire les changements nécessaires.

Comme illustre la figure 1.2 la méthode « SCRUM » divise le développement sur des taches qui durent un intervalle de temps variable selon la grandeur du travail, appelé Sprint. Cette méthode répartit aussi les personnes selon des rôles :

- **Product Owner (**Le propriétaire du produit**) :** généralement c'est une personne experte en métier qui gère les fonctionnalités développées.
- **Scrum Master (**Le directeur du Scrum**) :** c'est une personne qui supervise le déroulement des sprints et qui facilite la communication de l'équipe.
- **Scrum Team (**L'équipe de Scrum**) :** c'est un ensemble de développeurs, testeurs et d'autres qui seront occupés de l'implémentation des taches. ***[2]***

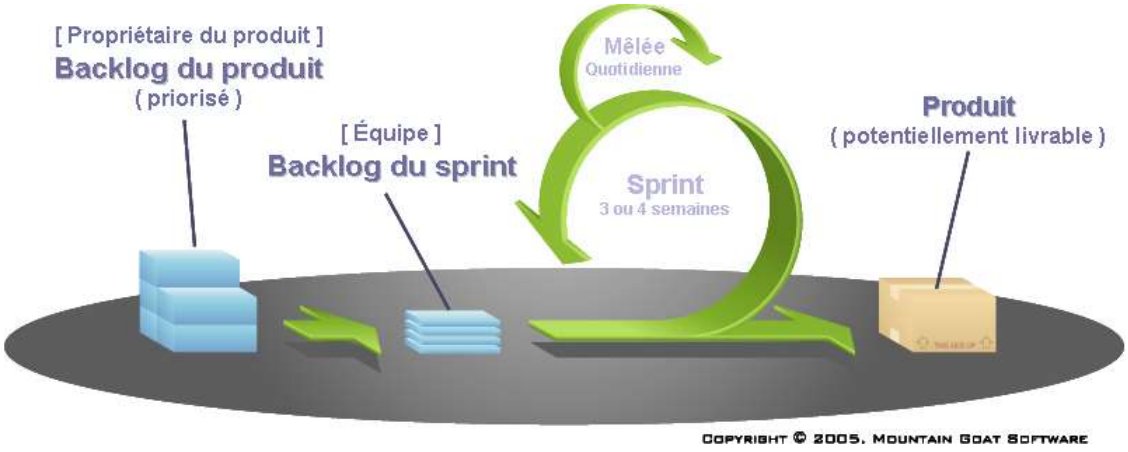

Figure 1.2 : La méthode "Scrum"

5. Conclusion

Dans ce chapitre nous avons introduit le contexte général du projet à réaliser et sa méthode de développement ainsi que la présentation de l'entreprise qui nous a accueillis.

Par suite nous allons aborder la phase d'étude du projet pour faire les bons choix techniques

Chapitre 2 : Etude théorique et choix techniques

1. Introduction

Ce chapitre présente des études théoriques sur les systèmes mobiles, et par la suite les différentes solutions disponibles du marché afin de les critiquer.

Tout ça pour enfin effectuer un bon choix technique pour réaliser le projet.

2. Etude préliminaire

Cette partie concerne les études des systèmes d'exploitation mobiles ainsi que les différents types de développement de ces derniers.

2.1. Les systèmes d'exploitation mobiles

Ces dernières années le marché mobile a connu une croissance figurante. Beaucoup de systèmes d'exploitation mobiles ont vu le jour, mais trois systèmes en particulier se sont imposés.

2.1.1. Android

Le système d'exploitation mobile le plus populaire s'est emparé aujourd'hui de la plus grosse part du marché. Cette croissance était très rapide. Du second trimestre 2009 au second trimestre 2010, la part de marché d'Android est passée de 1,8 % à 17,2 % avec un taux de croissance de 850 %. Le 15 novembre 2011, Android atteint les 52,5% de part du marché mondial des smartphones. En septembre 2014, la part de marché mondiale d'Android est passée à 85 %.

Android a été développé par une petite startup achetée ensuite par Google. ***[3]***

2.1.2. iOS

Beaucoup diront qu'Apple a révolutionné la notion d'appareils mobile avec l'iPhone en 2007. Et leur système d'exploitation iOS était une innovation en lui-même. iOS est dérivé du système d'exploitation Mac OS X et tous les appareils mobiles d'Apple l'utilisent, l'iPhone, l'iPad, l'iPod ou bien l'Apple Watch.

En septembre 2014, la part de marché d'iOS était de 11%. ***[3]***

2.1.3. Windows Phone

Le système d'exploitation de Microsoft n'a jamais été aussi populaire que prévu et ses parts de marché ont décliné ces dernières années.

En septembre 2014, la part de marché de Windows Phone était de 2,5 % ***[3]***

2.2. Les types d'applications mobiles

Afin de développer une application mobile il existe 3 différentes solutions comme le montre la figure 2.1.

- Développer une application native
- Développer une application web
- Développer une application hybride

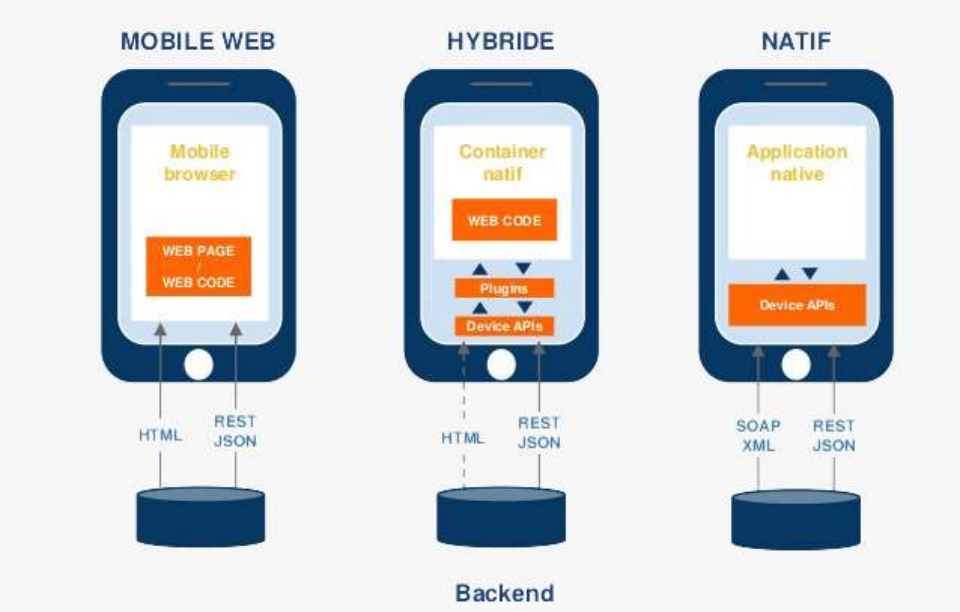

Figure 2.1 : Les types d'application mobile

2.2.1. Présentation

Application Native

Il s'agit des applications conçues pour une plateforme spécifique en se référant à un langage particulier propre au système d'exploitation choisi (Java pour Android, Objective-C ou SWIFT pour iOS et C# pour Windows Phone). Ces applications sont distribuées uniquement par l'intermédiaire de son magasin (Play Store, App Store, etc.).

Le développement natif recourt essentiellement à la mémoire du smartphone ainsi que la facilité d'intégrer toutes les fonctionnalités liées au système d'exploitation visé tel que le GPS et la caméra, cela nous amène à des applications mobiles performantes, bien développées et riches en matière de fonctionnalités.

Le souci du développement natif c'est qu'on doit développer la même application pour chaque plateforme ce qui implique que le temps de développement va être multiplié par le nombre de plateforme visé. ***[4] [5]***

- **Application Web**

Les applications web sont réellement des sites web développé en HTML et conçus pour être utilisé sur les tailles des écrans mobiles. Ils sont accessibles via le navigateur web du smartphone et ne nécessitent pas forcément de télécharger l'application.

Le point fort c'est qu'on n'a qu'à développer un seul format qui marche sur toutes les plateformes donc un gain de temps de développement énorme et une facilité de mise à jour.

Cependant l'application web ne sert pas de la mémoire du smartphone donc si l'application nécessite d'intégrer plusieurs fonctionnalités elle sera longue à charger. ***[4]***

- **Application Hybride**

Tout comme les applications web, ce type est développé à partir de langages web qui vont par suite s'intégrer à des « WebView » dans la partie native. Donc c'est des applications multiplateformes qui peuvent en plus utiliser les fonctionnalités natives.

Néanmoins, les applications hybrides peuvent parfois ne pas bien s'adapter au système d'exploitation du smartphone pour des raisons de résolution et d'autres. ***[4]***

2.2.2. Développement d'une application native

- **Application Android**

Pour développer une application Android il est nécessaire de maitriser le langage JAVA et il faut aussi avoir une idée sur le langage XML.

La composante principale d'une application Android est l'activité « Activity ». Une activité représente l'interface que nous voyions sur l'écran et le code JAVA qui alimente cette interface.

Une autre composante qui permet de créer des interfaces est le « Fragment ». Un fragment est similaire à une activité sauf qu'il est plus léger. Il est conseillé de toujours utiliser si c'est possible un fragment.

Un fragment doit toujours être contenu dans une activité ce qui fait que dans une application Android il faut y avoir au moins une activité.

Les interfaces graphiques sont constituées d'un ensemble de View codé en XML.

La figure 2.2 nous montre l'interface de l'environnement de développement « Android Studio » qui est le plus conseillé par Google pour développer une application Android native.

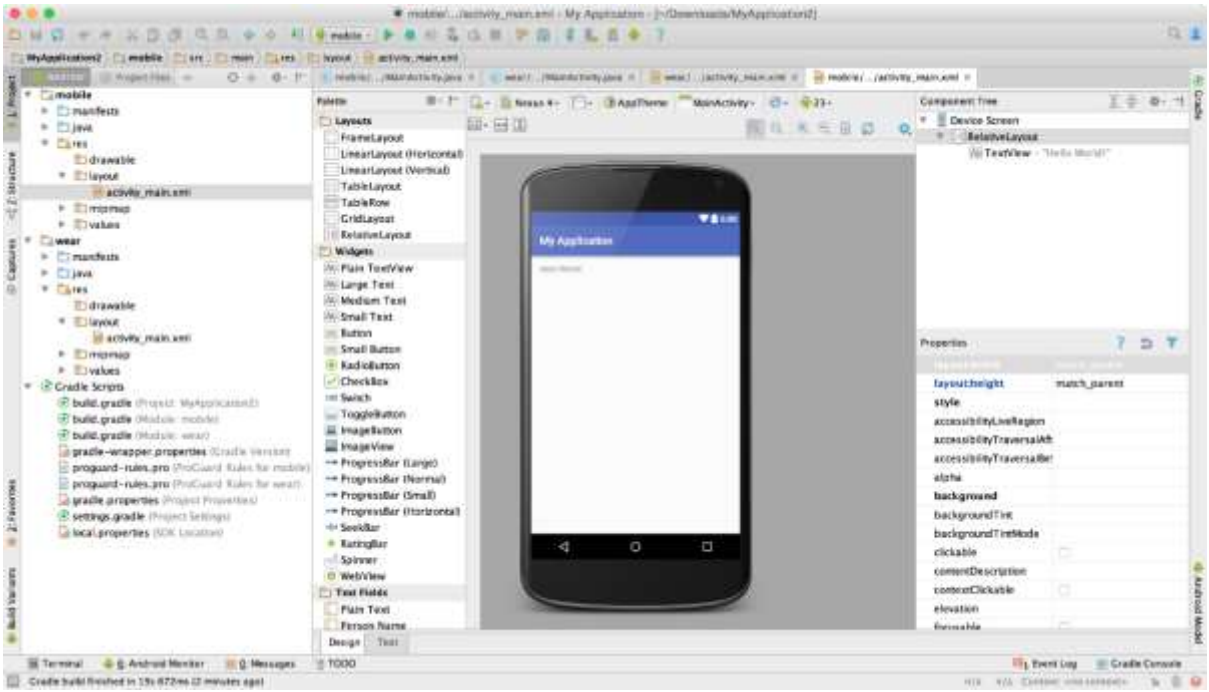

Figure 2.2 : Interface Android studio

- **Application iOS**

Les applications iOS sont développées en langage « objective-c » ou bien le langage « SWIFT ».

Comme illustre la figure 2.3, il existe 3 composantes essentielles pour une application iOS.

- Model : c'est l'objet qui contient les données d'une application, qui procure et gère l'accès à ces données.

- Controller : cet objet agit comme un coordinateur entre des différents objets de type « View objects » ou bien entre des différents objets de type Model.
- View : c'est l'interface qui s'affiche à l'écran et qui réagit aux actions de l'utilisateur.

« Xcode » est l'environnement de développement d'application iOS, il est disponible seulement sur le système d'exploitation Mac OS X et il est nécessaire de créer un compte « Apple ID » pour commencer à développer.

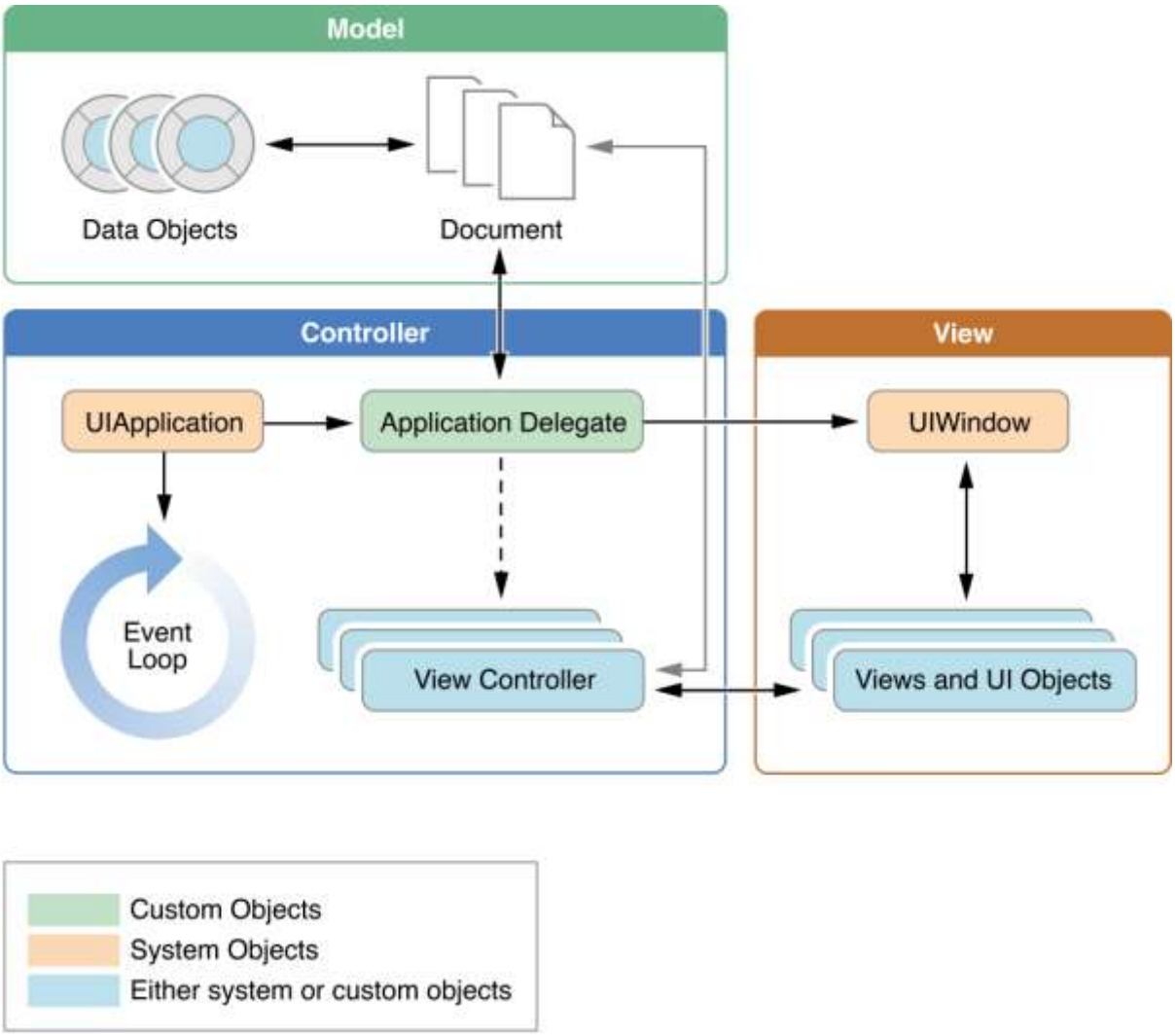

Figure 2.3 : Architecture de développement iOS

❖ **Application Windows Phone**

Le développement d'application « Windows Phone » nécessite la maitrise du langage C# mais on peut aussi développer des applications pour « Windows Phone » 7.5 en langage F# ou VB.Net.

L'environnement de développement d'application « Windows Phone » est « Visual Studio », la figure 2.4 montre son interface.

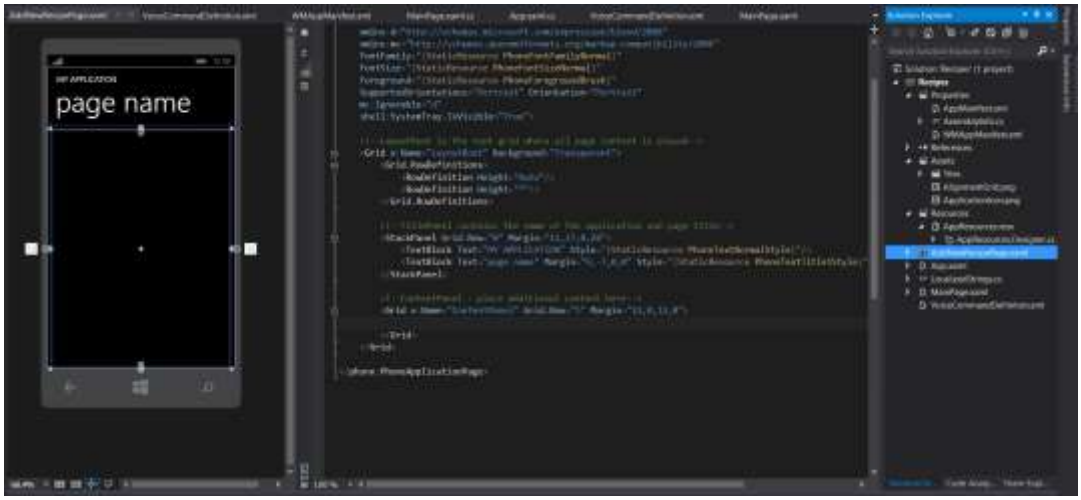

Figure 2.4 : Interface Visual Studio

2.2.3. Développement d'une application hybride

Le développement d'application hybride se base essentiellement sur des composantes HTML5, CSS et JAVASCRIPT. L'application est un site web sauf que ce site web tourne sur une application native. Généralement ces applications utilisent des plateformes comme Cordova qui acte comme un conteneur qui fait tourner l'application et qui leur donne accès aux ressources natives de l'appareil comme l'appareil photo, l'accéléromètre ou encore le GPS

- **Ionic v2**

Ionic est un Framework HTML5 récent dédié au développement d'application mobile hybride fonctionnant sur Windows Phone, iOS ou bien Android.

Le Framework offre de nombreuses composantes qui permettent de créer des applications avec des interfaces modern et le développement se fait avec le langage Angular js.

- **Mobile Angular UI**

Mobile Angular UI est un autre Framework HTML5 dédié au développement d'application hybride. Mobile Angular UI se base sur le langage Angular js ainsi que sur les composantes Bootstrap de Twitter.

❖ **Cordova**

Cordova est une plateforme pour le développement d'application mobile native avec du HTML5, CSS et JAVASCRIPT. L'interface d'une application Cordova est réellement une WebView qui occupe tout l'écran. Cela veut dire que seul le conteneur c'est-à-dire le WebView change d'un système d'exploitation à un autre.

2.3. Les services web

Les services web sont des protocoles qui permettent la communication et l'échange de données entre des systèmes informatiques distribués. Le grand avantage des services web c'est que la communication peut se faire entre des systèmes informatiques avec des architectures différentes. Les systèmes qui exploitent les services peuvent être dans des environnements de développement totalement différents.

❖ **Type de service web :**

Il existe deux principaux types de service web :

2.3.1. Les service web WS

Il s'agit des services web basés sur les standards « SOAP » et « WSDL ». Ces services exposent les fonctionnalités sous la forme de services exécutables à distance. Les services sont ensuite répertoriés dans des annuaires comme les « UDDI » (Universal Description, Discovery, and Integration).

L'échange des messages se fait avec une structure standardisée de format XML.

2.3.2. Les services web de type REST

Les services web de type « REST » (representational state transfer) appelé aussi « RESTful Web Services » sont des services web basés sur les technologies du web. C'est-à-dire qu'ils exposent leurs fonctionnalités à travers des « URI » accessibles par le protocole « HTTP ». Les messages de réponse peuvent avoir des représentations différentes (Texte, XML, JSON, …).

3. Etude de l'existant

Dans cette section nous allons étudier les différentes applications bancaires du marché pour bien prendre nos décisions techniques.

3.1. Description des solutions disponibles

Toutes les descriptions sont prises à partir de l'adresse électronique ***[6]***.

3.1.1. Solutions disponibles en Tunisie

❖ **Attijari Mobile Tunisie :**

Attijari Bank présente une application mobile facile à utiliser avec une interface accueillante et qui offre des services comme :

- Consulter le solde et les mouvements des comptes en temps réel, avec un historique de 6 mois.
- Effectuer des recherches multicritères sur les écritures du compte
- Commander un chéquier
- Consulter les cours des devises

La Figures 2.5 montre l'interface principale de l'application.

Figure 2.5 : Solution "Attijari Bank"

❖ **STB :**

STB Bank offre elle aussi une application mobile avec une interface originale. L'application offre des services comme :

- Consulter les soldes et les mouvements de ses comptes
- Parcourir l'historique de ses opérations de dépôt ou de retrait
- Visualiser l'image chèque.
- Gérer ses cartes bancaires
- Consulter son portefeuille Crédit, Titre et Sicav

- Consulter ses Placements
- Consulter ses impayés (effets et chèques)
- Consulter ses opérations monétiques
- Planifier et ordonner des ordres de virement unique
- Réaliser un transfert d'argent pour ses proches.
- Planifier un rendez-vous avec le chef d'agence ou le chargé clientèle (avec visioconférence)
- Lancer une demande de chéquier ou de carte
- Déposer une opposition de chèque ou carte bancaire.
- Suivre les cours de devise et l'historique des TMM

La figure 2.6 représente l'interface de l'application.

Figure 2.6 : Solution "STB Bank"

❖ **Banque de Tunisie :**

Banque de Tunisie est une autre banque qui offre une application mobile riche en fonctionnalités. Et comme, ces fonctionnalités sont divisées entre des services offerts au public et d'autres offerts aux abonnées, on trouve :

Les services offerts dans l'espace public :

- Géolocalisation des agences et des distributeurs de billets
- Consultation des cours des devises avec gestion des devises favoris
- Consultation des cours en bourse
- Simulateur de crédits

Les services offerts dans l'espace des abonnés :

- Consultation des comptes avec affichage de la liste des comptes du client
- Affichage des 15 derniers mouvements d'un compte avec l'option d'avoir un recul de 30 jours
- Affichage du détail des transactions
- Consultation de la liste des chéquiers et visualisation recto verso des chèques encaissés
- Le lancement d'une demande de chéquier
- La consultation du RIB et de L'IBAN
- La consultation des comptes à terme
- La consultation des encours de crédits, échéances ; capital restant dû
- Mise en opposition d'une carte
- Gestion des cartes : visualisation des 10 dernières transactions et activation de la carte à l'étranger
- Consultation du portefeuille titres
- Consultation des alertes configurées par le client
- Emission de virements internes et externes
- Paiement des factures
- Lancement d'une réclamation via le CRM

L'interface d'accueil de l'application est représentée dans la figure 2.7.

Figure 2.7 : Solution "banque de Tunisie"

3.1.2. Solutions disponibles dans le monde

❖ **Barclays Mobile Banking :**

La banque britannique Barclays offre une application mobile très conviviale avec une interface moderne. Ci-dessous la figure 2.8 montre l'interface de l'application.

Figure 2.8 : Solution "Barclays"

❖ **Chase Mobile :**

Chase est une banque américaine qui offre la meilleure application mobile bancaire selon le magazine « Forbes ». L'application offre un très grand nombre de fonctionnalités avec une interface moderne et conviviale.

La figure 2.9 expose les différentes interfaces de l'application.

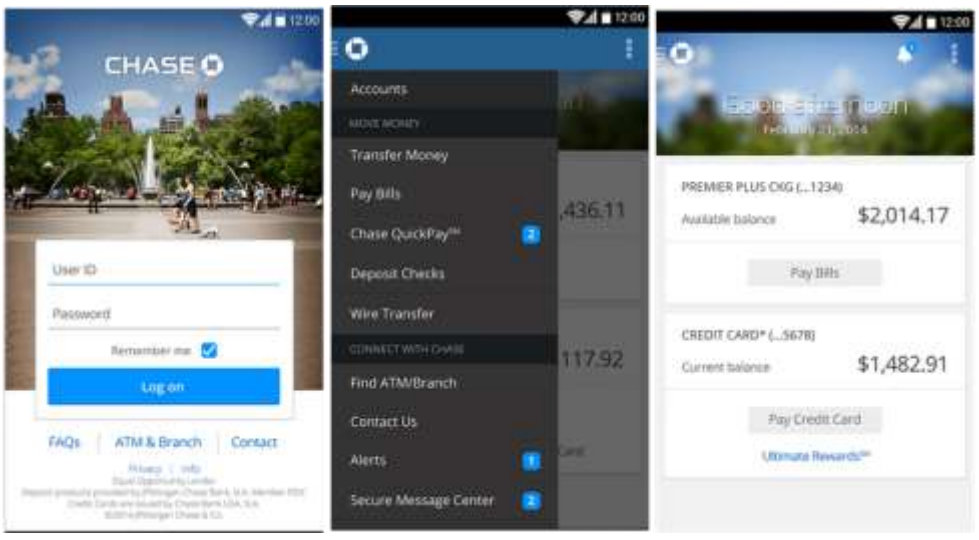

Figure 2.9 : Solution "Chase mobile"

❖ **Mes Comptes BNP Paribas :**

La banque française BNP Paribas présente une application mobile très performante avec une interface unique. L'application affiche un très grand nombre d'informations sans pour autant encombrer l'écran.

La figure 2.10 représente l'interface principale de l'application.

Figure 2.10 : Solution "BNP Paribas"

3.2. Critique de l'existant

Il existe un bon nombre d'applications mobiles bancaires en Tunisie. Mais ces applications ne réussissent pas à proposer à la fois un grand nombre de fonctionnalités et une interface moderne et conviviale.

L'application « Attijari mobile Tunisie » propose une interface conviviale mais pas très moderne et elle manque aussi de quelques fonctionnalités comme la géolocalisation des agences bancaires.

L'application « STB » offre un grand nombre de fonctionnalités mais l'interface n'est pas très bien organisée.

L'application « Banque de Tunisie » offre elle aussi un très grand nombre de fonctionnalités mais propose une interface non moderne.

4. Problématique

Étant donné cette diversité de système d'exploitation et de type d'application mobile, nous avons dû choisir entre développer une application spécifique à une plateforme ou bien une application multiplateforme.

5. Solution envisagée

Pour une banque l'application proposée doit être ergonomique, rapide et riche de fonctionnalités. Suite à notre étude on a constaté que les applications natives offrent les meilleures performances.

Ensuite il fallait choisir le système d'exploitation, Android est le système d'exploitation le plus populaire et c'est une plateforme open source, alors que pour développer une application iOS il faut une machine qui tourne sous Mac OS X. C'est pour ces raisons que nous avons choisi de développer une application Android native.

Pour terminer, nous avons constaté que les services web SOAP sont plus lents par rapport aux services web basé sur REST qui offrent une simplicité de mise en œuvre, du coup nous avons choisi de travailler avec REST.

6. Architecture des applications

Dans cette partie nous allons présenter l'architecture de l'application mobile et de l'application web.

6.1. Architecture de la partie mobile

Pour la partie mobile nous avons opté pour une architecture client/serveur 3-tièrs. L'application est devisée en trois couches comme présenté sur la figure 2.11 :

- **Couche présentation** : c'est l'application Android et les différentes interfaces.
- **Couche métier** : c'est la partie traitement, elle est repartie entre le serveur d'application et l'application mobile.
- **Couche accès aux données** : c'est la partie gérant l'accès aux données du système.

La partie client qui représente l'application Android demande un service de la partie serveur, le serveur d'application va alors demander des données du serveur de base de données pour ensuite retourner le résultat souhaité au client.

Figure 2.11 : Architecture 3-tières

6.2. Architecture de la partie administrative

Pour la partie administrative nous avons adopté une architecture J2EE standard d'un client léger qui s'applique au modèle MVC dont on distingue 3 couches comme nous le voyons sur la figure 2.12 :

- **Couche Modèle :** c'est la partie qui concerne la gestion des données.
- **Couche Vue :** c'est la partie de présentation des données.
- **Couche Contrôleur :** c'est un switcher qui précise la vue a renvoyer suite à la requête de l'utilisateur.

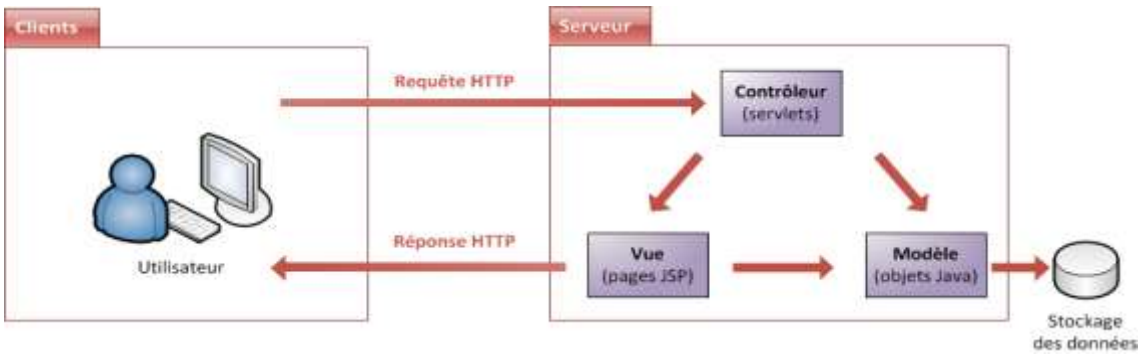

Figure 2.12 : Architecture j2ee client léger

7. Conclusion

Dans ce chapitre nous avons réalisé une étude sur tous les aspects de développement d'application mobile, notamment nous avons choisi la plateforme de développement, puis nous avons réalisé une étude sur l'existant pour proposer la meilleure application sur le marché.

Après cela, nous allons faire une analyse sur les besoins que nous devons offrir aux acteurs de notre projet

Chapitre 3 : Analyse des besoins et spécification

1. Introduction

Ce chapitre présente les acteurs de notre projet ainsi que l'analyse des différents besoins de chacun.

2. Analyse des besoins

Notre travail sert à offrir à un client de la banque accès en temps réel aux données et services bancaires depuis son téléphone, toutefois l'administrateur doit gérer les services offerts par l'application à partir de son navigateur.

2.1. Identification des acteurs

- **Le client :** c'est l'acteur principal de notre application il profite de toutes les fonctions offertes par l'application.
- **L'administrateur :** le rôle de cet acteur est de gérer les comptes et les clients ainsi que les demandes des cartes de crédit ou des chéquiers et de répondre aux messages envoyés par les utilisateurs.

2.2. Les besoins fonctionnels

Nous allons définir ici les actions qu'un acteur peut faire à l'aide de l'application mise à sa disposition.

2.2.1. Les besoins fonctionnels liés au client

Toute l'application mobile est disponible au client pour lui permettre après l'authentification par son email et son mot de passe d'avoir accès aux différents services offerts :

- Consulter ses informations personnelles ainsi que tous ses comptes bancaires.
- Consulter toutes les transactions effectuées sur ses comptes.
- Transférer de l'argent.
- Payer des factures.

- Envoyer des demandes de carte de crédit ou de carnet de chèque.

Toutefois, un simple utilisateur sans authentification peut aussi bénéficier de certains services :

- Contacter la banque
- Voir les distributeurs de billet et les agences les plus proches.
- Convertir de la devise et visualiser le cours de change.
- Simuler des crédits.

2.2.2. Les besoins fonctionnels liés à l'administrateur

Une application web est mise à la disposition de l'administrateur pour la gestion administrative de l'application mobile, pareil que le client, après l'authentification il a accès à :

- La gestion des clients ainsi que les comptes bancaires
- La gestion des demandes effectuées par les clients
- La gestion des communications avec la clientèle

2.3. Les besoins non fonctionnels

Nous allons déterminer l'ensemble de contraintes d'implémentation à respecter pour garantir le bon fonctionnement de l'application

- **La sécurité :** C'est un besoin très important à assurer lorsqu'on parle de banques, pour cela chaque opération qui semble avoir un risque potentiel il doit y avoir une autre validation d'identité par un code secret.
- **La performance :** Etant donné le nombre d'utilisateur simultanés, l'application doit être en mesure de satisfaire tous les clients. On distingue pour ça 3 axes de travail :
    - **Rapidité :** l'application doit être conçue pour avoir un temps de réponse minimum.

- **Disponibilité :** le réseau doit être disponible en permanence pour permettre à chaque utilisateur d'accéder aux services disponibles à n'importe quel moment.
- **Fiabilité :** l'application doit avoir une bonne qualité de contenu ainsi qu'une bonne adaptabilité aux différentes tailles des écrans des appareils mobiles. D'autre part il faut assurer le bon fonctionnement sans erreur.

- **L'ergonomie :** L'application doit être simple à maintenir et à comprendre.

3. Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation représente les actions réalisées par le système pour avoir un résultat qui répond au besoin d'un acteur particulier.

Nous allons présenter ici les diagrammes de cas d'utilisation de chaque partie.

3.1. Diagramme de cas d'utilisation du client

La figure 3.1 montre les différentes fonctionnalités qu'un client peut faire.

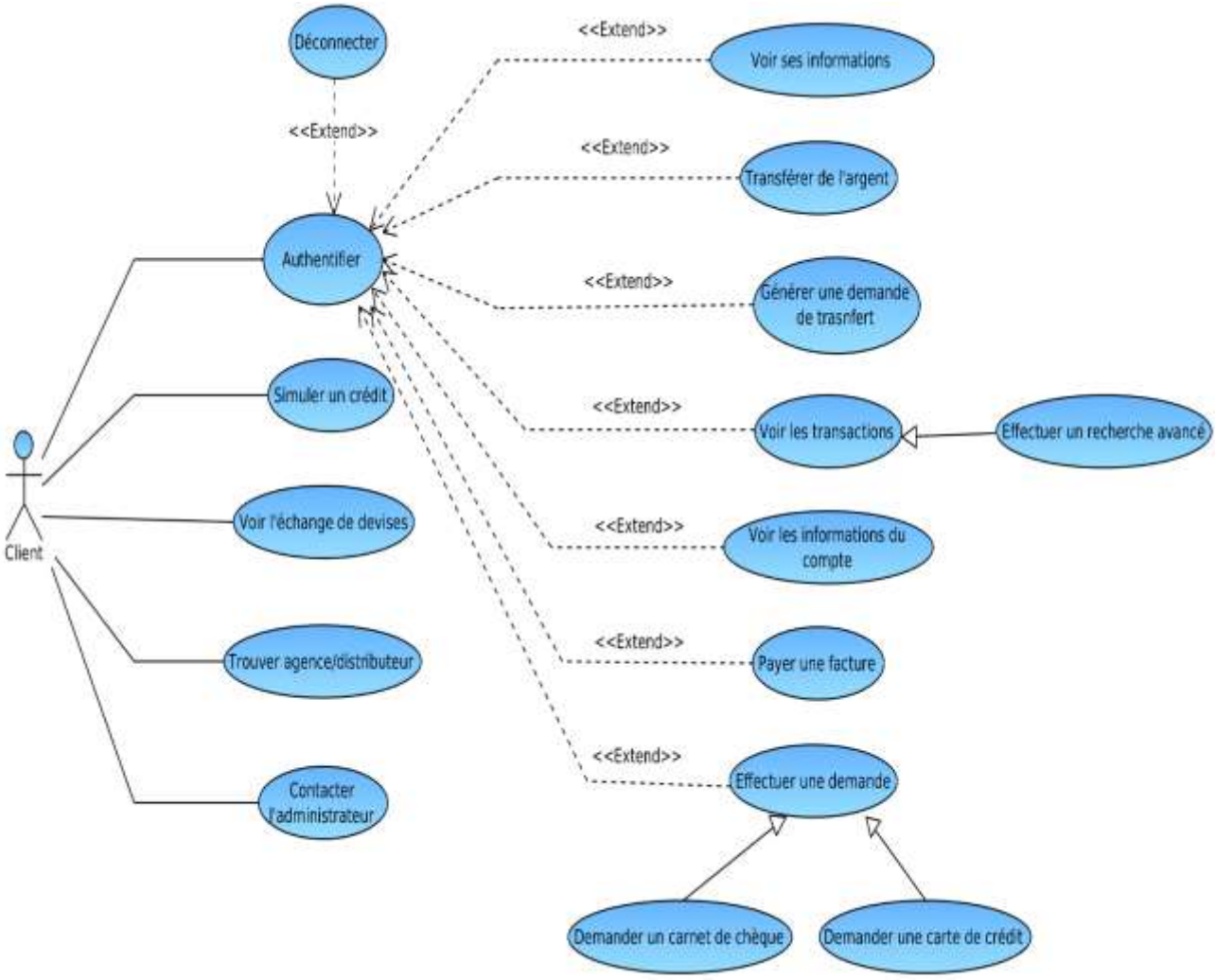

Figure 3.1 : Diagramme de cas d'utilisation du client

3.2. Diagramme de cas d'utilisation de l'administrateur

La figure 3.2 montre les différentes fonctionnalités qu'un administrateur peut faire.

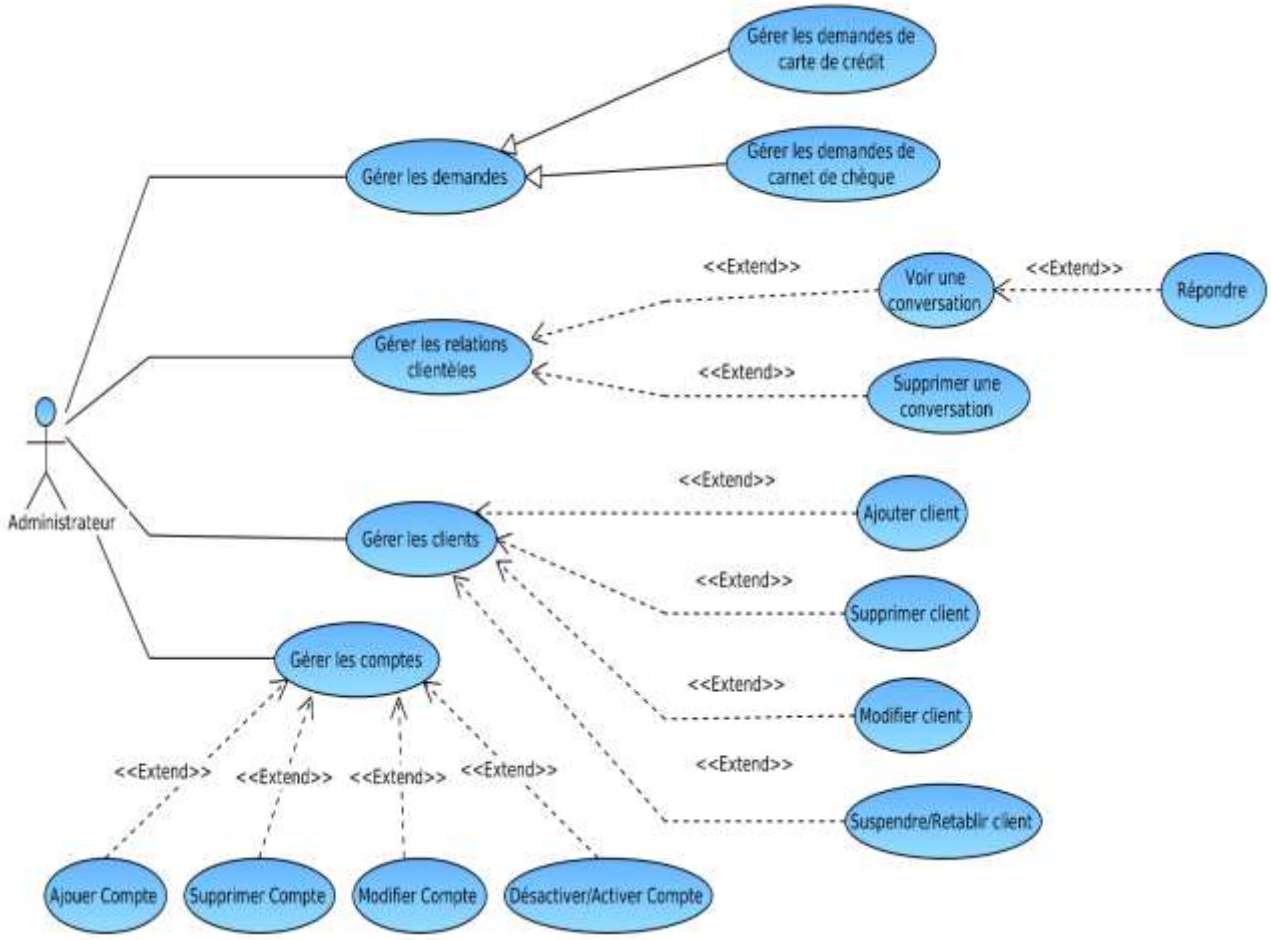

Figure 3.2 : Diagramme de cas d'utilisation de l'administrateur

4. Spécification fonctionnelle

Dans cette section nous allons citer les descriptions textuelles qui décrivent les scénarios des différents cas d'utilisation.

4.1. Les cas d'utilisation du client

4.1.1. Description du cas d'utilisation « Authentifier »

Tableau 3.1 : Description textuelle du cas d'utilisation "s'authentifier"

Sommaire d'identification
Titre : S'authentifier
Résumé : Permettre au client de s'authentifier pour bénéficier des services de l'application.
Acteur : Client
Description des enchainements
Pré condition : • Service web disponible • Serveur de base de données disponible
Scénario nominal : 1) Le client saisit son email et son mot de passe 2) Le client clique sur le bouton connexion 3) Le système vérifie les données de connexion 4) Le client accède à sa page d'accueil
Scénario alternative : Alt-1) champs vide : le système affiche le message d'erreur « veuillez remplir vos données » Alt-2) Email invalide : le système affiche le message d'erreur « vérifier que l'email soit valide » Alt-3) Mot de passe invalide : le système affiche le message d'erreur « Mot de passe trop coute » Alt-4) Email ou mot de passe incorrecte : le système affiche le message d'erreur « Email ou mot de passe incorrecte » Alt-5) Service web ou serveur de base de données indisponible : le système affiche le message d'erreur « Une erreur inattendue s'est produite »
Post condition : Le client est authentifié et accède à sa page d'accueil

4.1.2. Description du cas d'utilisation « Effectuer une demande »

Tableau 3.2 : Description textuelle du cas d'utilisation "Effectuer une demande"

Sommaire d'identification
Titre : Effectuer une demande
Résumé : Permettre au client de demander une carte de crédit ou un carnet de chèque
Acteur : Client
Description des enchainements
Pré condition : • Client authentifié • Service web disponible • Serveur de base de données disponible
Scénario nominal : 1) Le client choisi le type de sa demande 2) Le client confirme sa demande 3) Le système enregistre la demande dans la base de données 4) Le système envoie les coordonnées de la demande par email au client.
Scénario alternative : Alt-1) Le client possède déjà une demande équivalente Alt-2) Service web ou serveur de base de données indisponible : le système affiche le message d'erreur « Une erreur inattendue s'est produite »
Post condition : Le client est informé par email et attend qu'un administrateur approuve sa demande.

4.1.3. Description du cas d'utilisation « Voir les transactions »

Tableau 3.3 : Description textuelle du cas d'utilisation "Voir les transactions"

Sommaire d'identification
Titre : Voir les transactions
Résumé : Permettre au client de voir les transactions de l'un de ses comptes
Acteur : Client
Description des enchainements
Pré condition : • Client authentifié • Client adhéré à un compte • Service web disponible • Serveur de base de données disponible
Scénario nominal : 1) Le client choisit un compte 2) Le client accède à la page des transactions 3) Le système récupère les transactions du compte et les affiche
Scénario alternative : Alt-1) Pas de transaction pour ce compte : le système affiche un message d'erreur « aucune donnée n'est trouvée » Alt-2) Service web ou serveur de base de données indisponible : le système affiche le message d'erreur « Une erreur inattendue s'est produite »
Post condition : Le client voit les détails de chaque transaction sélectionnée

4.1.4. Description du cas d'utilisation « Transférer de l'argent »

Tableau 3.4 : Description textuelle du cas d'utilisation "Transférer de l'argent"

Sommaire d'identification
Titre : Transférer de l'argent
Résumé : Permettre au client d'effectuer une transaction
Acteur : Client
Description des enchainements
Pré condition : • Client authentifié • Client adhéré à un compte • Service web disponible • Serveur de base de données disponible
Scénario nominal : 1) Le client remplit les champs de transfert ou analyse un code QR 2) Le client entre son code secret 3) Le client confirme l'envoi 4) Le système vérifie les données 5) Le système effectue la transaction
Scénario alternative : Alt-1) L'un des champs obligatoires est vide : le système affiche un message d'erreur « veuillez remplir tous les champs obligatoires » Alt-2) Montant invalide : le système affiche un message d'erreur « le montant doit être plus que $0.01 » Alt-3) Solde insuffisant : le système affiche un message d'erreur « Solde insuffisant » Alt-4) Service web ou serveur de base de données indisponible : le système affiche le message d'erreur « Une erreur inattendue s'est produite »
Post condition : Transaction effectuée

4.1.5. Description du cas d'utilisation « Contacter l'administrateur »

Tableau 3.5 : Description textuelle du cas d'utilisation "Contacter l'administrateur"

Sommaire d'identification
Titre : Contacter l'administrateur
Résumé : Permettre à l'utilisateur de l'application de contacter l'administrateur
Acteur : Client
Description des enchainements
Pré condition : • Service web disponible • Serveur de base de données disponible
Scénario nominal : 1) Le client remplit le formulaire 2) Le client envoie son message 3) Le système enregistre le message dans la base de données 4) Le système envoie les coordonnées du message par email au client
Scénario alternative : Alt-1) Service web ou serveur de base de données indisponible : le système affiche le message d'erreur « Une erreur inattendue s'est produite »
Post condition : Le client est informé par email et attend qu'un administrateur lui réponde

4.1.6. Description du cas d'utilisation « Simuler un crédit »

Tableau 3.6 : Description textuelle du cas d'utilisation "Simuler un crédit"

Sommaire d'identification
Titre : Simuler un crédit
Résumé : Permettre au client de simuler une demande de crédit en impliquant les taxes et les impôts
Acteur : Client
Description des enchainements
Pré condition : application lancée
Scénario nominal : 1) Le client remplit les données de simulation de son choix 2) Le client clique sur le bouton de génération
Scénario alternative :
Post condition : Le crédit est généré

4.1.7. Description du cas d'utilisation « Voir l'échange des devises »

Tableau 3.7 : Description textuelle du cas d'utilisation "Voir l'échange des devises"

Sommaire d'identification
Titre : Voir l'échange des devises
Résumé : Permettre à l'utilisateur de visualiser l'échange entre les différentes devises du monde.
Acteur : Client
Description des enchainements
Pré condition : • Service des devises disponible
Scénario nominal : 1) Le client saisit le montant désiré 2) Le système convertit ce montant en plusieurs devises
Scénario alternative : Alt-1) Service des devises indisponible : le système affiche le message d'erreur « Une erreur inattendue s'est produite »
Post condition : Le montant saisi par le client est converti en plusieurs devises

4.1.8. Description du cas d'utilisation « Générer une demande de transfert »

Tableau 3.8 : Description textuelle du cas d'utilisation "Générer une demande de transfert"

Sommaire d'identification
Titre : Générer une demande de transfert
Résumé : Permettre au client de générer un code QR qui sert à remplir automatiquement les champs de transfert de celui qui analyse ce code
Acteur : Client
Description des enchainements
Pré condition : • Client authentifié • Client adhéré à un compte • Espace de stockage disponible sur l'appareil mobile
Scénario nominal : 1) Le client remplit le champ du montant à demander 2) Le client clique sur le bouton de génération 3) Le système enregistre le code QR sur l'appareil
Scénario alternative : Alt-1) Pas d'espace de stockage : le système affiche « Il n'y a plus d'espace de stockage dans votre mémoire »
Post condition : Le code QR est généré et peut être partagé

4.2. Les cas d'utilisation de l'administrateur

4.2.1. Description du cas d'utilisation « Gestion des clients »

Tableau 3.9 : Description textuelle du cas d'utilisation "Gestion des clients"

Sommaire d'identification
Titre : Gestion des clients
Résumé : Permettre à l'administrateur d'ajouter, modifier, supprimer ou suspendre/rétablir un client.
Acteur : Administrateur
Description des enchainements
Pré condition : administrateur authentifié
Scénario nominal : 1) L'administrateur voit la liste des clients 2) L'administrateur sélectionne le client désiré
Scénario alternative : Alt-1) Serveur de base de données indisponible : l'administrateur reçoit une liste vide
Post condition : L'administrateur gère le client sélectionné

4.2.2. Description du cas d'utilisation « Gestion des comptes »

Tableau 3.10 : Description textuelle du cas d'utilisation "Gestion des comptes"

Sommaire d'identification
Titre : Gestion des comptes
Résumé : Permettre à l'administrateur d'ajouter, modifier, supprimer ou désactiver/activé un compte
Acteur : Administrateur
Description des enchainements
Pré condition : administrateur authentifié
Scénario nominal : 1) L'administrateur voit la liste des comptes 2) L'administrateur sélectionne le compte désiré
Scénario alternative : Alt-1) Serveur de base de données indisponible : l'administrateur reçoit une liste vide
Post condition : L'administrateur gère le compte sélectionné

4.2.3. Description du cas d'utilisation « Gestion des demandes »

Tableau 3.11 : Description textuelle du cas d'utilisation "Gestion des demandes"

Sommaire d'identification
Titre : Gestion des demandes
Résumé : Permettre à l'administrateur de gérer les demandes de carte de crédit ou des carnets de chèque envoyé par les clients
Acteur : Administrateur
Description des enchainements
Pré condition : administrateur authentifié
Scénario nominal : 1) L'administrateur voit la liste des demandes 2) L'administrateur sélectionne la demande désirée
Scénario alternative : Alt-1) Serveur de base de données indisponible : l'administrateur reçoit une liste vide
Post condition : L'administrateur approuve ou désapprouve la demande sélectionnée

4.2.4. Description du cas d'utilisation « Gestion des relations clientèles »

Tableau 3.12 : Description textuelle du cas d'utilisation "Gestion des relations clientèles"

Sommaire d'identification
Titre : Gestion des relations clientèles
Résumé : permettre à l'administrateur de répondre aux différents besoins clientèle
Acteur : administrateur
Description des enchainements
Pré condition : administrateur authentifié
Scénario nominal : 1) L'administrateur voit la liste des sujets 2) L'administrateur sélectionne le sujet désiré 3) L'administrateur répond au sujet sélectionné
Scénario alternative : Alt-1) Serveur de base de données indisponible : l'administrateur reçoit une liste vide Alt-2) Service d'email indisponible : la réponse ne s'envoie pas
Post condition : Le client reçoit sa réponse par email

5. Conclusion

Dans ce chapitre nous avons énuméré les besoins fonctionnels et non fonctionnels de notre projet, puis nous avons élaboré les différents diagrammes de cas d'utilisation ainsi que leurs descriptions textuelles.

Le prochain chapitre va modéliser ces services par des diagrammes pour mieux comprendre le développement.

Chapitre 4 : Conception

1. Introduction

Ce chapitre présente la partie conception qui sert à réaliser des différents types de diagramme qui modélise les différentes parties du projet afin de mieux comprendre le système.

2. Diagramme de classes

Ce diagramme est classé comme un modèle statique du système, il sert à réaliser une modélisation des relations entre les classes du système orienté objet.

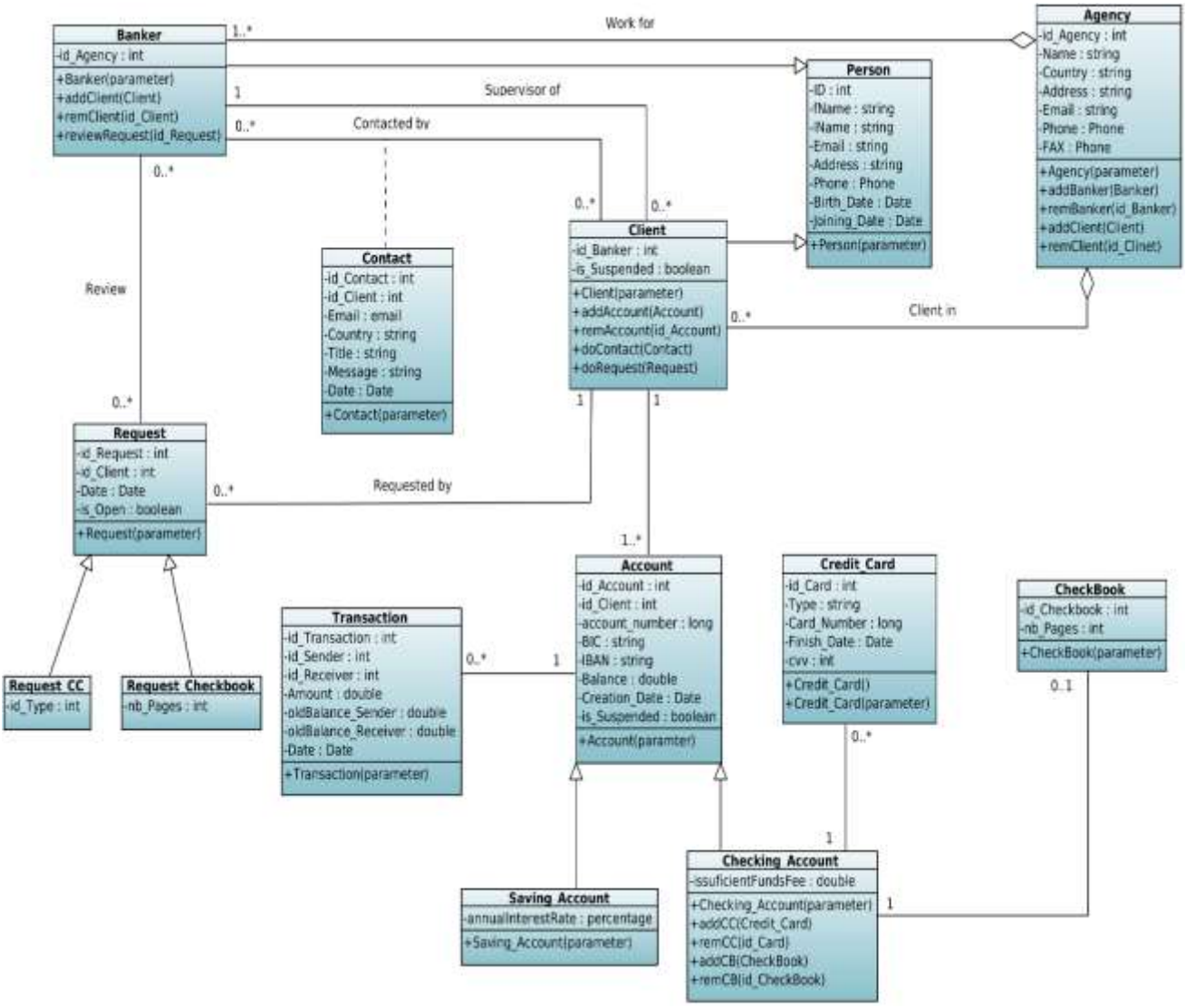

Figure 4.1 : Diagramme de classes

La figure 4.1 présente le diagramme de classes de notre projet :

- **La classe « Agency » :** c'est la classe qui représente les informations d'une agence.
- **La classe « Person » :** c'est la classe qui contient les informations des différents types de personnes.
- **La classe « Banker » :** c'est une classe qui hérite de la classe « Person » et qui comporte des attributs et des opérations supplémentaires que sa classe mère qui ont rapport avec le banquier.
- **La classe « Client » :** c'est une classe qui hérite de la classe « Person » et qui dispose des attributs et des opérations supplémentaires concernant le client.
- **La classe « Account » :** c'est la classe qui comporte les informations générales d'un compte.
- **La classe « Checking_Account » :** c'est une classe qui hérite de la classe « Account » et qui contient les attributs concernant le compte courant.
- **La classe « Saving_Account » :** c'est une classe qui hérite de la classe « Account » et qui comporte les attributs supplémentaires d'un compte épargne.
- **La classe « Credit_Card » :** c'est la classe qui représente les informations d'une carte de crédit.
- **La classe « Checkbook » :** c'est la classe qui représente les informations d'un carnet de chèque.
- **La classe « Transaction » :** c'est la classe qui représente une transaction.
- **La classe « Contact » :** c'est la classe qui indique les informations d'un message envoyé par un utilisateur.
- **La classe « Request » :** c'est la classe qui contient les informations d'une demande.
- **La classe « Request_CC » :** c'est une classe qui hérite de la classe « Request » et qui représente une demande de carte de crédit.
- **La classe « Request_CheckBook » :** c'est une classe qui hérite de la classe « Request » et qui représente une demande de carnet de chèque.

3. Les diagrammes de séquences

Ce diagramme représente l'interaction entre l'acteur et le système ainsi que les communications entre les différents composants systèmes.

3.1. Cas d'utilisation « Authentifier »

La figure 4.2 représente les différentes séquences qui se produisent lorsqu'un utilisateur appuie sur le bouton de connexion de l'interface d'authentification, le système client va tout d'abord tester s'il y a un champ vide ou invalide, après il va demander au serveur de vérifier les informations d'authentification afin de recevoir l'identifiant du client pour enfin pouvoir obtenir les informations et les comptes du client.

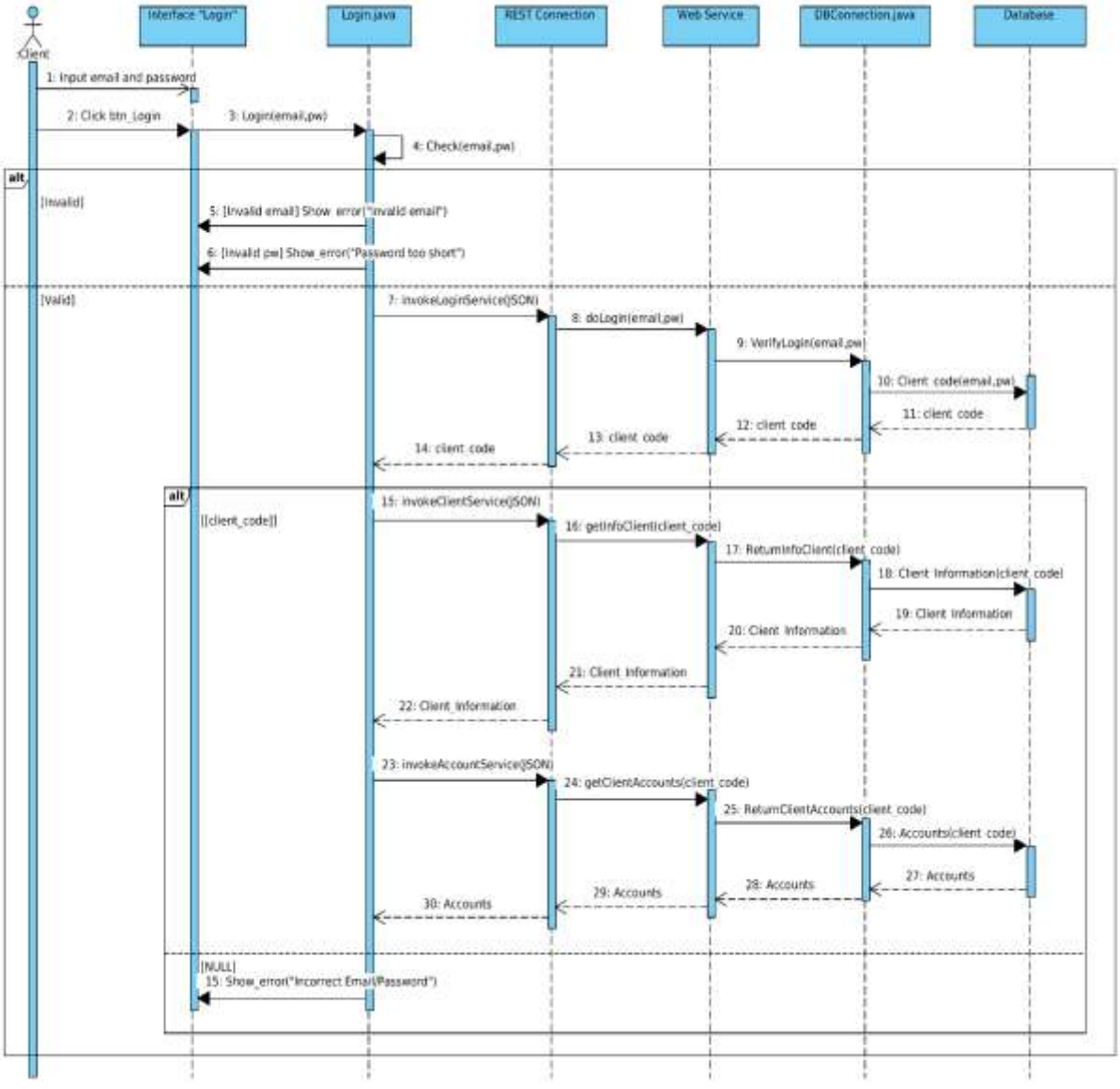

Figure 4.2 : Diagramme de séquence du cas d'utilisation "s'authentifier"

3.2. Cas d'utilisation « Effectuer une demande »

Comme illustre la figure 4.3, quand un client veut demander une carte ou un chéquier il va sélectionner le type de sa demande, après lorsqu'il clique sur le bouton de la demande le système va invoquer le service web pour l'enregistrer sur la base de données et informer le client par son email pour les coordonnées de sa demande.

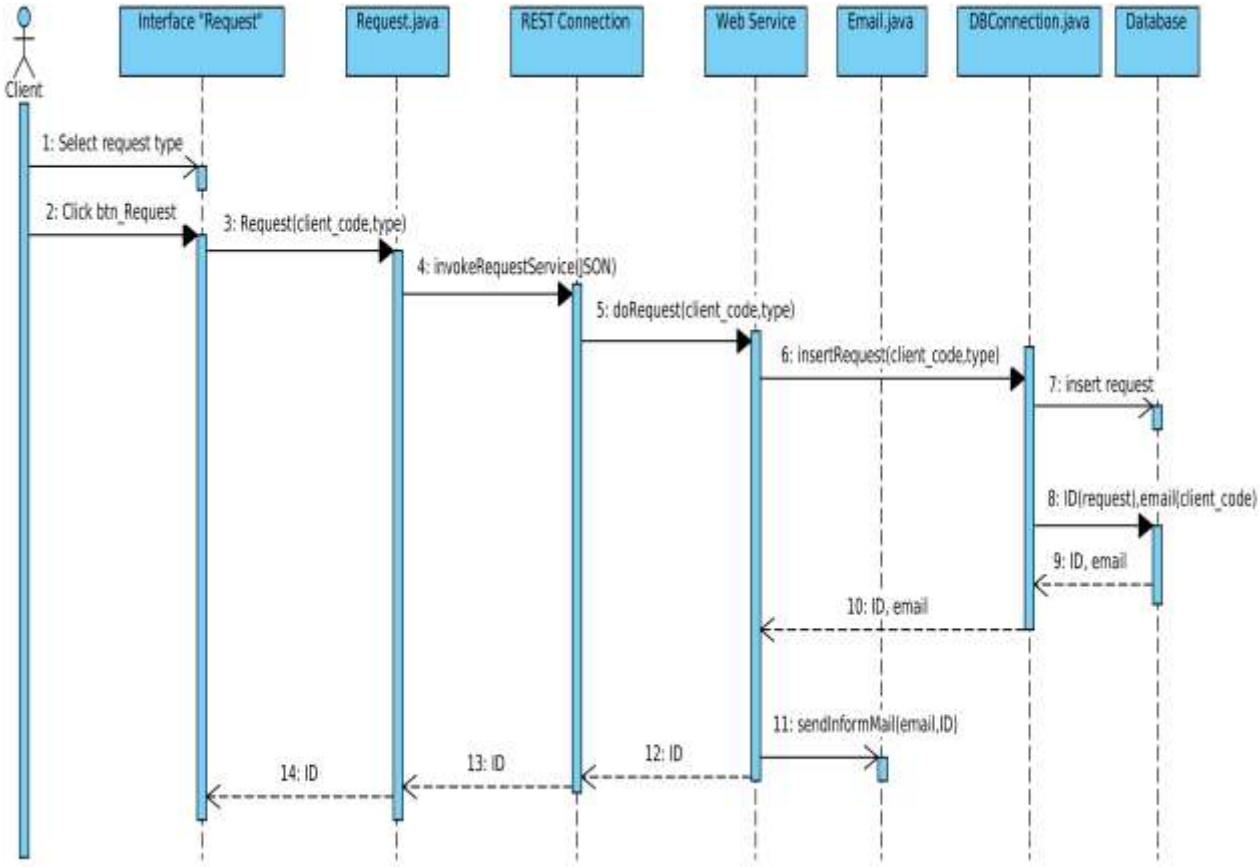

Figure 4.3 : Diagramme de séquence du cas d'utilisation "Effectuer une demande"

3.3. Cas d'utilisation « Voir les transactions »

Lorsqu'un client veut voir les transactions de l'un de ses comptes, il va l'ouvrir à partir du menu de l'application et le système va directement faire appel au service de la transaction pour les récupérer et l'afficher sur l'interface, la figure 4.4 expose ce scénario.

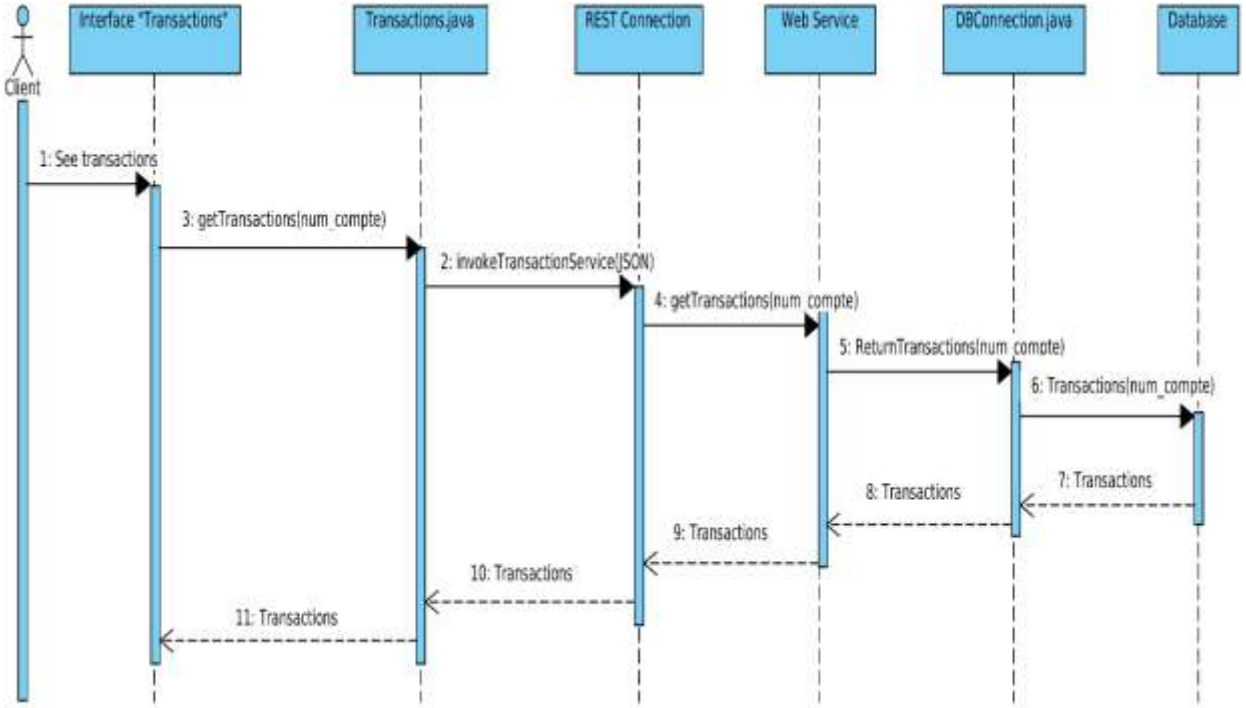

Figure 4.4 : Diagramme de séquence du cas d'utilisation "Voir les transactions"

Cependant le client peut effectuer une recherche avancée sur les transactions du compte choisi, la figure 4.5 nous montre les étapes qui se présentent lors d'une recherche avancée.

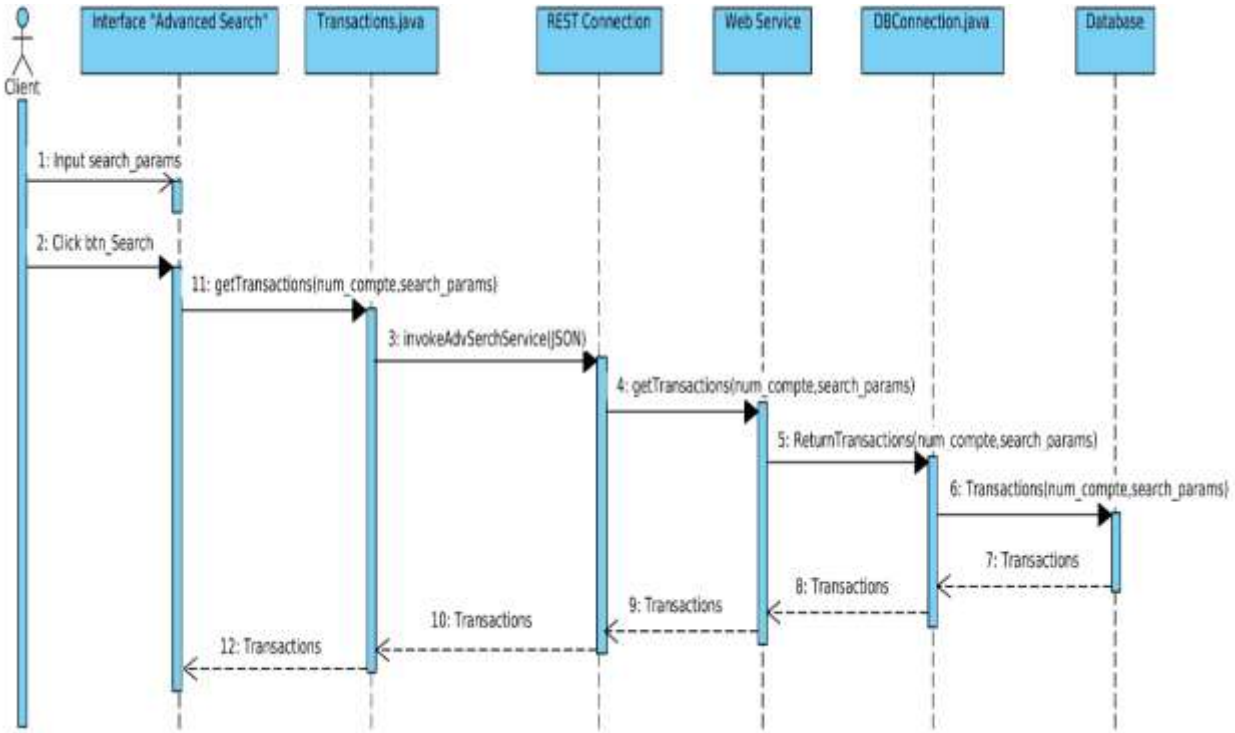

Figure 4.5 : Diagramme de séquence du cas d'utilisation "Recherche Avancée"

3.4. Cas d'utilisation « Contacter l'administrateur »

Pour contacter un administrateur, le client doit remplir un formulaire pour envoyer son message, après le système va enregistrer ces informations et informe le client par email ainsi qu'un simple message d'alerte qui s'affiche sur l'interface de l'application contenant l'identifiant de son message, ces différentes séquences sont exposées sur la figure 4.6.

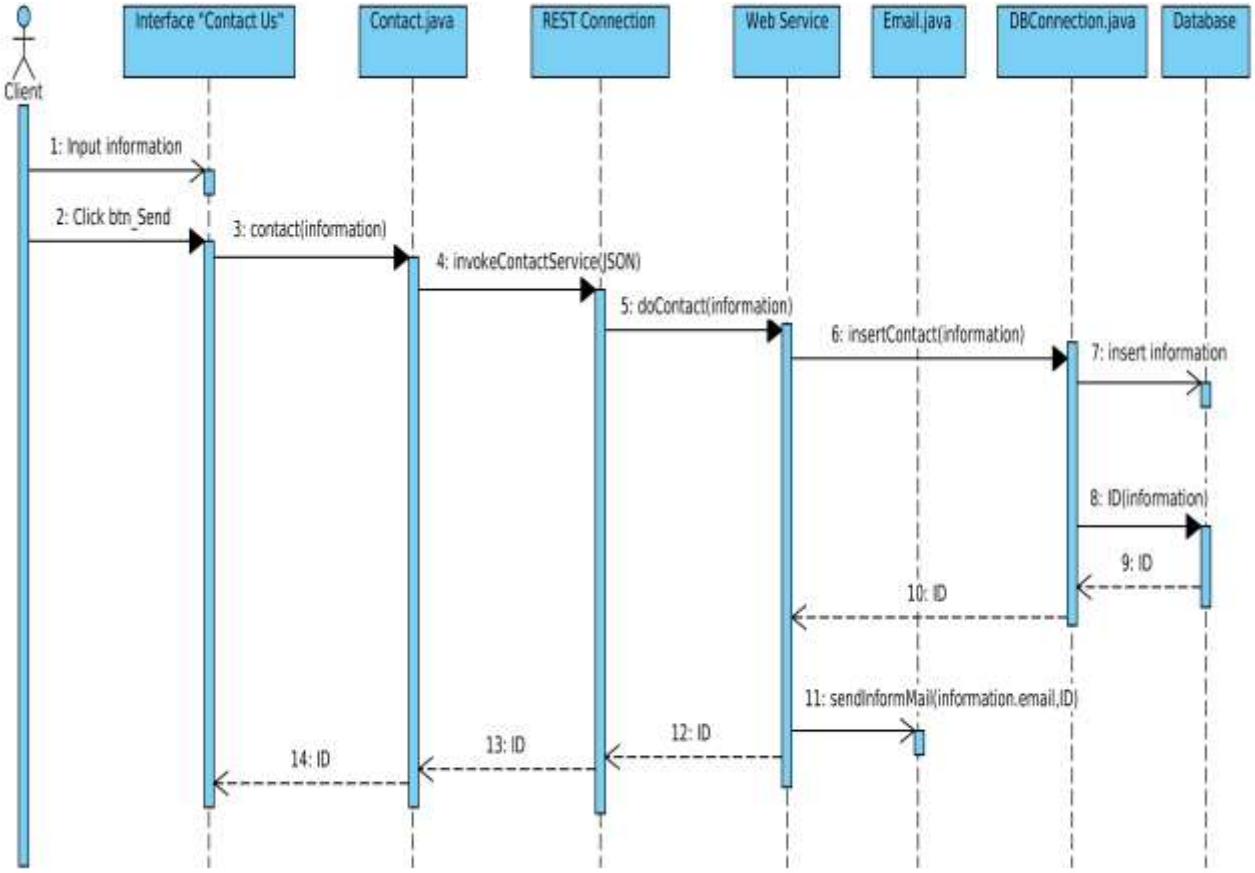

Figure 4.6 : Diagramme de séquence du cas d'utilisation "contacter l'administrateur"

3.5. Cas d'utilisation « Transférer de l'argent »

Le transfert d'argent requiert beaucoup de vérification comme le montre la figure 4.7. Après qu'un client saisit les informations du transfert ou analyse un code QR, le système client va vérifier s'il y a des champs invalides et informe le client par message d'erreur si c'est le cas, sinon il va invoquer le service d'envoi qui va tout de même vérifier le code de sécurité ainsi que la disponibilité du montant envoyé sur le compte de l'expéditeur, si tout est valide alors la transaction aura lieu et le client va être notifié et dans le cas contraire un message d'erreur est envoyé pour le client.

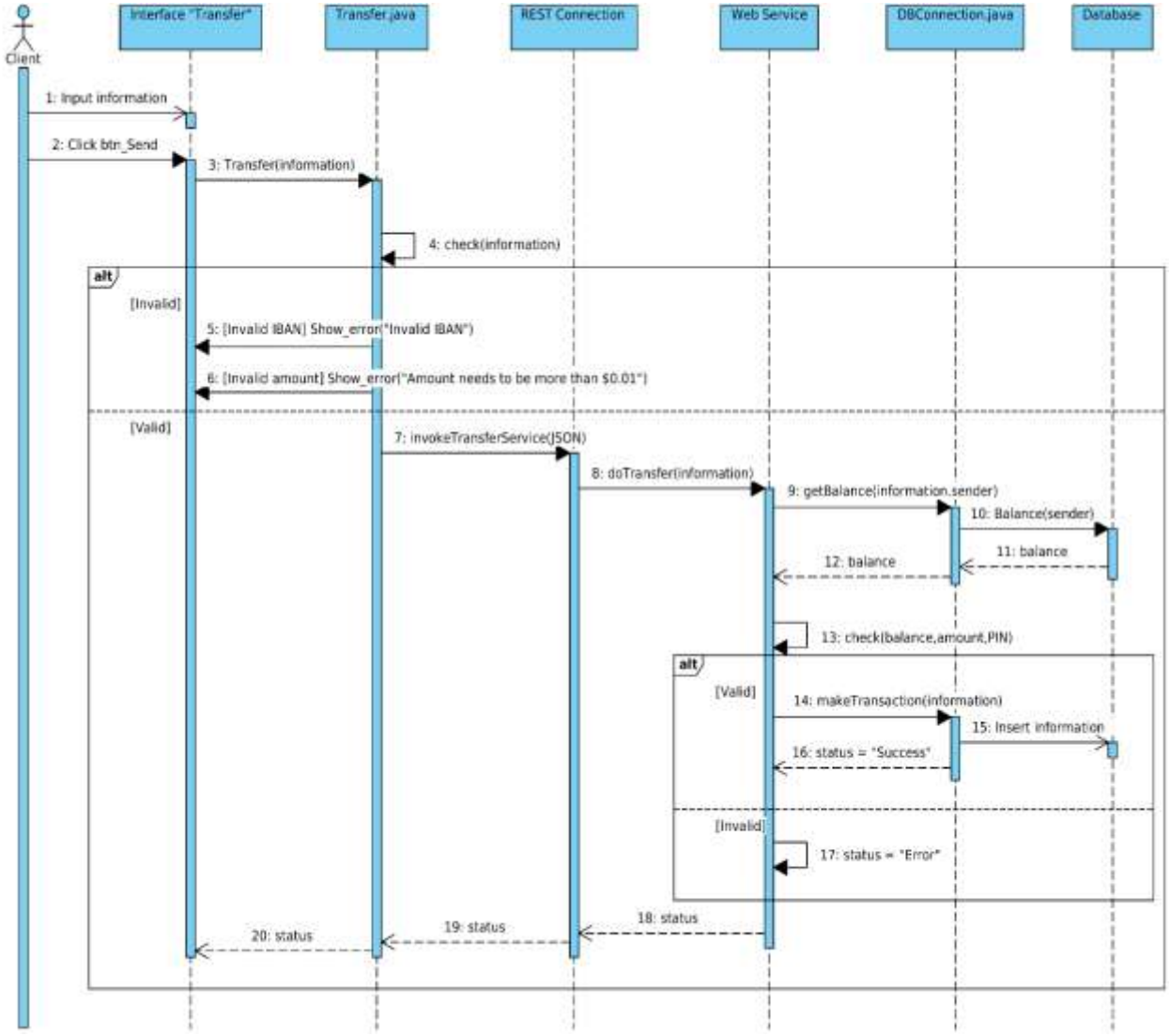

Figure 4.7 : Diagramme de séquence du cas d'utilisation "Transférer de l'argent"

3.6. Cas d'utilisation « Générer une demande de transfert »

Pour faciliter la tâche du transfert, le recevant de l'argent peut générer un code QR pour remplir les champs du transfert automatiquement de l'expéditeur qui analyse ce code.

La figure 4.8 représente les étapes qui se produisent lors d'une demande de transfert. Le client va saisir le montant désiré puis le système va générer le code QR si la somme est supérieure à 1 dollar, par suite le code est enregistré dans l'appareil et le client a l'opportunité de partager ce code.

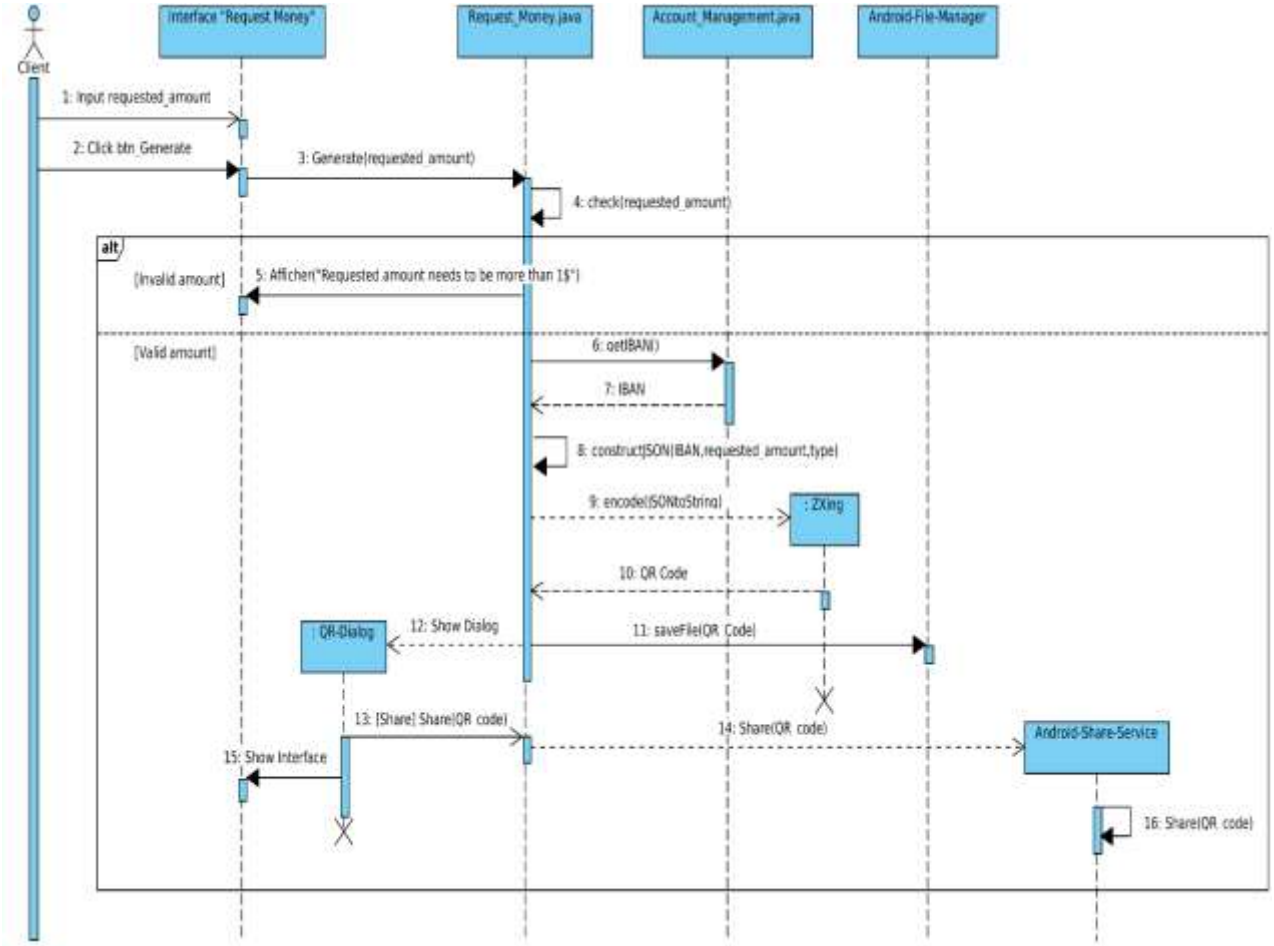

Figure 4.8 : Diagramme de séquence du cas d'utilisation "Générer une demande de transfert"

3.7. Cas d'utilisation « Trouver une agence/un distributeur »

Lorsqu'un utilisateur veut savoir où se trouve l'agence ou le distributeur le plus proche, le système va faire appel au service de localisation pour récupérer les coordonnées des places, puis il va construire une carte avec les positions récupérées et la focaliser sur la position de l'utilisateur. Ces séquences sont représentées sur la figure 4.9.

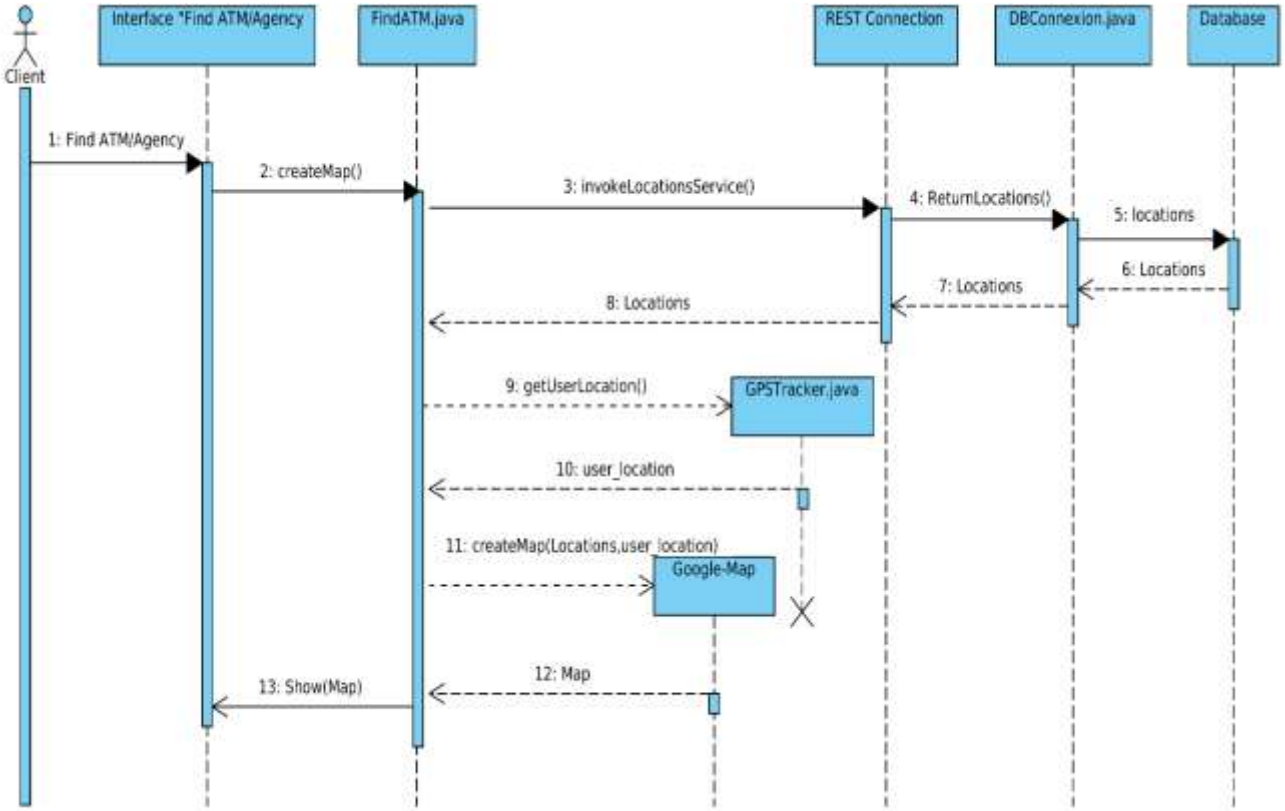

Figure 4.9 : Diagramme de séquence du cas d'utilisation "Trouver agence/distributeur"

4. Modélisation des composants systèmes de l'application mobile

Étant donné la structure du système Android comme nous avons déjà mentionné dans le chapitre 2, nous distinguons des différents prototypes de classes qui construisent les composants de cette architecture.

La figure 4.10 nous montre la liaison d'attachement entre ces différentes classes dans notre cas.

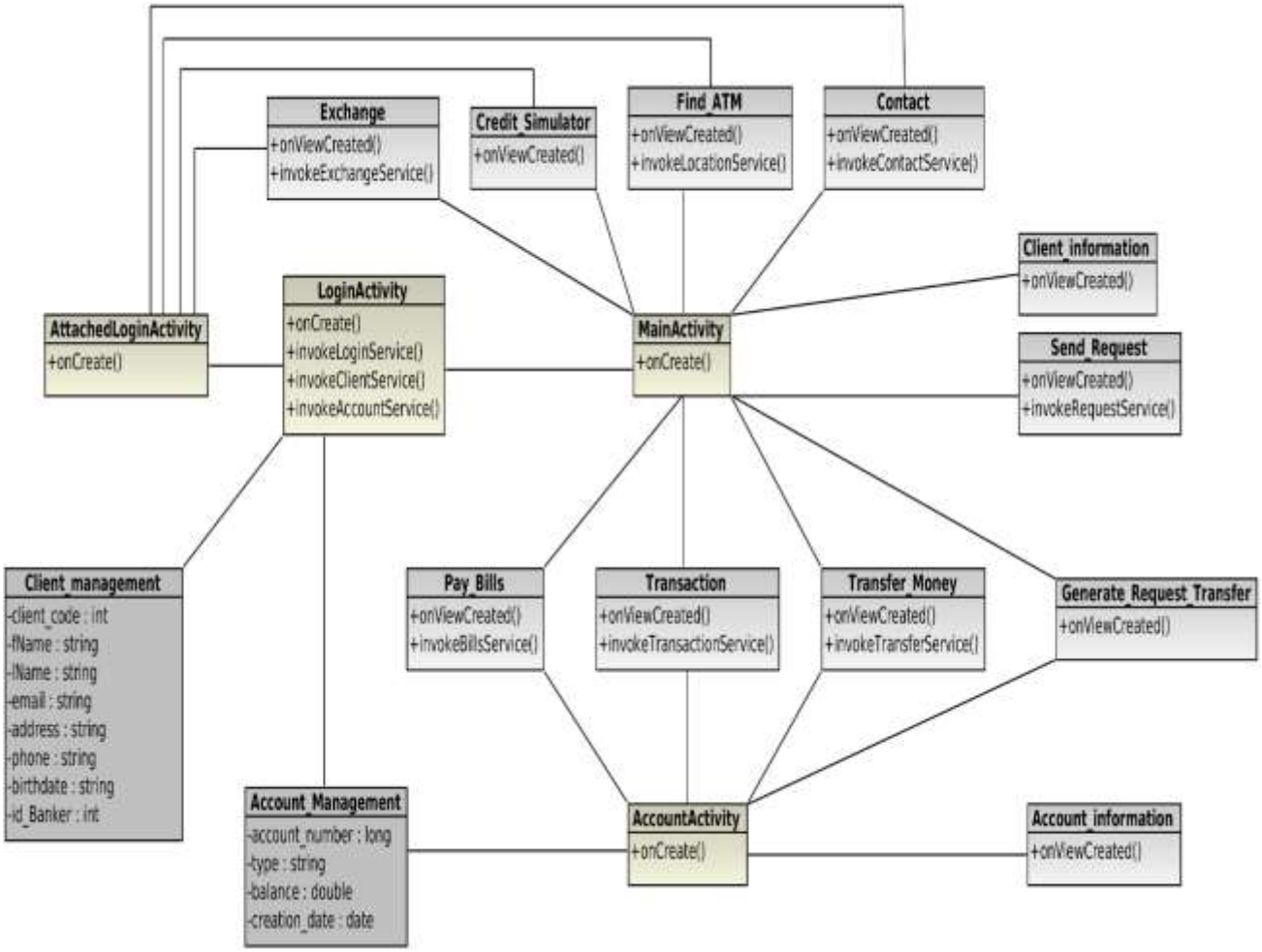

Figure 4.10 : Modélisation des composantes de l'application mobile

4.1. Les activités

Ce sont des classes qui héritent de la classe « Activity » qui sert soit d'afficher des interfaces ou soit d'un conteneur pour les fragments.

Dans notre projet, nous avons les activités suivantes :

- **LoginActivity :** c'est la première activité qui se lance dans le système et elle sert à authentifier l'utilisateur pour qu'il puisse avoir accès à l'activité principale.
- **AttachedLoginActivity :** c'est une activité optionnelle qui sert à lancer des fonctionnalités dont on n'a pas besoin d'être authentifié pour les utiliser.
- **MainActivity :** c'est l'activité principale du projet elle englobe presque tous les services disponibles de l'application.

- **AccountActivity :** c'est l'activité qui se lance quand le client choisit un compte particulier pour effectuer certaines opérations dont il a besoin de s'adhérer à un compte.

4.2. Les fragments

Ce sont des classes qui héritent de la classe « Fragment » et qui sont très légères par rapport aux activités, leur rôle c'est d'afficher des interfaces.

Le tableau 4.1 nous montre les différents fragments de notre projet ainsi que leurs rôles et les activités qui leur sont attachées.

Tableau 4.1 : Liste des fragments

Nom du fragment	Rôle	Activité attaché
Client_information	Permettre au client de voir ses informations ainsi que tous ses comptes	**MainActivity**
Account_information	Permettre au client de voir les informations en détail d'un compte sélectionné	**AccountActivity, MainActivity (***s'il choisit un compte depuis le menu)*
Transfer_Money	Permettre au client de transférer de l'argent à partir d'un compte sélectionné	**AccountActivity, MainActivity (***s'il choisit un compte depuis le menu)*
Transactions	Permettre au client de voir tous ses transactions d'un compte particulier	**AccountActivity, MainActivity (***s'il choisit un compte depuis le menu)*
Generate_Request_ Transfer	Permettre au client de générer une demande de transfert de l'argent sur un compte particulier	**AccountActivity, MainActivity (***s'il choisit un compte depuis le menu)*
Pay_bills	Permettre au client de payer une facture à distance	**AccountActivity, MainActivity (***s'il choisit un compte depuis la menu)*
Send_request	Permettre au client de demander une carte de crédit ou un carnet de chèque	**MainActivity**

Exchange	Permettre aux utilisateurs de voir les échanges de devises	**MainActivity (***client authentifié***), AttachedLoginActivity (***client non-authentifié***)**
Credit_Simulator	Permettre aux utilisateurs de simuler un crédit	**MainActivity (***client authentifié***), AttachedLoginActivity (***client non-authentifié***)**
Find_ATM	Permettre à l'utilisateur de trouvé l'agence ou le distributeur le plus proche	**MainActivity (***client authentifié***), AttachedLoginActivity (***client non-authentifié***)**
Contact	Permettre à l'utilisateur de contacter l'administrateur	**MainActivity (***client authentifié***), AttachedLoginActivity (***client non-authentifié***)**

5. L'architecture du middleware

Dans cette partie nous allons modéliser le service web qui présente le middleware de notre application 3-tièrs.

Comme illustre la figure 4.11 il s'agit des API REST qui communiquent avec les contrôleurs de la couche présentation par des messages parsés en JSON pour récupérer des données stockées dans la couche de données à travers la classe DBConnection.

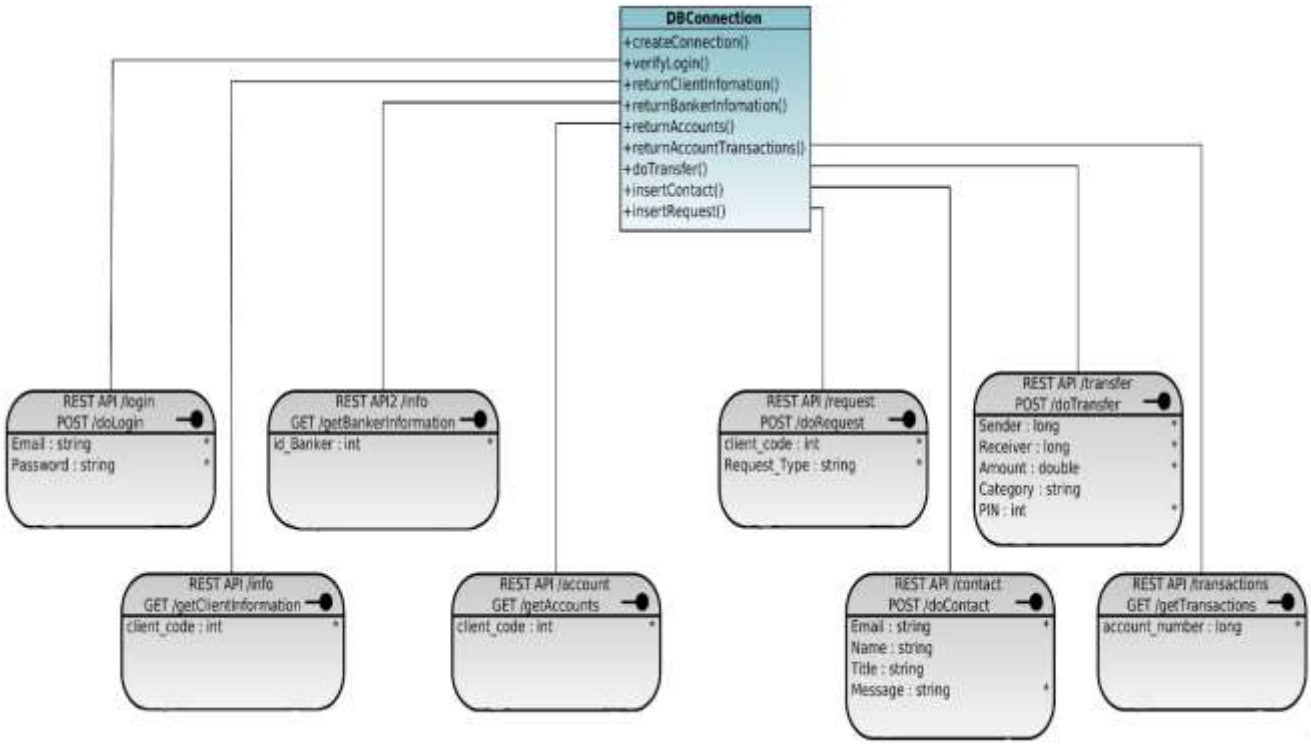

Figure 4.11 : Modélisation des classes du middleware

6. Modèle MVC de la partie administrative

Le principe fondamental du modèle MVC est de répartir le système en trois parties

- Partie donnée
- Partie métier
- Partie présentation

La figure 4.12 montre les différentes classes du système administratif.

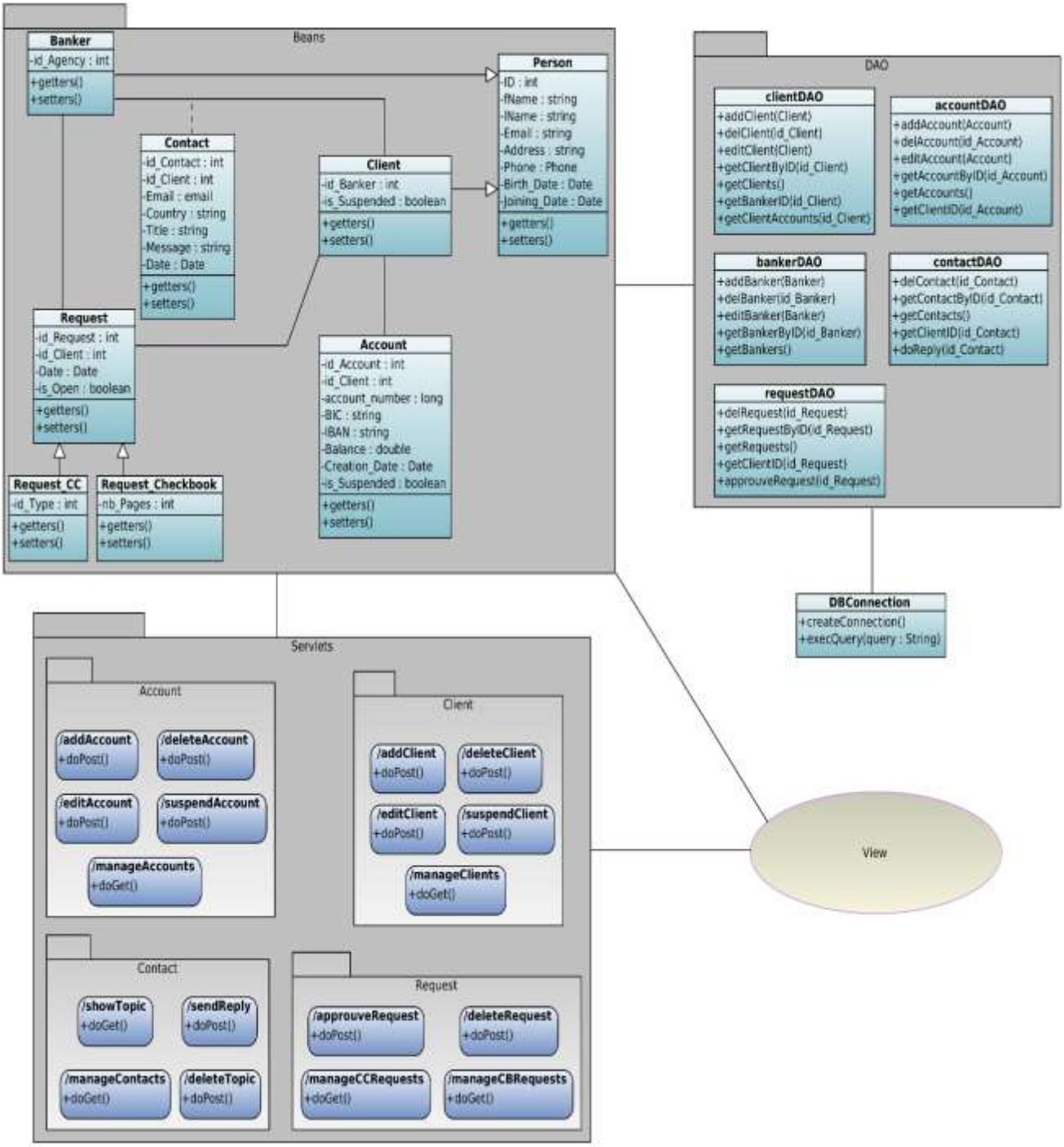

Figure 4.12 : Modélisation du système administratif

6.1. Les modèles

Ce sont des classes contenant des getters et des setters qui modélisent les données, appelé JavaBean.

Pour notre application nous avons les JavaBean suivants :

- **Person :** un Bean qui représente les informations d'une personne.
- **Client :** un Bean qui hérite du Bean Person et qui représente les informations d'un client provenant de la table Client de la base de données.
- **Account :** un Bean qui représente les informations d'un compte provenant de la table Account de la base de données.
- **Banker :** un Bean qui hérite du Bean Person et qui représente les informations d'un banquier provenant de la table Banker de la base de données.
- **Agency :** un Bean qui représente les informations d'une agence provenant de la table Agency de la base de données.
- **Request :** un Bean qui représente les informations d'une demande.
- **Card_Request :** un Bean qui hérite du Bean Request et qui représente les informations d'une demande de carte de crédit provenant de la table Request de la base de données avec le champ type égale à 'Credit card'.
- **Checkbook_Request :** un Bean qui hérite du Bean Request et qui représente les informations d'une demande de carnet de chèque provenant de la table Request de la base de données avec le champ type égale à 'Checkbook'.
- **Contact :** un Bean qui représente les informations d'un sujet de contact provenant de la table Contact de la base de données.

6.2. Les vues

Il s'agit ici des pages de présentation que ce soit des pages HTML ou des pages JSP.

Dans notre cas, nous avons :

- **Login.html :** c'est l'interface d'authentification.
- **Main.jsp :** c'est l'interface globale qui contient le menu de la navigation et qui a un corps dynamique.
- **Main.html :** c'est un corps qui représente l'interface d'accueil après l'authentification.

- **Account :** c'est un dossier qui contient les corps JSP concernant la liste des comptes et l'ajout ou l'édition d'un compte.
- **Client :** c'est un dossier qui contient les corps JSP qui ont rapport avec la gestion des clients.
- **Contact :** c'est un dossier qui comporte les corps JSP relatif à la gestion des relations clientèles.
- **Request :** c'est un dossier qui possède les corps JSP concernant les demandes clientèles.

6.3. Les contrôleurs

Ce sont les servlets et qui jouent le rôle d'un simple aiguilleur.

Comme illustre la figure 4.12, nous avons classé nos servlets selon leur cadre de travail sur des différents packages :

- **Account :** c'est un package qui comporte tous les servlets relatifs aux comptes.
- **Client :** c'est un package qui contient tous les servlets qui concernent la gestion des clients.
- **Contact :** c'est un package qui englobe tous les servlets qui concernent les relations clientèles.
- **Request :** c'est un package qui contient tous les servlets qui ont rapport avec la direction des demandes envoyées par les clients.

7. Base de données

Dans cette section nous allons présenter le schéma de la base de données puis nous allons étudier les champs de chaque table.

7.1. Le modèle conceptuel de données

La figure 4.13 nous montre la liaison entre les différentes tables de notre base de données.

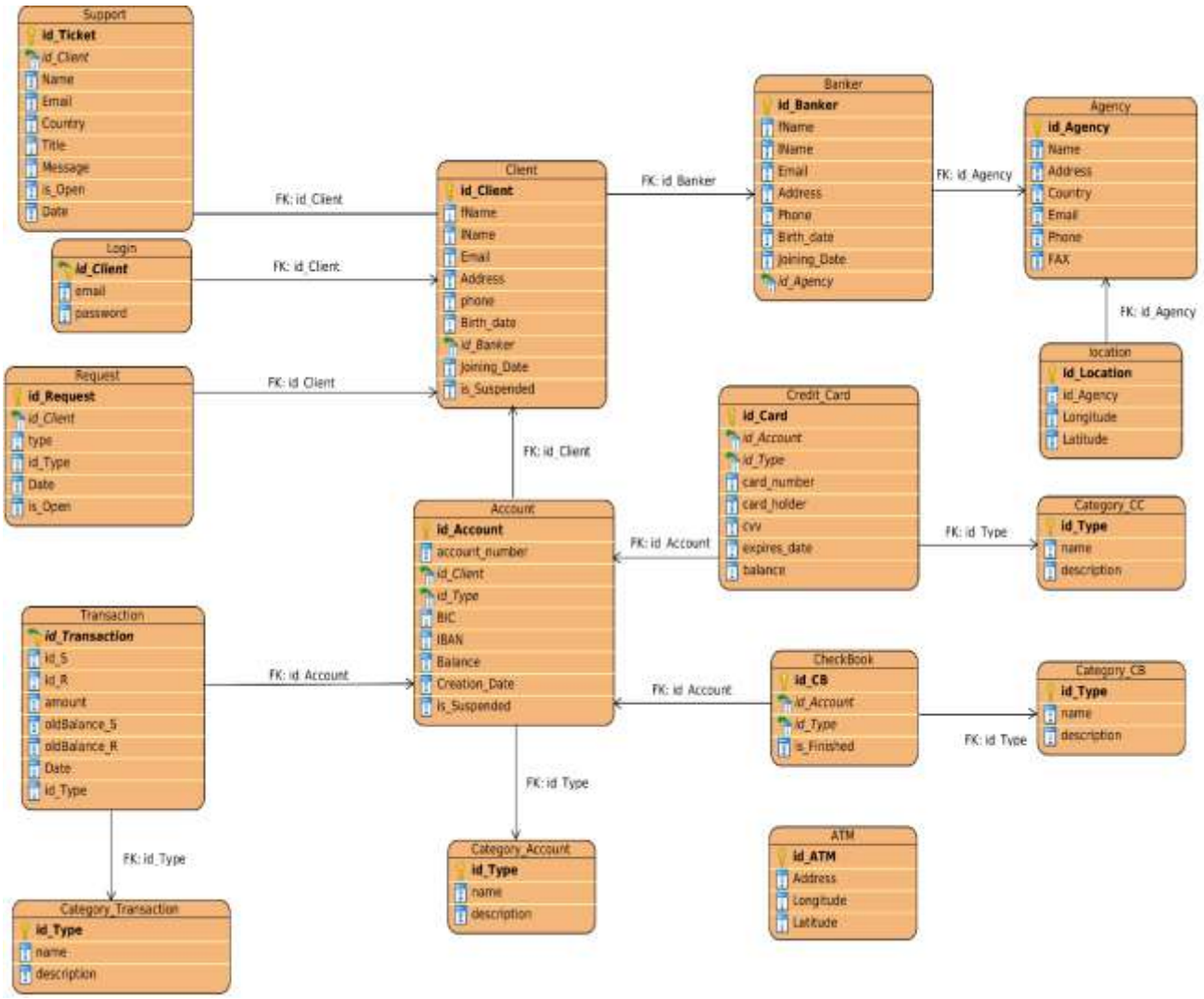

Figure 4.13 : La base de données

7.2. Le dictionnaire de données

Nous présentons ici la signification des différents champs de chaque table de la base de données.

La table « Agency »

Le tableau 4.2 décrit chaque champ de la table qui définisse les agences au niveau de la base de données.

Tableau 4.2 : Table "Agency" de la base de données

Champ	Signification	Type	Observation
id_Agency	Identifiant de l'agence	Entier	Clé primaire
Name	Nom de l'agence	Varchar	
Address	Adresse de l'agence	Varchar	
Country	Pays de l'agence	Varchar	
Email	Email de l'agence	Varchar	
Phone	Téléphone de l'agence	Varchar	
FAX	FAX de l'agence	Varchar	

La table « Banker »

Le tableau 4.3 illustre les significations des champs contenus dans la table des banquiers dans la base de données.

Tableau 4.3 : Table "Banker" de la base de données

Champ	Signification	Type	Observation
id_Banker	Identifiant du banquier	Entier	Clé primaire
fName	Prénom du banquier	Varchar	
lName	Nom du banquier	Varchar	
Email	Email du banquier	Varchar	
Address	Adresse du banquier	Varchar	
Phone	Numéro de téléphone du banquier	Varchar	
Birth_Date	Date de naissance du banquier	Date	
Joining_Date	Date d'engagement du banquier	Date	
#id_Agency	Identifiant de l'agence	Entier	Clé étrangère « Agency »

La table « Client »

Les champs de la table client de la base de données sont décrits dans le tableau 4.4.

Tableau 4.4 : Table "Client" de la base de données

Champ	Signification	Type	Observation
id_client	Identifiant du client	Entier	Clé primaire
fName	Prénom du client	Varchar	
lName	Nom du client	Varchar	
Email	Email du client	Varchar	
Address	Adresse du client	Varchar	
Phone	Numéro de téléphone du client	Varchar	
Birth_Date	Date de naissance du client	Date	
Joining_Date	Date d'inscription du client	Date	
is_Suspended	Savoir si le client est suspendu		
#id_Banker	Identifiant de l'agence	Entier	Clé étrangère « Banker »

La table « Login »

Les informations d'authentifications des utilisateurs sont stockées dans la table login de la base de données, le tableau 4.5 présente la signification de ces champs.

Tableau 4.5 : Table "Login" de la base de données

Champ	Signification	Type	Observation
#id_Client	Identifiant du client	Entier	Clé primaire, Clé étrangère « Client »
#Email	Email du client	Varchar	Clé primaire, Clé étrangère « Client »
Password	Mot de passe du client	Varchar	

La table « Account »

Les champs de la table qui représente les comptes sont expliqués dans le tableau 4.6

Tableau 4.6 : Table "Account" de la base de données

Champ	Signification	Type	Observation
id_Account	Identifiant du compte	Entier	Clé primaire
account_number	Numéro du compte	Entier long	
#id_Client	Identifiant du client	Entier	Clé étrangère « Client »
#id_Type	Type du compte	Entier	Clé étrangère « Category_Account »
BIC	Relevé d'identité bancaire	Varchar	
IBAN	Numéro du compte international	Varchar	
Balance	Solde du compte	Double	
Creation_Date	Date de création du compte	Date	
is_Suspended	Savoir si le compte est suspendu	Entier	

La table « Transaction »

Le tableau 4.7 illustre l'explication des champs de la table des transactions de la base de données.

Tableau 4.7 : Table "Transaction" de la base de données

Champ	Signification	Type	Observation
id_Transaction	Identifiant de la transaction	Entier	Clé primaire
#id_S	Identifiant du compte de l'expéditeur	Entier	Clé étrangère « Account »
#id_R	Identifiant du compte du destinataire	Entier	Clé étrangère « Account »
Amount	Montant envoyé	Réel	
oldBalance_S	Ancien solde de l'expéditeur	Réel	
oldBalance_R	Ancien solde du destinataire	Réel	
Date	Date de la transaction	Date	
#id_Type	Identifiant du type	Entier	Clé étrangère « Category_Transaction »

La table « Credit_Card »

Dans le tableau 4.8, nous trouvons les significations des champs qui décrivent une carte de crédit.

Tableau 4.8 : Table "Credit_Card" de la base de données

Champ	Signification	Type	Observation
id_Card	Identifiant de la carte	Entier	Clé primaire
#id_Account	Identifiant du compte	Entier	Clé étrangère « Account »
#id_Type	Identifiant du type de la carte	Entier	Clé étrangère « Category_CC »
Card_Number	Numéro de la carte	Entier long	
Card_Holder	Nom attribué à la carte	Varchar	
cvv	Valeur de vérification de la carte	Entier	
Expires_Date	Date d'expiration de la carte	Date	
Balance	Solde de la carte	Double	

La table « CheckBook »

Le tableau 4.9 explique les champs de la table des carnets de chèques.

Tableau 4.9 : Table "CheckBook" de la base de données

Champ	Signification	Type	Observation
id_CB	Identifiant du carnet de chèque	Entier	Clé primaire
#id_Account	Identifiant du compte	Entier	Clé étrangère « Account »
#id_Type	Identifiant du type du carnet de chèque	Entier	Clé étrangère « Category_CB »
is_Finished	Savoir si le carnet est fini	Entier	

La table « Support »

Les champs de table des messages clientèles sont représentés dans le tableau 4.10

Tableau 4.10 : Table "Support" de la base de données

Champ	Signification	Type	Observation
id_Ticket	Identifiant du sujet	Entier	Clé primaire
#id_Client	Identifiant du client	Entier	Clé étrangère « client »
Name	Nom d'utilisateur	Varchar	
Email	Email d'utilisateur	Varchar	
Country	Pays de l'utilisateur	Varchar	
Title	Titre du sujet	Varchar	
Message	Message du sujet	Varchar	
is_Open	Savoir si le sujet est encore ouvert	Entier	
Date	Date de création de sujet	Date	

La table « Request »

Le tableau 4.11 explique les champs d'une demande.

Tableau 4.11 : Table "Request" de la base de données

Champ	Signification	Type	Observation
id_Request	Identifiant de la demande	Entier	Clé primaire
#id_Client	Identifiant du client	Entier	Clé étrangère « Client »
Type	Type de la demande : - Carte de crédit - Carnet de chèque	Varchar	
#id_Type	Identifiant du type	Entier	Clé étrangère « Category_CC » Ou « Category_CheckBook »
Date	Date de la demande	Date	
is_Open	Savoir si la demande est encore ouverte	Entier	

La table « Location »

Les champs de la table de localisation des agences sont représentés dans le tableau 4.12.

Tableau 4.12 : Table "Location" de la base de données

Champ	Signification	Type	Observation
id_Location	Identifiant de la position	Entier	Clé primaire
#id_Agency	Identifiant de l'agence	Entier	Clé étrangère « Agency »
Longitude	Longitude de la position	Réel	
Latitude	Latitude de la position	Réel	

La table « ATM »

Le tableau 4.13 illustre la signification des champs qui représente un distributeur.

Tableau 4.13 : Table "ATM" de la base de données

Champ	Signification	Type	Observation
id_ATM	Identifiant du distributeur	Entier	Clé primaire
Address	Adresse du distributeur	Varchar	
Longitude	Longitude de la position du distributeur	Réel	
Latitude	Latitude de la position du distributeur	Réel	

Les tables « Category »

Les champs des tables de catégorie sont décrits dans le tableau 4.14.

Tableau 4.14 : Les Tables "Category" de la base de données

Champ	Signification	Type	Observation
id_Type	Identifiant du type	Entier	Clé primaire
Name	Nom du type	Varchar	
Description	Description du type	Varchar	

8. Conclusion

Dans ce chapitre nous avons représenté les différentes parties de notre projet par des diagrammes et des schémas.

Par suite, nous allons aborder l'étape de la réalisation du projet.

Chapitre 5 : Réalisation

1. Introduction

Dans ce chapitre nous allons présenter les différents environnements physiques et logiciel puis nous allons aborder les différentes étapes du développement.

2. Environnement du développement

Ce sont les différentes composantes logicielles et matérielles nécessaires au développement.

2.1. Environnement matériel

Les moyens mis à notre disposition dans le cadre de la réalisation du projet sont nos deux machines HP et Samsung, ainsi qu'un serveur VPS hébergé chez Amazon.

- **HP :**
  - ✓ Un processeur Intel® Core i3-3110M @ 2.40GHz
  - ✓ Une mémoire vive de 6 GB
  - ✓ Système d'exploitation : Linux Mint 64-bits, Windows 10 (x64)
- **Samsung :**
  - ✓ Un processeur Intel® Core i3-2350M @ 2.30GHz
  - ✓ Une mémoire vive de 4 GB
  - ✓ Système d'exploitation : Linux Mint 64-bits, Windows 10 (x64)
- **VPS :**
  - ✓ Intel Xeon Family @ 2.50GHz
  - ✓ Une mémoire vive de 1GB
  - ✓ Système d'exploitation : Amazon Linux AMI 2016.09.1 (HVM)

2.2. Environnement logiciel

Le projet est composé de trois parties logiciel, l'application Android pour les clients, les services web nécessaires au fonctionnement de l'application et l'application Web pour les administrateurs de la banque.

Les définitions sont prises à partir de l'adresse électronique ***[7]***.

2.2.1. Technologies utilisées

- **Android**

C'est un système d'exploitation mobile basé sur un Linux modifié par Google.

- **Gradle**

Gradle est un système de construction (Build system) de JVM qui prend les fonctionnalités des autres systèmes comme Maven et les combines en un, pour enfin avoir un système meilleur qui s'améliore à partir des défauts de ses prédécesseurs.

Ce build est intégré par default dans Android Studio et il met une structure bien organisée de fichier.

Arborescence d'un projet gradle :

Comme nous avons mentionné dans le chapitre 2, Android est composé des activités et des fragments. Ces 2 composantes sont situées dans le répertoire « java » comme on le voit sur la figure 5.1.

Dans le répertoire « res » nous trouvons toutes les ressources qui concernent la présentation, comme :

- layout : ce répertoire regroupe tous les fichiers XML liées aux interfaces des fragments et activités.
- drawable : c'est un répertoire qui regroupe toutes les images et forme géométrique qu'utilise l'application.
- values : c'est le répertoire qui contient tous les fichiers XML des valeurs utilisées dans le projet. Comme strings.xml pour les chaines de caractères, colors.xml pour les couleurs ou encore style.xml pour les thèmes.

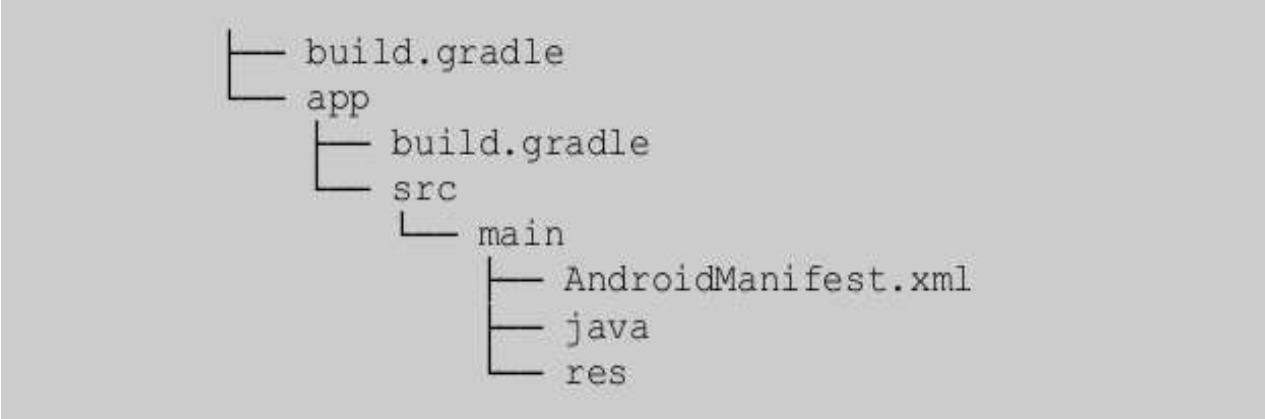

Figure 5.1 : Arborescence des fichiers avec Gradle

- **Plateforme J2EE**

C'est une plateforme Java destinée aux applications d'entreprise.

- **Framework Jersey**

« Jersey » est un framework open-source écrit en « JAVA » développer par « Oracle » qui permet de développer des service web selon l'architecture « REST » suivant les spécifications de JAX-RS.

- **MySQL**

C'est un système de gestion de bases de données relationnelles.

- **Langage Java**

C'est un langage informatique de programmation orienté objet qui permet de créer des programmes, des applications mobiles, etc.

- **Langage XML**

C'est un langage informatique de balisage qui sert à organiser les données de manière structurée et hiérarchisée.

- **Langage HTML5**

C'est un langage informatique utilisé pour la construction des pages web.

- **Bootstrap**

C'est un Framework CSS et JavaScript qui contient un ensemble de codes utiles qui facilite la tâche du design d'une page web.

2.2.2. Bibliothèques utilisées

- **JDBC**

C'est une interface de programmation qui permet aux applications Java d'accéder à la base de données.

- **JavaMail**

C'est un API utilisée pour la gestion des emails destinée à la plateforme J2EE, et il fournit des paquets optionnels pour l'utilisation sur la plateforme Java SE.

- **Zxing**

C'est un projet open-source qui implémente des bibliothèques 1D/2D de code à barre.

- **AsyncHTTPClient**

C'est une bibliothèque qui permet aux applications Java d'exécuter des requêtes HTTP et recevoir des réponses asynchrones.

- **Material Drawer**

Une bibliothèque qui propose un menu similaire à celui utilisé dans les applications Android de Google. ***[8]***

- **Android Iconics**

Cette bibliothèque propose un grand nombre d'icônes de plusieurs source différentes (Material icons, Font Awesome, …). ***[8]***

- **FastAdapter**

Une bibliothèque qui facilite la création des adaptateurs pour les interfaces qui affichent des listes. ***[8]***

2.2.3. Outils utilisés

- **Android Studio**

C'est l'environnement de développement le plus conseillé par Google pour développer des applications Android. Il est rapide, simple et bien organisé, c'est pour ces raisons que nous l'avons utilisé tout au long du développement de l'application mobile.

- **Eclipse**

C'est un environnement de développement très connu dans le domaine de la programmation, il sert à développer avec des différents langages mais fondamentalement en Java et pour cela que nous l'avons choisi pour le développement des Web services ainsi que pour le développement de l'application web.

- **Apache Tomcat**

C'est un serveur d'applications Java qui joue le rôle d'un conteneur web libre de servlets et des JSP. Il comporte aussi un serveur http.

Nous l'avons choisi parce qu'il est léger, gratuit et assez complet pour ce que nous voulons.

- **Xampp**

C'est un logiciel qui englobe un ensemble d'outils nécessaire pour le fonctionnement d'une application, comme : MySQL, Apache et Tomcat.

- **Filezilla**

C'est un client FTP qui nous permet d'échanger des fichiers avec la machine VPS.

- **SSH Client**

C'est un outil de connexion sur le protocole SSH qui nous permet d'utiliser le terminal de la machine VPS à distance.

- **MySQL Wrokbench**

C'est un outil de connexion MySQL à distance, il nous a permis de gérer la base de données hébergé.

- **Visual paradigm**

C'est un outil de modélisation en UML utilisée pendant la phase de conception.

- **Microsoft Word**

C'est un éditeur de texte utilisé pour la rédaction du rapport.

3. Le backlog du projet

Le backlog de projet est la liste des fonctionnalités attendues d'un produit. Le tableau 5.1 illustre le backlog de notre cas.

Tableau 5.1 : Le backlog du projet

Id	Nom	En tant que	Je veux qu'	Pour
1	Authentification	Client	Il soit possible de se connecter à l'application grâce à un login et mot de passe.	Avoir accès à l'application mobile.
		Administrateur		Gérer les données de l'application mobile.
2	Gestion Administrative	Administrateur	Il soit possible d'avoir un accès administratif.	Pouvoir contrôler les clients de la banque ainsi que leurs comptes et cartes bancaires.
3	Gestion des demandes	Administrateur	Il soit possible de gérer les demandes de carte de crédit et de carnet de chèque.	Pouvoir les approuver ou désapprouver.
4	Gestion des messages	Administrateur	Il soit possible de gérer les messages envoyés par les clients.	Pouvoir les répondre ou supprimer.
5	Gestion des comptes	Client	Il soit possible de voir mes anciennes transactions.	Contrôler mes comptes.
			Il soit possible de transférer de l'argent	Effectuer des transactions à distance.
			Il soit possible de demander une carte de crédit ou un carnet de chèque.	Eviter le déplacement vers l'agence.
6	Gestion des services	Client	Il soit possible de visualiser et convertir de la devise	Reconnaitre le cours de change en temps réel.
			Il soit possible de simuler des crédits	Avoir une idée sur la somme à rembourser.
			Il soit possible de contacter la banque	Avoir de l'aide en cas de besoin.

			Il soit possible de localiser les agences et les distributeurs	Trouver les plus proches.

Le projet a été divisé en 2 releases :

- **Release 1 :** Application web

Le premier release s'intitule sur la gestion des données du projet par une application web. Le tableau 5.2 illustre la distribution des sprints pour cette release.

Tableau 5.2 : Distribution des sprints (Release 1)

Id	Nom	Durée estimée (jour)	Durée réel (jour)
Sprint 0	Authentification	2	1
Sprint 1	Gestion Administrative	7	4
Sprint 2	Gestions des demandes	4	2
Sprint 3	Gestion des messages	2	1

- **Release 2 :** Application mobile

Après que nous avons organisé nos données, la deuxième release c'est l'application mobile qui va consommer ces données. Le tableau 5.3 nous montre la distribution des sprints de la deuxième release.

Tableau 5.3 : Distribution des sprints (Release 2)

Id	Nom	Durée estimée (jour)	Durée réel (jour)
Sprint 0	Authentification	4	2
Sprint 1	Gestion des comptes	20	12
Sprint 2	Gestions des services	20	13

4. Interface et utilisation de l'application

Dans cette section, nous exposerons les différentes interfaces de nos applications ainsi que leurs descriptions.

4.1. Release 1 : L'application web

- **Sprint 0 :** Authentification

L'interface de la figure 5.2 est mise à la disposition de l'administrateur pour qu'il puisse s'authentifier pour gérer les données systèmes. Il doit juste saisir son nom d'utilisateur et son mot de passe.

Figure 5.2 : Interface d'authentification web

- **Sprint 1 :** Gestion administrative

L'administrateur peut gérer les clients, les comptes et les cartes. Il peut donc ajouter, supprimer, modifier et suspendre des clients comme on le voit sur la figure 5.3 de même pour les comptes, mais pour les cartes il ne peut que les modifier et suspendre.

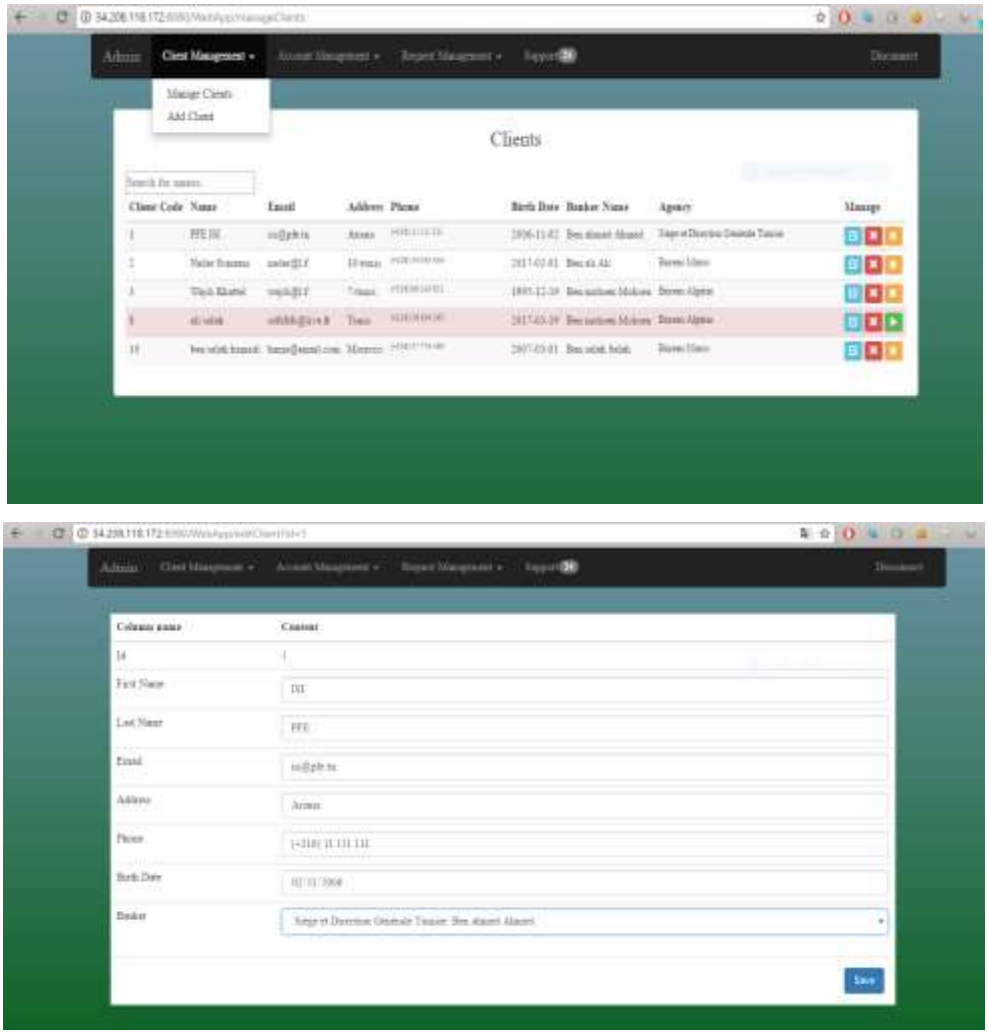

Figure 5.3: Interfaces de gestions des clients

- **Sprint 2 :** Gestion des demandes

Pour la gestion des demandes l'administrateur peut soit approuver la demande pour que la carte ou le chéquier s'envoie chez l'adresse du client, soit la supprimer. La figure 5.4 illustre l'interface offerte pour ce scénario.

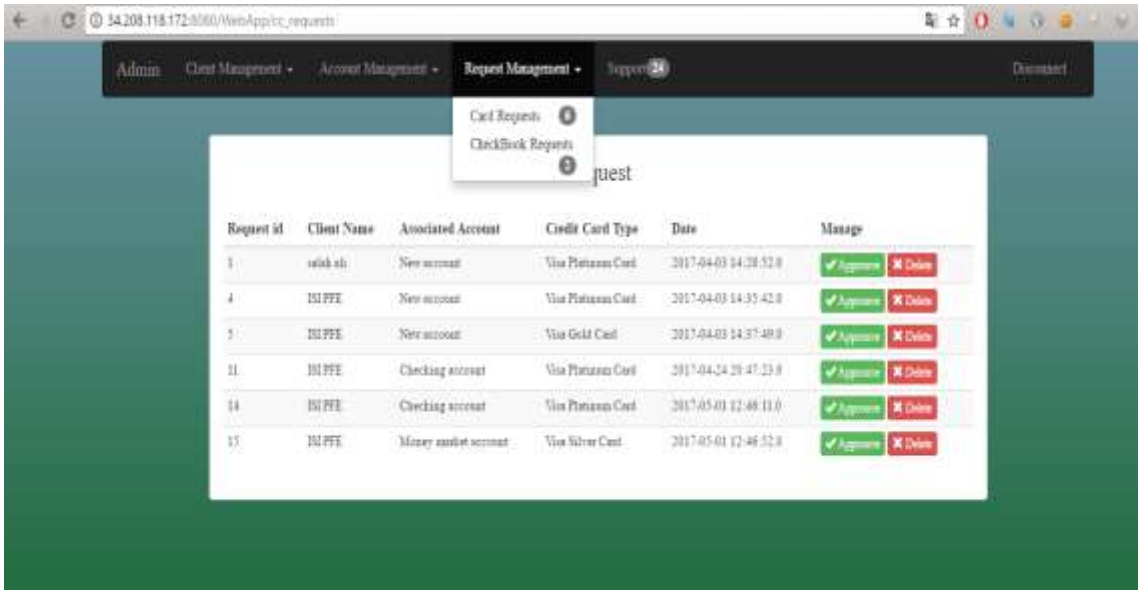

Figure 5.4 : Interface de demandes

- **Sprint 3 :** Gestion des messages

Les messages envoyés par les clients sont sur l'interface qui s'affiche sur la figure 5.5. Pour voir et répondre à un sujet, l'administrateur doit sélectionner ce sujet et puis l'interface de la figure 5.6 est mise à sa disposition pour lui permettre de gérer le message.

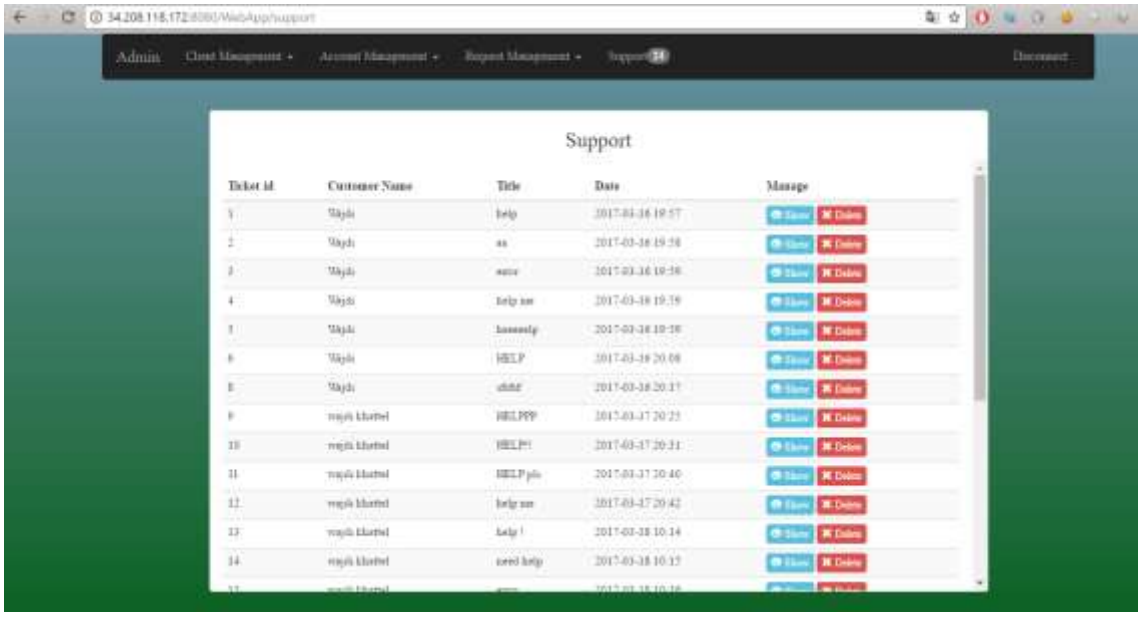

Figure 5.5 : Interface des messages

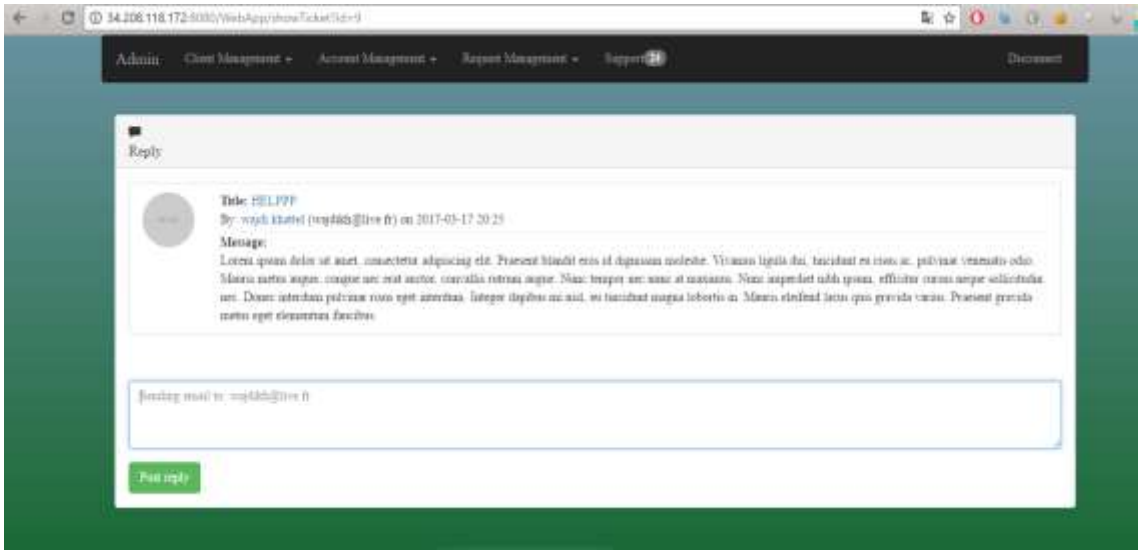

Figure 5.6 : Interface d'un message sélectionné

4.2. Release 2 : L'application mobile

- **Sprint 0 :** Authentification

Lors de l'exécution de l'application, la première interface qui s'affiche c'est celle de l'authentification qui contient deux champs pour l'email et le mot de passe et un bouton de connexion comme sur la figure 5.7, ainsi qu'un menu flottant qui propose les services disponibles sans être authentifié comme nous le voyons sur la figure 5.8.

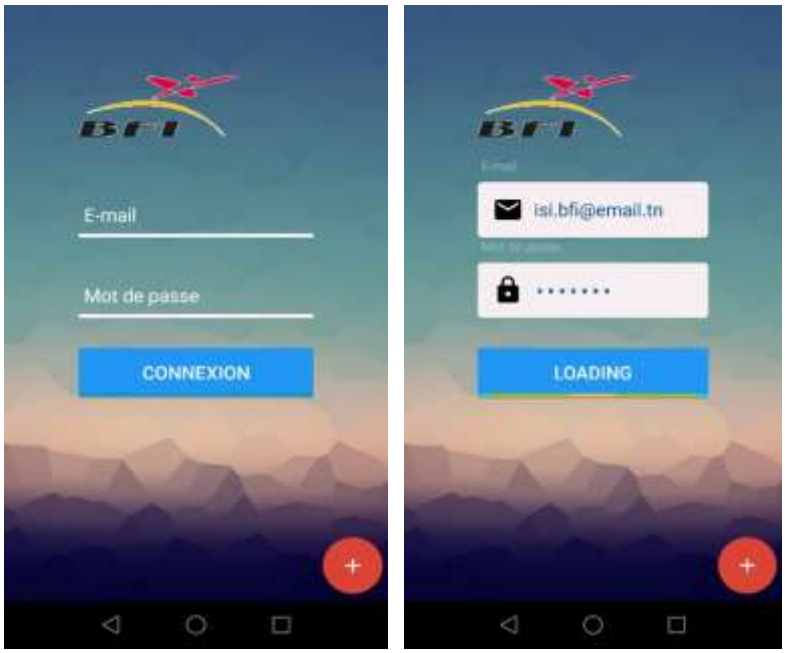

Figure 5.7 : Interface d'authentification mobile

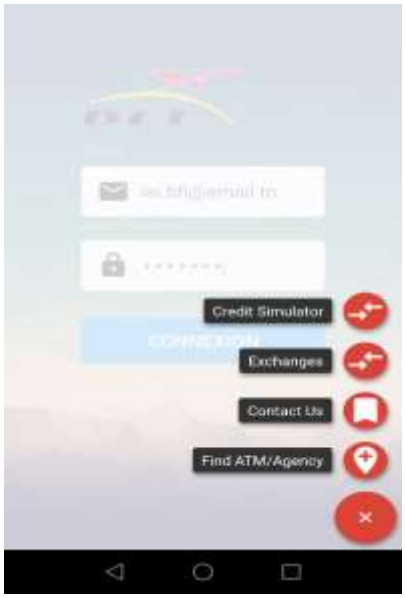

Figure 5.8 : Menu supplémentaire

Après qu'un utilisateur soit authentifié, l'interface de ses informations s'affiche sur son écran et il peut par suite accéder à ses comptes comme le montre la figure 5.9.

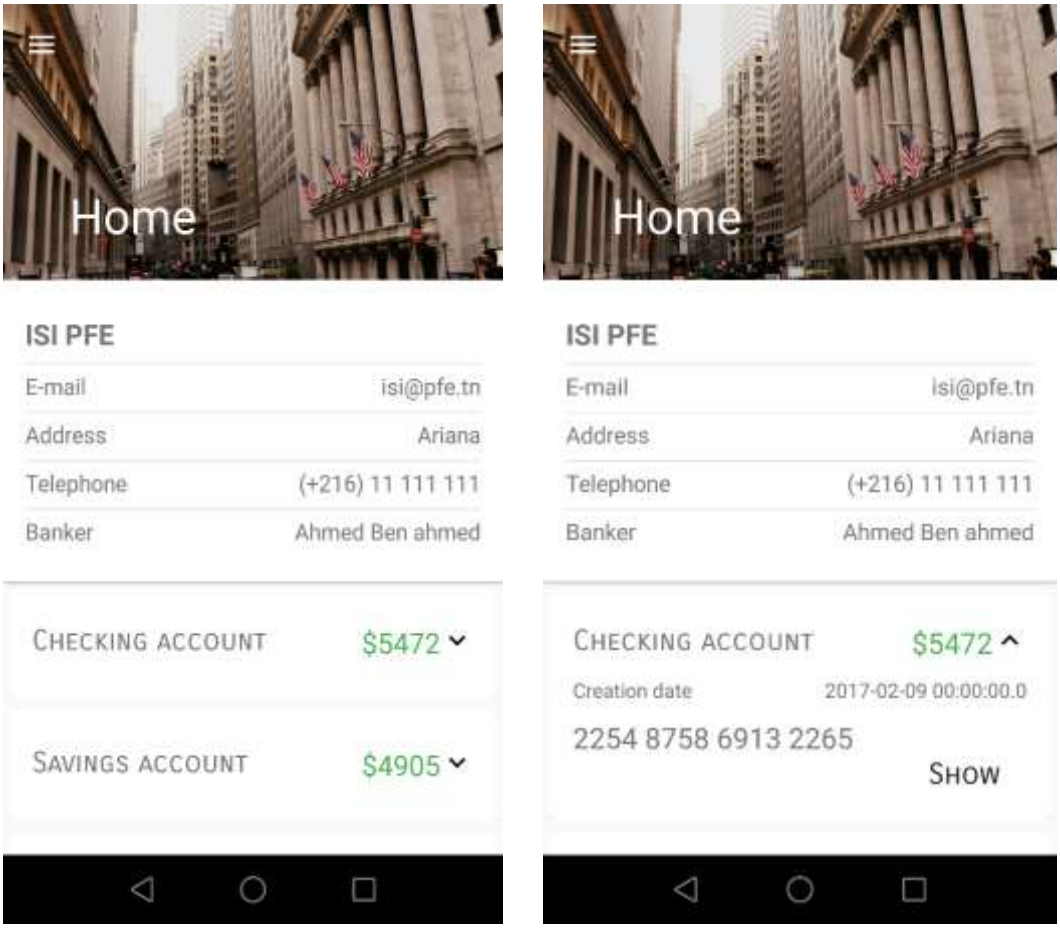

Figure 5.9 : Interface principale

- **Sprint 1 :** Gestion des comptes

Quand le client clique sur le bouton « show » de l'interface de la figure 5.9, il accède directement aux fonctionnalités liées au compte sélectionné :

- **Les informations du compte**

La figure 5.10 représente l'interface qui montre les informations générales d'un compte comme le solde et les différentes adresses de transfert.

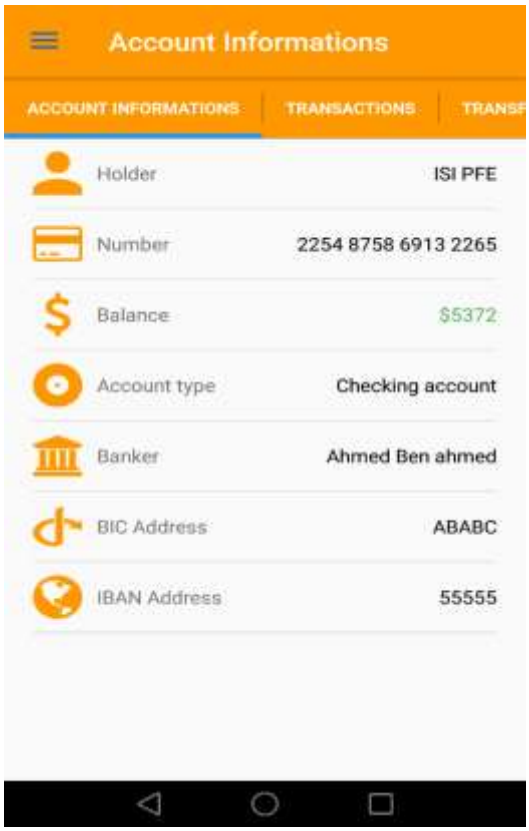

Figure 5.10 : Interface d'information du compte

- **Les transactions du compte**

Comme l'illustre la figure 5.11, le client peut voir les détails d'une transaction et il peut procéder à une recherche avancée.

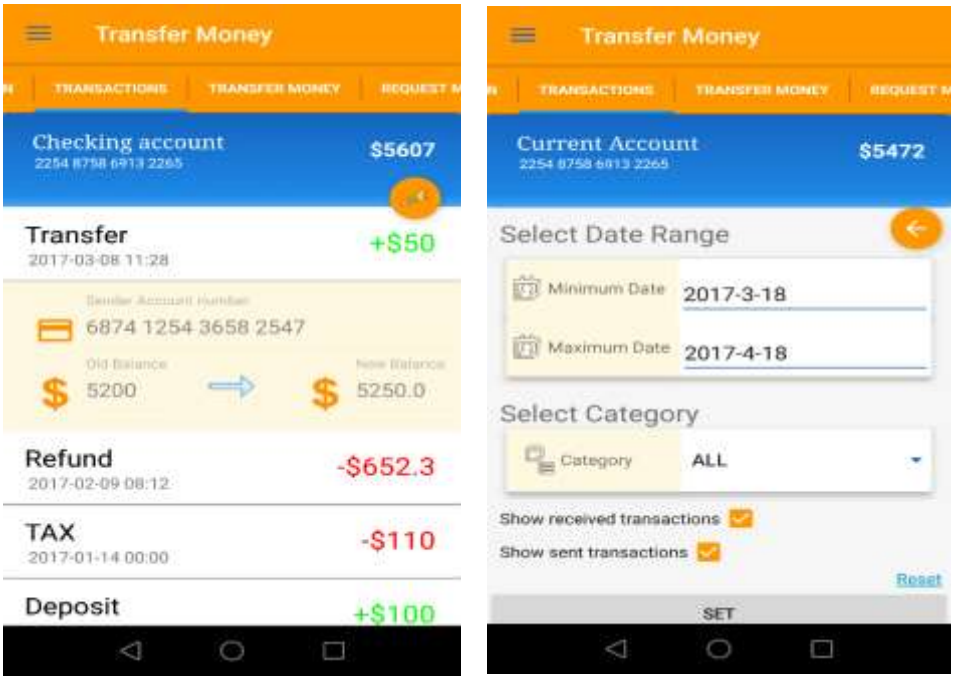

Figure 5.11 : Interface des transactions

- **Le transfert à partir de ce compte**

Pour transférer l'argent le client doit tout d'abord choisir s'il s'agit d'une adresse de destination de type relevé d'identité bancaire RIB (BIC en anglais) ou d'un numéro de compte international IBAN et après qu'il remplit les champs du transfert ou qu'il analyse un code QR il doit saisir son code PIN de sécurité.

La figure 5.12 montre l'interface mis à la disposition du client pour le transfert d'argent.

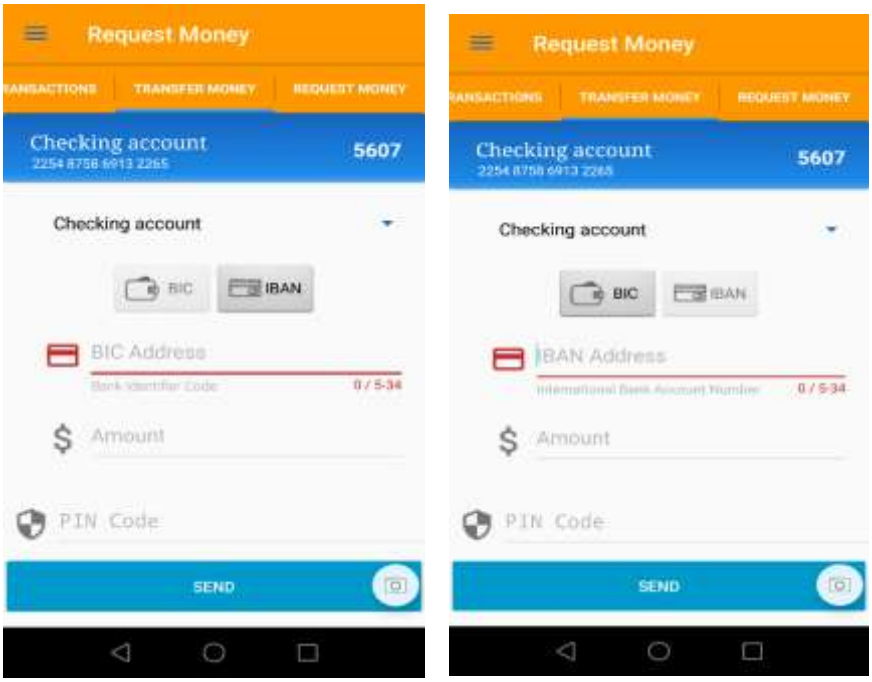

Figure 5.12 : Interface de transfert d'argent

- **La demande d'un transfert pour ce compte**

Pour faciliter la tâche de transfert le client peut générer un code QR, il suffit juste de saisir le montant demandé et le code va être stocké et prêt à être partagé tout comme illustre la figure 5.13.

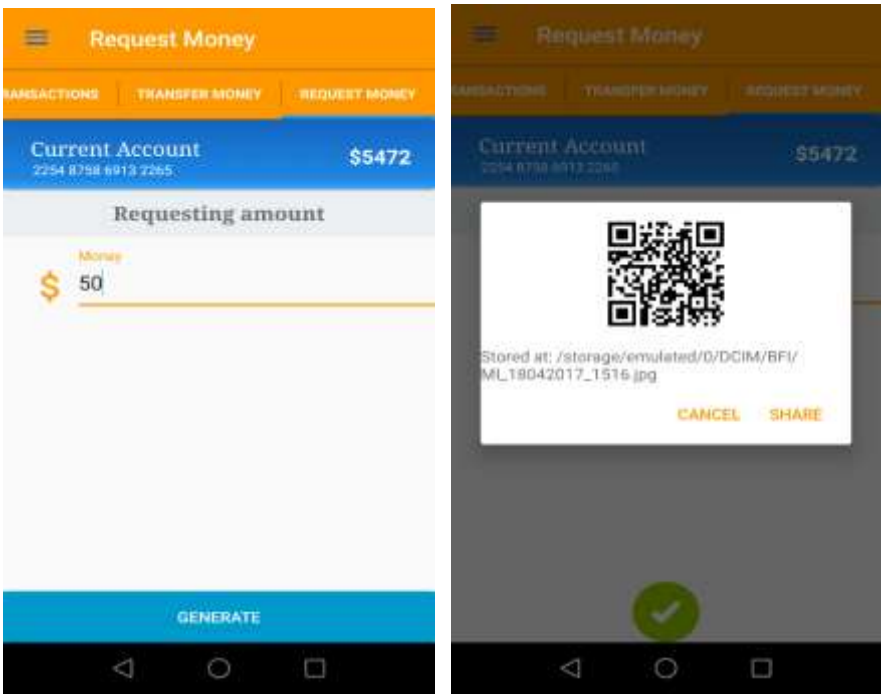

Figure 5.13 : Interface de demande de transfert

- **La demande des cartes et de carnet de chèque**

L'interface présentée sur la figure 5.14 est proposée pour les demandes. Le client doit tout d'abord séléctionner le type de la demande ensuite le modèle desiré puis il clique sur le bouton de demande.

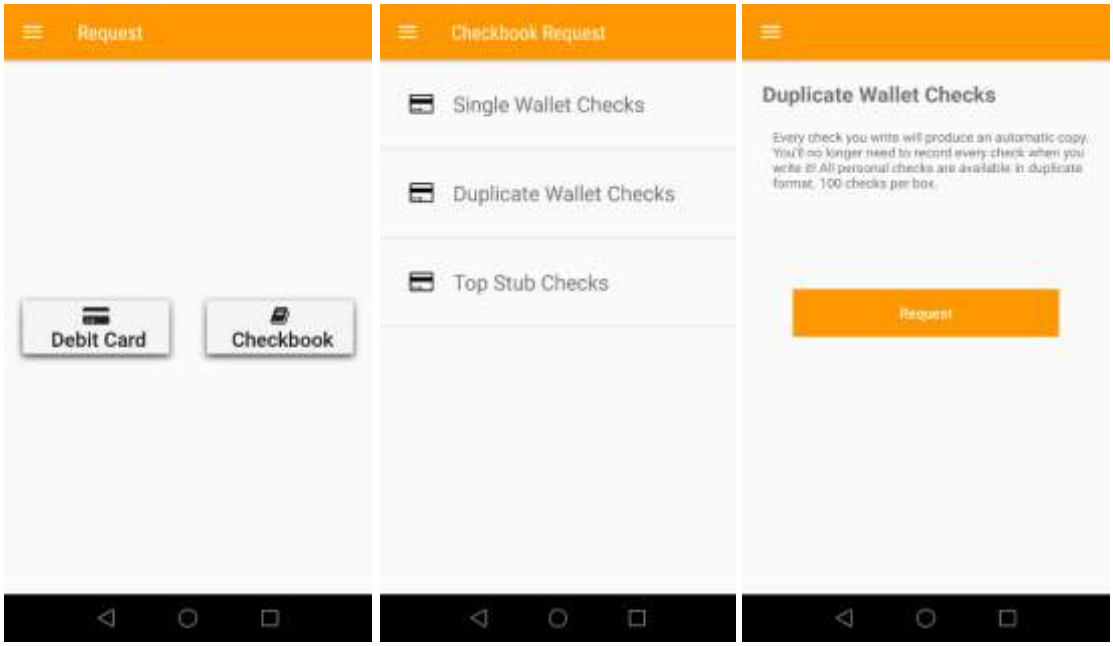

Figure 5.14 : Interface des demandes

- **Sprint 2 :** Gestion des services

Afin de conforter le client, nous avons mis à sa disposition un menu où il peut trouver tous les services offerts par l'application comme nous montre la figure 5.15. Et il peut encore choisir le compte qu'il veut utiliser pour certaines fonctionnalités qui nécessitent l'adhérence à un compte comme sur la figure 5.16.

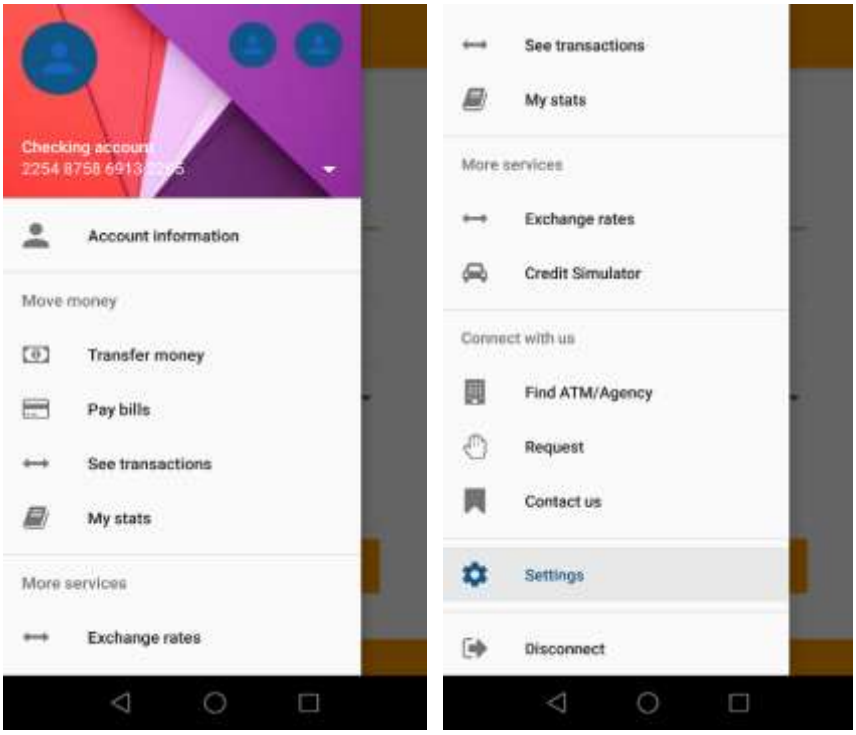

Figure 5.15 : Menu de l'application

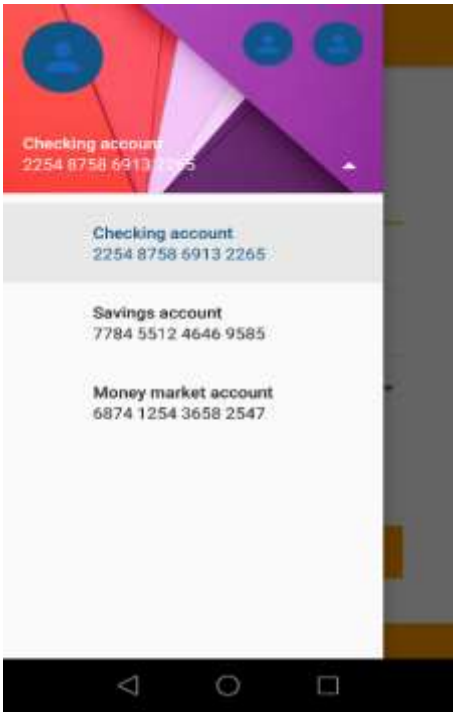

Figure 5.16 : Menu des comptes

- **Visualiser et convertir de la devise**

Comme illustre la figure 5.17, l'utilisateur peut visualiser l'échange de devises et simuler la conversion d'une somme en choisissant les devises désirées.

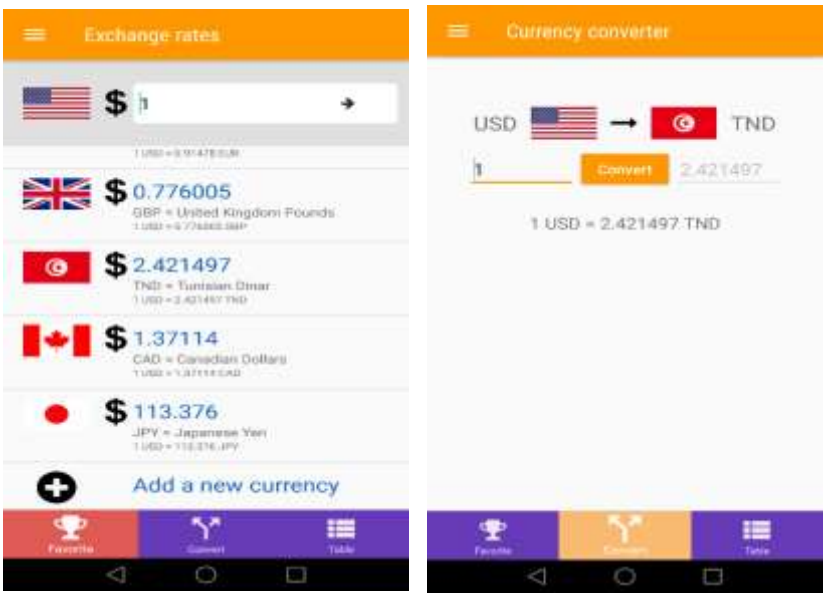

Figure 5.17 : Interface de l'échange de devises

- **Simuler des crédits**

Pour simuler des crédits, le client doit juste choisir la somme et nombre de mois, et le système va générer la somme à payer chaque mois, comme le montre la figure 5.18.

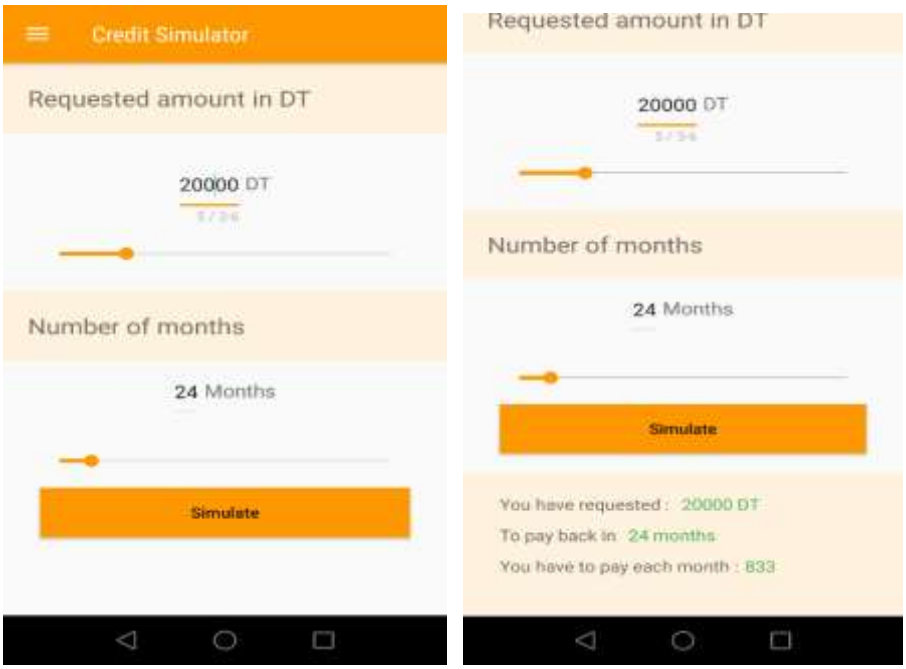

Figure 5.18 : Interface de simulation de crédit

- **Contacter la banque**

Afin de contacter la banque, le client doit remplir le formulaire mis à sa disposition et cliquer sur le bouton d'envoi. La réponse va être transmise par email. Le formulaire est sur la figure 5.19.

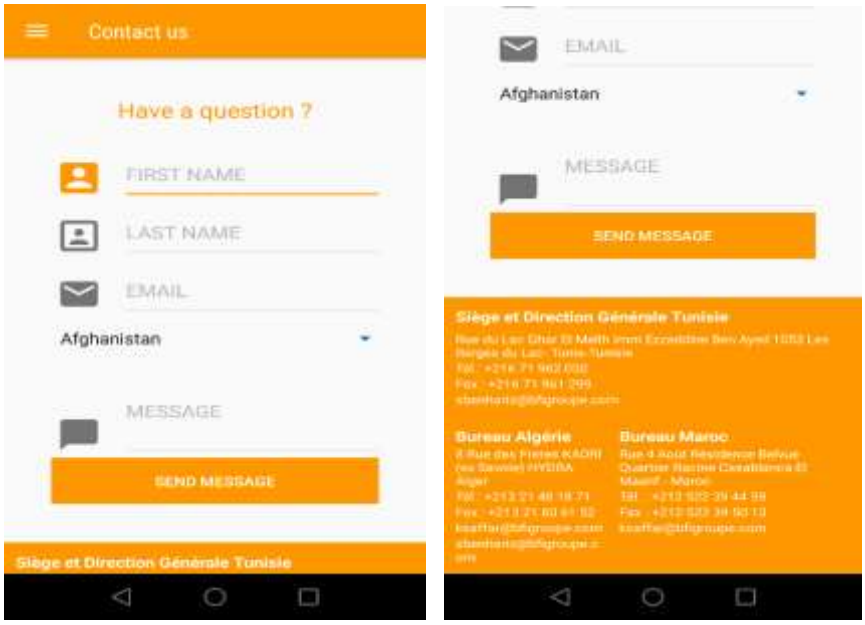

Figure 5.19 : Interface de contact

- **Localiser les agences et les distributeurs**

La figure 5.20 montre l'interface qui facilite la tâche de trouver une agence ou un distributeur proche du client.

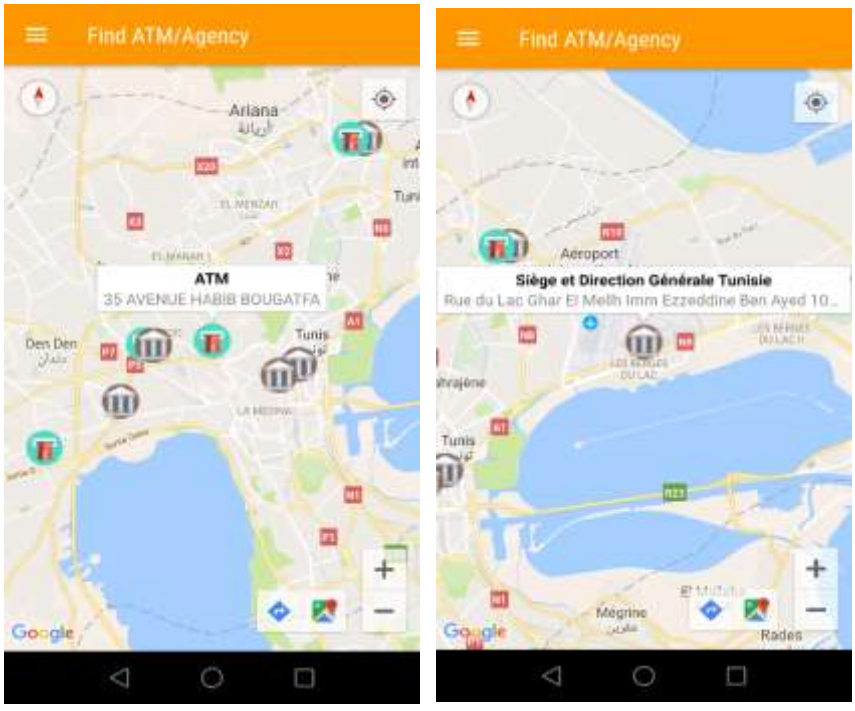

Figure 5.20 : Interface de localisation

5. Conclusion

Dans ce chapitre nous avons présenté les environnements du travail pour notre projet puis nous avons décrit les différentes interfaces offertes par nos applications.

Conclusion générale

L'objectif de notre projet était de développer une solution bancaire mobile qui propose à la fois une application mobile aux clients d'une banque ainsi qu'une application web pour l'administration du système.

Avant d'entamer ce projet, nous n'avions pas d'idée sur le développement d'application mobile et encore moins sur le développement d'une application Android. Au début on a passé un peu de temps pour apprendre comment concevoir les services web et comment les intégrer avec l'application Android, mais après, tout s'est bien passé surtout étant donné qu'on a trouvé le développement d'application Android très accessible.

Nous avons réussi à atteindre tous les objectifs principaux et nous avons ajouté un grand nombre de fonctionnalités secondaire comme la conversion de devises etc. Nous avons aussi intégré dans les applications mobiles et web des bibliothèques externes, et cela demande un temps d'apprentissage pour chaque bibliothèque.

Ce projet de fin d'étude a été une bonne opportunité pour apprendre des nouvelles technologies. Nous nous sommes initiés bien évidement au développement Android mais aussi à la plateforme J2EE pour concevoir l'application Web. Nous avons appris aussi à concevoir et à consommer des services web REST.

Cette expérience nous a permis de nous familiariser à la vie professionnelle et au travail d'équipe. Nous souhaitons finalement que l'entreprise d'accueil « BFI » adopte cette solution.

Webographie

- [1] http://www.bfigroupe.com/Fr/presentation-de-bfi_11_5
- [2] http://www.thierry-pigot.fr/scrum-en-moins-de-10-minutes/
- [3] https://fr.wikipedia.org/wiki/Syst%C3%A8me_d%27exploitation_mobile
- [4] https://www.taktilcommunication.com/blog/applications-mobile/definition-typologie-applications-mobiles.html
- [5] https://fr.wikipedia.org/wiki/Application_mobile
- [6] https://play.google.com/
- [7] https://fr.wikipedia.org/wiki
- [8] https://github.com/mikepenz

تلخيص : الهدف من هذا المشروع هو تحقيق تطبيق بنكي عبر الهاتف المحمول على نظام أندرويد، وتطبيق ويب لمحاكاة الجزء الإداري.

تطبيق الهاتف المحمول يسمح للمستخدمين التصرف فى حساباتهم البنكية، وتنفيذ عمليات مثل تحويل الأموال، والتحقق من تاريخ المعاملة، والتعرف على أماكن الوكالات البنكية وأجهزة الصراف الآلي والعديد من الميزات الأخرى.

يسمح تطبيق ويب المسؤول لتلبية احتياجات المستخدمين ويعطبه السيطرة على حسابات المستخدمين.

كلمات مفاتيح: أندرويد, J2EE, واب, راست, تطبيق مصرفي, Web service

Résumé : L'objectif de ce projet est de réaliser une application bancaire mobile sous la plateforme Android ainsi qu'une application web pour simuler la partie administrative.

L'application mobile permet aux utilisateurs de consulter leurs comptes bancaires en temps réel et d'effectuer des opérations comme le transfert d'argent, consulter l'historique des transactions, géolocaliser les agences et les distributeurs de billet et plein d'autres fonctionnalités.

L'application web permet à l'administrateur de répondre au besoin des utilisateurs et lui donne le contrôle sur les comptes des utilisateurs.

Mots clès : Android, J2EE, Web, REST, Application bancaire, Service Web

Abstract : The objective of this project is to realize a mobile banking application under Android platform as well as a web application to simulate the administrative part.

The mobile application allows users to view their bank accounts in real time and perform transactions such as money transfer, view transaction history, locate agencies and ATMs, and many other features.

The web application allows the administrator to respond to user's requests and gives them control over their accounts.

Keywords : Android, J2EE, Web, REST, Banking application, Web service