***Remerciements***

Je tiens, tout d’abord, à remercier Dieu le tout puissant et miséricordieux, qui nous a donné la force et la patience d’accomplir ce modeste travail.

Je dédié ce modeste travail et ma profonde gratitude à mon père et ma mère pour leurs patiences et encouragements.

Au terme de mon projet de fin d’études, qu’il me soit permis d’exprimer mes sentiments de gratitude aux différentes personnes qui m’ont encadrés et soutenues tout au long de notre travail rendant ainsi possible l’accomplissement du présent projet.

Je tiens tout particulièrement à exprimer ma profonde gratitude et respect à Monsieur **BASSI Selim**, de m’avoir offert l’opportunité d’effectuer mon stage au sein de BFI.

Notre profonde gratitude s’adresse tout particulièrement à Monsieur **JEMAI Abderrazak**, qui en dépit de ses multiples occupations et sollicitations, a accepté de diriger, de lire et d’apporter des correctifs à ce travail sans oublier ses conseils et suggestions éclairées.

Mes remerciements s’adressent aussi à toutes l'équipe du département informatique de BFI qui m’a accompagné et aidé d’une manière et d’une autre à la réalisation du présent travail.

***Table des matières***

***Introduction Générale***

Pressés par la modernisation croissante, par l'invasion des nouvelles technologies et par l'émergence des nouveaux acteurs sur le marché des services financiers, les banques cherchent à adapter des solutions non seulement pour asseoir leurs spécificités, mais aussi pour se bâtir une identité propre.

En fait, et afin de fidéliser ses clients et attirer une clientèle potentielle, la banque est amenée à approcher différemment ses clients en se différenciant, soit dans ses moyens de traitement de l'information et de la communication bancaire, soit dans l'exécution des différentes fonctions liées directement à la banque. On parle ainsi de la révolution technologique en matière d'information et de communication qui a permis de nouvelles formes de distribution de services bancaires telle que la banque électronique.

Cette entité est reconnue la plupart du temps par son appellation commercial d'origine anglais: « E- Banking». D'ailleurs, ce terme englobe le processus par lequel un consommateur peut gérer ses transactions bancaires électroniquement sans être obligé de visiter une succursale physique.

En outre, l'adoption de la banque électronique devient une nécessité pour la banque qui souhaite maintenir sa part de marché, ainsi que retenir ses clients. Ce concept est entré dans les mœurs de plusieurs institutions financières pour le traitement de leurs opérations courantes, dans la mesure où il répond aux attentes des utilisateurs, notamment les clients, les fournisseurs et les partenaires institutionnelles.

Notons toutes fois que l'E-Banking du point de vue de ses acteurs tel que la banque, les clients, n'est ni une technologie, ni une finalité en soi mais bien une pratique d'affaire et plus précisément une pratique d'affaire innovante.

Dans ce contexte, ce projet de fin d’études a pour but de concevoir et développer une application d’E-Banking comprenant 2 parties :

* Une application web du système d’informations (SI) qui utilise les services web pour transférer les données stockées par la banque (client, compte, …).
* Une application Web Persistante qui consiste d’une interface web Abonne, pour les utilisateurs, qui sert à consulter les données financières et gérer les virements et les opérations et d’une interface web Administrateur, pour les employés de la banque, qui sert à manipuler les abonnées et leurs virements/bénéficiaires.

Notre travail s’intitulant « Développement d’un projet E-Banking en JEE » est composé de quatre chapitres :

* Dans le premier chapitre, on présente la revue de la littérature : l’analyse théorique et empirique du problème étudié.
* Le deuxième chapitre présentera la planification.
* Le troisième chapitre portera sur les étapes de conception de l’application.
* Et le dernier chapitre sera dédié à la partie déploiement et codage de ce projet où on va détailler l'architecture des deux applications et le schéma de la base de données.

On finira par une conclusion générale ainsi que la proposition de quelques perspectives.

***Chapitre 1 : Revue de la littérature***

1. Introduction :

Dans ce chapitre, nous abordons l’environnement du stage en présentant l’entreprise d’accueil BFI. Puis, nous présentons la problématique et l’analyse de l’existant tel que la solution existante et ses critiques. Nous clôturons ce chapitre par la description du plan de notre projet en précisant les spécifications fonctionnelles et non- fonctionnelles.

1. Présentation de l’organisme d’accueil :

**BFI** est spécialisée depuis 1994 dans l’édition et l’[intégration de solutions logicielles](http://www.bfigroupe.com/Fr/implementation-integration_11_88) destinées aux banques et institutions financières. Bien implantée sur le marché africain, **BFI** a réussi à se forger une réputation de très haut niveau, en équipant plus de 200 clients dans 22 pays différents notamment avec des institutions de renommée mondiale.

Forte d’une expérience de plus de 20 ans d'installations réussis en Afriqueet d’une politique de proximité auprès de ses clients, **BFI**offre un savoir faire reconnu dans la mise en œuvre de projets couvrant: le suivi et le pilotage, l’intégration de systèmes, la méthodologie de test et de recette, la migration des données, [la formation](http://www.bfigroupe.com/Fr/formation_11_65) et le transfert de compétences, [le conseil](http://www.bfigroupe.com/Fr/conseil_11_12) et[l’accompagnement](http://www.bfigroupe.com/Fr/accompagnement_11_13) et enfin le [support et la maintenance](http://www.bfigroupe.com/Fr/support-maintenance_11_90).

**BFI** a construit son offre autour de trois axes essentiels du domaine de la banque et de la finance : le Global Banking urbanisé couvrant l'ensemble des métiers de la banque, les systèmes de place et les activités de marchés.

Ainsi, **BFI** offre à ses clients une gamme de solutions couvrant la globalité des besoins d’une banque en matière de systèmes d’information front et back-office ainsi que tous les métiers de la banque : gestion des clients, gestion des risques, contrôle des engagements, des crédits, trésorerie, marché des capitaux, moyens de paiement, comptabilité… leur permettant ainsi de disposer d'un système d'information à la hauteur des enjeux stratégiques. [1]

1. Problématique :

A l'heure actuelle, les entreprises ont besoin de suivre de près les changements sur leurs systèmes d'information pour pouvoir répondre aux exigences des différents utilisateurs. Ce besoin est apparu dès qu'il y a eu une prise de conscience de la situation des entreprises à cause de la mauvaise administration de la gestion des changements.

Cela a engendré de graves conséquences, entre autres, la non-conformité aux termes et aux conditions prédéfinies ainsi que le non-respect des délais. C'est dans ce cadre que BFI a constaté qu'il est urgent de développer une application de gestion des changements pour répondre à son propre besoin de la gestion de nouveaux projets ou de la modification des applications déjà réalisées.

Pour cette raison, nous avons intégré le département informatique au sein de BFI pour accomplir cette mission. Notre projet consiste alors à réaliser une application permettant à la banque en question de contrôler, de communiquer et de répondre plus efficacement à différentes opérations par les utilisateurs.

1. Analyse de l’existant :

BFI allie la maîtrise des nouvelles technologies de l’information à la connaissance du métier de ses clients. Ses équipes pluridisciplinaires maîtrisent un large éventail de compétences, toutes au service d’un même objectif : optimiser la croissance de ses clients.

1. Solution Existante :

**Carthago** est une solution logicielle globale bancaire couvrant l’ensemble des métiers de la banque de détail notamment la [gestion des clients](http://www.bfigroupe.com/Fr/noyau_56_24_9_D4), des [dépôts à vue et de l’épargne](http://www.bfigroupe.com/Fr/depots-et-placements_56_25_9_D5), des [opérations de guichet](http://www.bfigroupe.com/Fr/depots-et-placements_56_25_9_D5), des [engagements](http://www.bfigroupe.com/Fr/engagements_56_28_9) et du [contrôle des risques](http://www.bfigroupe.com/Fr/controle-et-comptabilite-generale_56_34_9_D27), des [opérations de financement du commerce international](http://www.bfigroupe.com/Fr/bancaire-etranger_56_32_9), de la [comptabilité](http://www.bfigroupe.com/Fr/controle-et-comptabilite-generale_56_34_9_D24) et du [contrôle budgétaire](http://www.bfigroupe.com/Fr/controle-et-comptabilite-generale_56_34_9_D26). La solution Carthago est multi-établissements, multi-agences, multi-langues, multidevises, multi-plans de comptes et multicanaux de distribution. [2]

Les solutions Carthago sont développées autour de notre Framework transactionnel garantissant évolutivité et indépendance. Architecture fonctionnelle modulaire basée sur un socle technique conforme à l’état de l’art des technologies JEE, garantissant interopérabilité, évolutivité, pérennité et montée en charge. Facilité de déploiement assurée grâce à une couche présentation totalement WEB (client léger) et RIA (Rich Internet Application). Cycle de développement basé sur un processus itératif conforme au modèle UP (Unified Process) permettant une industrialisation du développement via le Framework et ainsi une importante réactivité à de faibles coûts, multi-canal en natif. [3]

Carthago est modulaire et personnalisable assurant une évolutivité aisée et offrant à chaque banque la liberté de :

* Choisir son environnement technique et de réutiliser son existant (liberté dans le choix des systèmes d’exploitation, des serveurs d’application, et des serveurs de bases de données),
* Conserver et d’intégrer les applications de son choix,
* Personnaliser la solution par des paramétrages spécifiques,
* Intégrer les métiers au rythme choisi sans nécessairement une remise en cause brutale de l’existant. [2]

1. Critiques :

* L’accessibilité est la disponibilité et le caractère utilisable d’une ressource pour des groupes spécifiques.
* Les risques opérationnels et de sécurité sont liés à des insuffisances dans la fiabilité ou l’intégrité du système. Les considérations de sécurité jouent un rôle prépondérant car les systèmes d’information peuvent faire l’objet d’attaques autant internes qu’externes.

1. La solution proposée:

Le travail à faire est de concevoir et développer une application d’E-Banking que nous avons nommée « Développement d’un projet E-Banking en JEE ». Une application Spring MVC qui sera installée sur un terminal web développé en langage de programmation JAVA dédiée à la fois aux clients de la banque et aux administrateurs.

1. Les spécifications fonctionnelles :

D’une part, l’application mobile permet aux clients de la banque de:

* S’authentifier : Pour bénéficier des fonctionnalités offertes de l’application, le client de la banque doit s'authentifier en saisissant son code abonné et son mot de passe.
* Accéder aux services clients :
  + Consulter les comptes et leurs opérations
  + Vérifier les bénéficiaires valides.
  + Demander et/ou consulter un chéquier.
  + Demander et/ou consulter une carte bancaire.
  + Effectuer des virements sécurisés simples intra-bancaires.
  + Effectuer des virements sécurisés simples inter-bancaires et la vérification du code de validation par e-mail.
* Accéder aux données personnelles :
  + Changer le mot de passe.
  + Mettre à jour les coordonnées du client dans la base de données de la banque.

D’autre part, notre application s'adresse aux administrateurs qui sont chargés de la supervision des abonnés et leurs virements et assurer le bon fonctionnement de l’application:

* S’authentifier : Pour bénéficier des fonctionnalités offertes de l’application, l’administrateur doit s'authentifier en saisissant son nom utilisateur et son mot de passe.
* Accéder aux services administrateurs :
  + Consulter la liste des abonnés.
  + Ajouter, manipuler et supprimer des abonnés.
  + Consulter les bénéficiaires de chaque abonné.
  + Ajouter, manipuler et supprimer des bénéficiaires.
  + Consulter les virements intrabancaires et interbancaires.
  + Valider ou invalider les virements bancaires.
* Accéder aux données personnelles :
  + Changer le mot de passe.

1. Spécifications non-fonctionnelles :

Outre que les besoins fonctionnels cités précédemment, pour bien répondre aux exigences des utilisateurs, le système devra pouvoir assurer les besoins non fonctionnels suivants :

* L’ergonomie et la convivialité :
  + L'application fournira une interface conviviale, simple et homogène à utiliser et qui ne requiert aucun pré requis, donc elle pourra être exploitable par tout type d'utilisateurs (même les non informaticiens).
  + L’utilisateur doit être guidé lors de la saisie de certaines informations, afin de respecter les formats des champs de notre base de données, dans le cas d’une faute, l’application affiche un message d’erreur.
  + La navigation entre les interfaces de notre future application web doit être légère et fluide.
* La sécurité : L'accès aux informations n'est possible qu'après vérification des privilèges et des droits d'accès. Ainsi tout utilisateur passera par une phase d'authentification pour pouvoir consulter les services offerts par l'application : (nom utilisateur + mot de passe numérique) récupérés lors de l’ajout d’un nouveau abonné. Les abonnés et les administrateurs ne peuvent pas accéder qu’aux leurs options relatives.
* L'extensibilité : L'architecture de l'application permettra l'évolution et la maintenance (ajout ou suppression ou mise à jour) au niveau de ses différents modules d'une manière flexible.
* Contraintes humaines : Notre solution doit tenir compte des besoins de chaque acteur participant dans la gestion des changements. Elle doit leur faciliter la tâche et assurer leur activité dans les bonnes conditions. Donc, notre application doit être facile à utiliser en présentant des interfaces ergonomiques.
* Contraintes techniques :
  + Toutes les applications web nécessitent une authentification par certificat qui sera établit en mettant en place le Framework Spring Security. Pour notre cas, on a dû suivre le protocole interne et intégrer ce Framework d’authentification et de contrôle d’accès.
  + Lors d’un virement bancaire le client est invité à rentrer son mot de passe pour vérifier la transaction. Dans le cas d’un virement interbancaire, un code de validation est envoyé à son e-mail.
  + Les mots de passes des abonnés doivent êtres cryptés « BCrypt » au niveau de la base de données afin de garder sécurisé l’accès à l’application.
  + La saisie du mot de passe doit être masquée et seulement possible par une pad numérique pour éviter le vol des mots de passe.
  + Les requêtes doivent être optimisées afin d’assurer un temps de réponse minimal.

1. Conclusion :

Dans ce premier chapitre, nous avons commencé par présenter notre organisme d'accueil BFI. Ensuite, nous avons indique la solution existante et ses déficiences. Enfin, après avoir décrire le contexte du projet et la solution proposée qui constituera le sujet de notre projet en précisant les spécifications fonctionnelles et non-fonctionnelles, nous nous dirigerons naturellement vers la planification de l’application.

***Chapitre 2 : Planification***

1. Introduction :

Dans ce chapitre, le planisme sera présenté. Il s’agit en fait d’une phase de planification et architecture qui présente les travaux réalisés dans cette période conduits à construire une bonne vision du produit: préparer l’environnement de développement et indiquer les outils de conception/développement.

1. Technologies et outils de développements :
   1. Outils de conception :

* Enterprise Architect :

Enterprise Architect est un logiciel de [modélisation](https://fr.wikipedia.org/wiki/Mod%C3%A9lisation_informatique) et de [conception](https://fr.wikipedia.org/wiki/Conception_de_logiciel) [UML](https://fr.wikipedia.org/wiki/Unified_Modeling_Language), édité par la société australienne « [Sparx Systems](https://fr.wikipedia.org/w/index.php?title=Sparx_Systems&action=edit&redlink=1) ». Couvrant, par ses fonctionnalités, l'ensemble des étapes du cycle de conception d'application, il est l'un des logiciels de conception et de modélisation les plus reconnus. [4]

* 1. Language de modélisation :

UML (en anglais Unified Modeling Language ou « langage de modélisation unifié ») est un langage de modélisation graphique à base de représentations graphiques. Il est apparu dans le monde du génie logiciel, dans le cadre de la «conception orientée objet». Couramment utilisé dans les projets logiciels, il peut être appliqué à toutes sortes de systèmes ne se limitant pas au domaine informatique. [5]

* 1. Environnement technique du travail :

Le tableau 3 détaille la configuration matérielle des machines de développement et de déploiement utilisées ainsi que les différents logiciels qui y sont installés :

|  |  |
| --- | --- |
| Machine | Machine1 |
| Propriétaire | Missaoui Sadok |
| Processeur | Intel core i7 |
| RAM | 8Go |
| Système d’exploitation | Windows 8.1 Entreprise (x64) |
| Logiciels installées | Eclipse - Apache Tomcat Server  Enterprise architect - Maven  WampServer – MySQL - JDK |

Tableau 3 : Environnement technique du travail

* 1. Frameworks de développement :

Tout système d'information nécessite la réalisation de trois groupes de fonctions: le stockage des données, la logique applicative et la présentation (MVC). Ces trois parties sont indépendantes les unes des autres. D'où notre choix de l'architecture 3 tiers pour l'application Web (Figure 1).

L’architecture 3-tiers, est une architecture partagé entre :

* Les pc clients (ou administrateurs)
* Le serveur Web et le serveur d’application.
* Le serveur de base de données.

En effet, cette architecture est un modèle logique d’architecture applicative qui vise à séparer, très nettement trois couches logicielles au sein d’une même application ou système, à modéliser et à présenter cette application comme un empilement de trois couches, étages, ou niveaux qui sont les suivantes :

* Couche Présentation (premier niveau) :

Elle correspond à la partie de l’application visible et interactive avec les utilisateurs. On parle d’interface homme machine. La couche présentation relaie les requêtes de l’utilisateur à destination de la couche métier, et lui présente en retour les informations renvoyées par les traitements de cette couche. Il s’agit donc d’un assemblage de services métiers et applicatifs offerts par la couche inférieure.

* Couche Métier / Business (second niveau) :

Elle correspond à la partie fonctionnelle de l’application, celle qui implémente la «logique», et qui décrit les opérations que l’application opère sur les données en fonction des requêtes des utilisateurs effectuées au travers de la couche présentation.

* Couche Accès aux données (troisième niveau) :

C'est la partie qui gère l’accès aux données du système. Ces données peuvent être propres au système, ou gérées par un autre système.

La figure 1 schématise l’architecture générale de note application web.



Figure 1: Architecture générale d'application web [6]

Parlons maintenant de la mise en place de la plate-forme J2EE : Cette opération se constitue de deux étapes fondamentales : la mise en place des couches et celle du serveur web.

* Mise en place des couches

Nous avons choisi de coder l’application en se basant sur les couches Hibernate, Spring et Spring Security afin de faciliter l’effort de l’implémentation et de favoriser l’aspect sécurité.

* La couche Spring

Spring est un Framework libre pour construire et définir l’infrastructure d’une application java, dont il facilite le développement et les tests. «SPRING est effectivement un conteneur dit «léger» .Il prend donc en charge la création d'objets et la mise en relation d'objets par l'intermédiaire d'un fichier de configuration qui décrit les objets à fabriquer et les relations de dépendances entre ces objets.» [7]

Pour l'authentification on a du intégrer le Framework Spring Security. En effet C’est est un sous-projet de Spring, il a été lancé en 2003 sous le nom d’Acegi Security. En 2007 il sera renommé Spring Security. C’est l’un des projets les plus avancés de Spring.

«Spring Security est un cadre qui se concentre sur la fourniture de l'authentification et l’autorisation aux applications Java. Comme tous les projets de Spring, la puissance réelle de Spring Security se trouve dans la façon dont facilement il peut être étendu pour répondre aux besoins personnalisés». [8]

* La couche Hibernate

C’est une couche qui sert à la persistance des objets dans les bases de données relationnelles. Elle présente un avantage remarquable contre l’accès classique aux données manifestés par les JDBCs «Java Data Base Connectivity», vue la rapidité de traitement via des appels à des méthodes objet de haut niveau.

* Mise en place du serveur Web : Apache Tomcat

«Apache Tomcat est un serveur d’application Java permettant d’exécuter des servlets et des pages serveurs Java(JSP). Il peut être utilisé ou couplé avec un serveur Web, et porté sur n’importe quel système sur lequel une machine virtuelle Java est installée». [9] Il est paramétrable par des fichiers XML et de propriétés, et inclut des outils pour la configuration et la gestion. Il comporte également un serveur HTTP.

* 1. Outils de développements :
* Environnement de développement : Eclipse Oxygen 4.6 (La nouvelle version de l'IDE)

Eclipse est un environnement de développement open source, décliné et organisé en un ensemble de sous-projets de développements logiciels, de la Fondation Eclipse visant à développer un environnement de production de logiciels libre qui soit extensible et universel en s’appuyant principalement sur le langage Java.[10]

* L'ensemble des bibliothèques logicielles : Java Développent Kit (JDK7 64bits)

Le JDK désigne un ensemble de bibliothèques logicielles de base du langage de programmation Java, ainsi que les outils avec lesquels le code Java peut être compilé, transformé en byte code destiné à la JVM (Java Virtual Machine). [11]

* WampServer 64 bits

WampServer est une plate-forme de développement Web sous Windows pour des applications Web dynamiques à l’aide du serveur Apache2, du langage de scripts PHP et d’une base de données MySQL. Il possède également PHPMyAdmin pour gérer plus facilement vos bases de données. [12]

* MySQL

MySQL est un système de gestion de bases de données relationnelles (SGBDR). Il est distribué sous une double licence GPL et propriétaire. Il fait partie des logiciels de gestion de base de données les plus utilisés au monde3, autant par le grand public (applications web principalement) que par des professionnels, en concurrence avec Oracle, Informix et Microsoft SQL Server. [13]

* Maven

Couramment appelé Maven, Apache Maven est un outil de [gestion et d'automatisation de production des projets logiciels](https://fr.wikipedia.org/wiki/Moteur_de_production) [Java](https://fr.wikipedia.org/wiki/Java_(langage)) en général et [Java EE](https://fr.wikipedia.org/wiki/Java_EE) en particulier. Maven est géré par l'organisation [Apache Software Foundation](https://fr.wikipedia.org/wiki/Apache_Software_Foundation). L'outil était précédemment une branche de l'organisation [Jakarta Project](https://fr.wikipedia.org/wiki/Apache_Jakarta). L'objectif recherché est de produire un logiciel à partir de ses sources, en optimisant les tâches réalisées à cette fin et en garantissant le bon ordre de fabrication. [14]

* 1. Language de programmation :
* Java

Java est un langage de programmation orienté objet, développé par Sun Microsystems permettant de créer des logiciels compatibles avec des nombreux systèmes d’exploitation et donne aussi la possibilité de développer des programmes pour téléphones portables. [15]

* AngularJS

AngularJS est une Framework qui facilite la création d'applications sur le Web. Angular combine des modèles déclaratifs, des injections de dépendances, des outils de bout en bout et des meilleures pratiques intégrées pour résoudre les problèmes de développement. [16]

* REST

REST (Representational State Transfer) est un style d'architecture réseau pour Web Services qui met l'accent sur la définition de ressources identifiées par des URI, et utilise les messages du protocole HTTP pour définir la sémantique de la communication client/serveur: [17]

* GET pour le rapatriement d'une ressource,
* POST pour une création
* PUT pour mise à jour
* DELETE pour suppression

REST n'est pas vraiment un standard. Il n'existe donc pas de spécifications de REST. Il faut comprendre le style REST et ensuite concevoir des applications ou des services Web selon ce style.

Bien que REST ne soit pas un standard, il utilise des standards. En particulier :

* URI comme syntaxe universelle pour adresser les ressources
* HTTP un protocole sans état (stateless) avec un nombre très limité d'opérations,
* Des liens hypermedia dans des documents HTML et XML pour représenter à la fois le contenu des informations et la transition entre états de l'application,
* Les types MIME comme text/xml, text/html, image/jpeg, application/pdf, video/mpeg pour la représentation des ressources. [18]
* JSON

JSON (JavaScript Object Notation) un format d'encodage de données structurées en JavaScript qui date de 2002. Il permet de représenter de l’information structurée. Il n'est pas rare d'associer le couple REST/JSON pour des raisons de simplicité. Son avantage est de fournir un support pour une écriture simple et légère au format texte, relativement compréhensible par les développeurs JavaScript, mais aussi - et surtout - d'être nativement interprété. [19]

1. Conclusion :

Au long de ce chapitre, nous avons présenté la planification, est en fait une phase de préparation pendant laquelle nous avons défini les technologies et les environnements de développement ainsi que l’architecture globale de l’application.

***Chapitre 3 : Conception***

1. Introduction :

Dans ce chapitre, nous présentons la conception de l’application. C’est une étape primaire et essentielle dédiée à la mise au point et à l'écriture d'un projet. Nous allons alors identifier les rôles des utilisateurs, indiquer les cas d’utilisation, déterminer un backlog initial, formuler les classes en précisant l’architecture MVC et préparer la démarche de l’emploi de l’application.

1. Capture des besoins:

Nous procéderons par l’identification des acteurs puis la description des cas d’utilisation de la future application.

* 1. Identification des acteurs :

Les deux acteurs principaux sont :

* Abonne
* Administrateur

En tenant compte des fonctionnalités offertes en commun pour les deux acteurs, il existe un autre plus général « Utilisateur » avec lequel ils constituent une relation d’héritage comme il est présenté dans la figure 2.

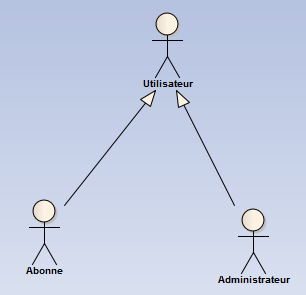


Figure 2 : Héritage entre les acteurs

* 1. Description des acteurs :

Le tableau 1 définit les acteurs de l’application ainsi que leurs rôles.

|  |  |  |
| --- | --- | --- |
| Acteur | Définition | Rôle |
| Utilisateur | Toute personne utilisant l’application E-Banking. | Il a un accès à un espace publique (l’index) ou il pourra accéder aux d’autres fonctionnalités de site sauf celles d’E-Banking. Il peut s’authentifier pour accéder à ces fonctionnalités. |
| Abonne | Tout client de la banque possédant un login et son mot de passe afin d’utiliser l’application. | Il s’authentifie afin qu’il puisse accéder à son espace client où il bénéficiera d’un ensemble de fonctionnalités (consultation comptes, demande carte ou chéquier, virements ...). Il peut aussi accéder à son profile afin de le mettre à jour. |
| Administrateur | Il s’agit de la personne surveillant les abonnées et leurs opérations. | Il est responsable de la gestion des abonnés E-Banking, la gestion des bénéficiaires et la mise à jour des demandes de virements (valider ou les invalider). |

Tableau 1 : Description détaillée des acteurs

* 1. Diagrammes de cas d’utilisation :

Un cas d'utilisation est la description d'un ensemble de séquences d'actions qu'un système effectue pour produire un résultat observable à un acteur. Un cas d'utilisation représente une exigence fonctionnelle de votre système dans son ensemble. Les diagrammes de cas d'utilisation décrivent ce qu'un système fait du point de vue d'un observateur externe. L'accent est mis sur ce qu'un système fait, plutôt que sur la façon dont il le fait.

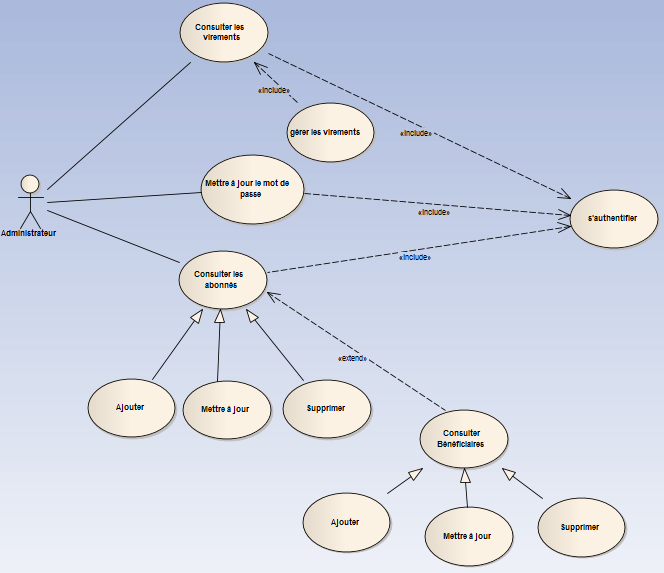


Figure 3 : Diagramme de cas d’utilisation de l’administrateur

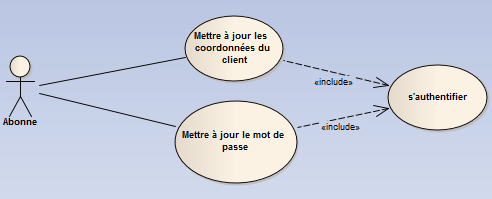


Figure 4 : Diagramme de cas d’utilisation de la gestion de données personnelles par l’abonné

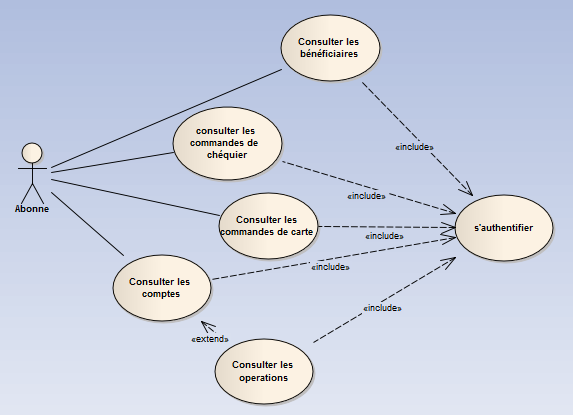


Figure 5 : Diagramme de cas d’utilisation de la consultation de données financières par l’abonné

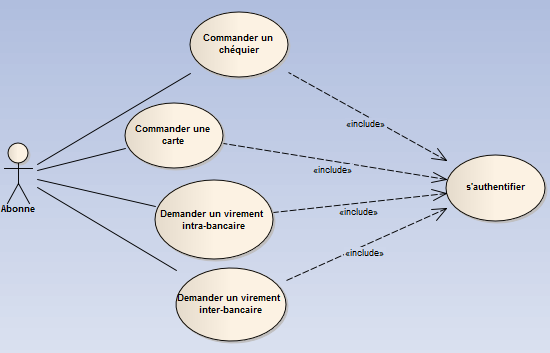


Figure 6 : Diagramme de cas d’utilisation de la gestion des opérations par l’abonné

1. Backlog du produit :

Pour chaque cas d’utilisation on identifie la priorité, Le risque et la description présenté dans le tableau 2.

* Priorité: Elle permet de classer les cas d’utilisation par ordre de priorité dans le bon fonctionnement de l’application.
* Estimation : Sert à estimer l’effort nécessaire pour implémenter une fonctionnalité.
* Description : Elle permet de décrire les cas d’utilisation de sorte à indiquer leurs rôles dans l’application.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Nom | Priorité | Estimation | Description |
| 1 | S’authentifier | Haute | Moyenne | En tant qu’un utilisateur de l’application E-Banking, je veux m’authentifier afin de bénéficier des fonctionnalités de l’application. |
| 2 | Commander un chéquier | Haute | Moyenne | En tant qu’abonné, je veux demander un nouveau chéquier afin de faciliter mes transactions quotidiennes. |
| 3 | Commander une carte | Haute | Moyenne | En tant qu’abonné, je veux demander une nouvelle carte afin de faciliter mes transactions quotidiennes. |
| 4 | Demander un virement intra-bancaire | Haute | Haute | En tant qu’abonné, je veux transférer de l’argent entre mes comptes domiciles au sein de la banque. |
| 5 | Demander un virement inter-bancaire | Haute | Haute | En tant qu’abonné, je veux transférer de l’argent entre mes comptes et mes bénéficiaires. |
| 6 | Consulter les comptes | Haute | Faible | En tant qu’abonné, je veux consulter mes comptes et leurs soldes. |
| 7 | Consulter les bénéficiaires | Haute | Faible | En tant qu’abonné, je veux consulter mes bénéficiaires valides et leurs comptes. |
| 8 | Consulter les opérations | Haute | Faible | En tant qu’abonné, je veux consulter mes opérations et de prendre un extrait en utilisant la date et la pagination. |
| 9 | Consulter les commandes de chéquier | Haute | Faible | En tant qu’abonné, je veux consulter l’état des commandes de chéquier. |
| 10 | Consulter les commandes de carte | Haute | Faible | En tant qu’abonné, je veux consulter l’état des commandes de carte. |
| 11 | Mettre à jour les coordonnées du client | Faible | Moyenne | En tant qu’abonné, je veux voir et/ou mettre à jour mes informations personnelles. |
| 12 | Mettre à jour le mot de passe de l’abonné | Haute | Moyenne | En tant qu’abonné, je veux changer mon mot de passe. |
| 13 | Consulter les abonnés | Haute | Faible | En tant qu’administrateur, je veux consulter la liste des abonnés. |
| 14 | Ajouter un abonné | Haute | Moyenne | En tant qu'administrateur, je veux ajouter un abonné. |
| 15 | Mettre à jour un abonné | Haute | Moyenne | En tant qu'administrateur, je veux mettre à jour les coordonnées des abonnées afin de leur faciliter l'accès. |
| 16 | Supprimer un abonné | Haute | Faible | En tant qu'administrateur, je veux supprimer un abonné. |
| 17 | Consulter les bénéficiaires | Haute | Faible | En tant qu’administrateur, je veux consulter la liste des bénéficiaires de chaque abonné. |
| 18 | Ajouter un bénéficiaire | Haute | Moyenne | En tant qu'administrateur, je veux ajouter un bénéficiaire à l’abonné. |
| 19 | Mettre à jour un bénéficiaire | Haute | Moyenne | En tant qu'administrateur, je veux mettre à jour les coordonnées des bénéficiaires (valides ou invalides, nom, …). |
| 20 | Supprimer un bénéficiaire | Haute | Faible | En tant qu'administrateur, je veux supprimer un bénéficiaire d’un abonné. |
| 21 | Consulter les virements | Moyenne | Faible | En tant qu'administrateur, je veux consulter la liste des virements. |
| 22 | Gérer les virements | Moyenne | Faible | En tant qu'administrateur, je veux gérer les virements en les validant/invalidant. |
| 23 | Mettre à jour le mot de passe de l’administrateur | Haute | Moyenne | En tant qu’administrateur, je veux changer mon mot de passe. |

Tableau 2 : Backlog du produit

1. Phase d’analyse (diagramme de classes d’analyse):

La phase d'analyse a comme but l'élaboration du diagramme de classe de domaine. Le diagramme de domaine est la modélisation des entités et concepts présentes dans le domaine de l'application. Il s'agit dans ce diagramme de classes métiers qui ne contiennent que des données.

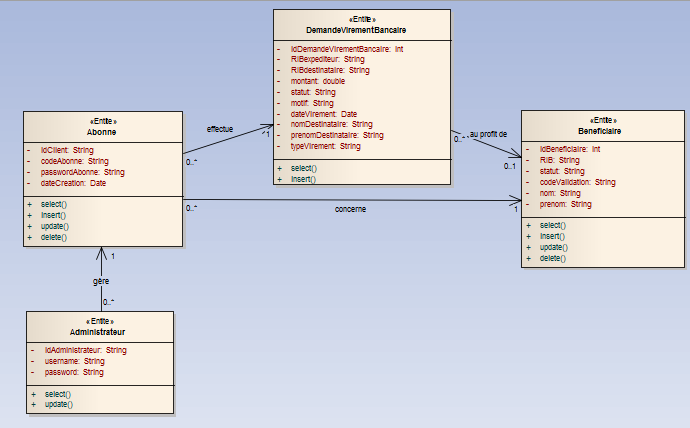


Figure 7 : Diagramme de classes de domaine Persistantes

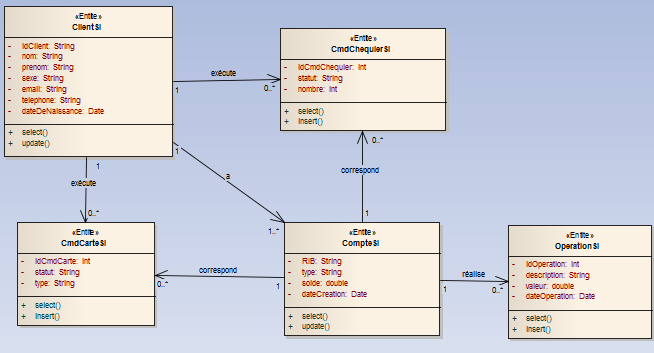


Figure 8 : Diagramme de classes de domaine Persistantes

1. Phase de conception :
   1. Cas d’utilisation « S’authentifier »:
      1. Description textuelle :

|  |  |
| --- | --- |
| Cas d’utilisation | S’authentifier |
| Acteur | Utilisateur |
| Pré condition | L’utilisateur entre le site de l’application. |
| Post condition | L’utilisateur E-Banking est identifié comme étant un abonné ou un administrateur. |
| Description du scénario principal | -L’utilisateur remplit les champs d’authentification et clique sur le bouton «Login».  -Le système vérifie les informations saisies par l’utilisateur et affiche l’interface suivante. |
| Exception | Si un des champs est invalide, le système affiche un message d’erreur et réinitialise les champs. |

Tableau 3 : Description textuelle du cas d’utilisation « s’authentifier »

* + 1. Diagramme de classes de cas d’utilisation :

Le diagramme de classes du cas d’utilisation «s’authentifier» se traduit par les différentes classes participantes à la réalisation de ce dernier (figure 9).

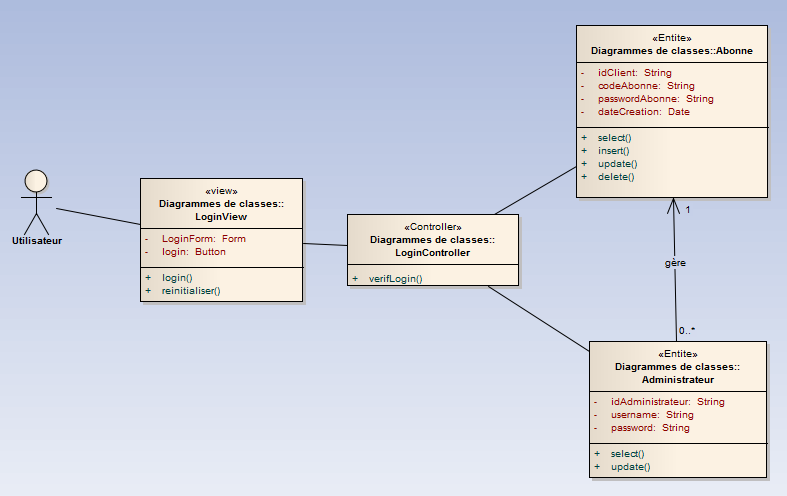


Figure 9 : Diagramme de classes de cas d’utilisation « s’authentifier »

* + 1. Diagramme de séquences :

A travers ce diagramme, nous décrivons le scénario du même cas d’utilisation. Dans un premier lieu, l’utilisateur E-Banking remplit le formulaire d’authentification et clique sur le bouton «Login». Ensuite le système vérifie s’il s’agit bien d’un abonné ou non puis s’il s’agit d’un administrateur ou non. Dans le cas du succès, l’utilisateur va être connecté et réorienter vers la page destinataire. Dans le cas d’échec, il doit remplir à nouveau le formulaire.

La figure 10 illustre le diagramme de séquences du cas d’utilisation «S'authentifier».

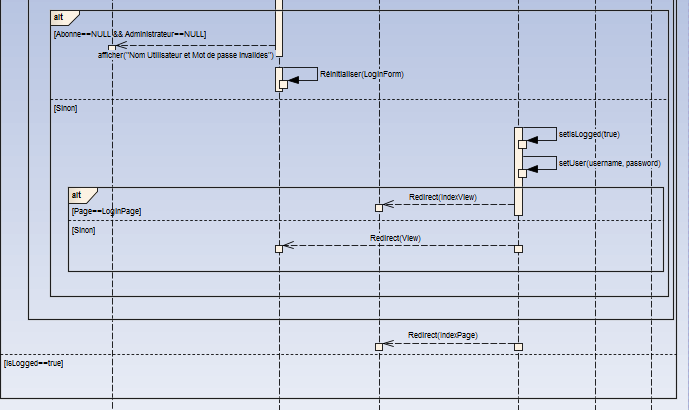
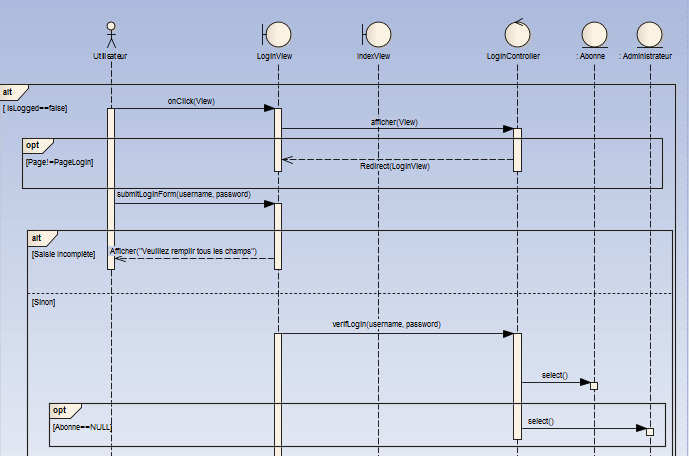


Figure 9 : Diagramme de séquences de cas d’utilisation « s’authentifier »

* 1. Cas d’utilisation « Commander un chéquier »:
     1. Description textuelle :
     2. Diagramme de classes de cas d’utilisation :
     3. Diagramme de séquences :

1. Conclusion :