

PROJET DE FIN D’ANNEE

*3éme Année en Ingénierie Informatique et Réseaux*

**Conception et développement d’une application web pour la gestion des comptes Bancaires**

***Réalisé par :***

*Mr Omar Tazi*

*Mlle Meriam Erragragui*

***Tuteur :***

*Encadrant Pédagogique : Pr Amina Ouatiq.*

**Année universitaire : 2024/2025**

# Remerciements

Nous tenons à adresse un grand merci au **Pr Amina Ouatiq** pour son soutien et son encadrement de ce projet ainsi quepour toute aide et assistance qui nous ont permit de surmounter et affronter les different problémes techniques et pratiques de notre projet.

**Table des matières**

[Introduction générale 1](#_bookmark0)

[Chapitre 1 : Présentation du cadre de projet](#_bookmark12)

1. [Introduction 3](#_bookmark28)
2. Etude de l’existant 3
   1. Solution proposée 3
3. Choix du modèle de développement 3
4. Planning prévisionnel 4
5. Conclusion 4

[Chapitre 2 : Spécification des besoins](#_bookmark12)

1. [Introduction 6](#_bookmark28)
2. [Spécification des besoins fonctionnels 6](#_bookmark14)
   1. [Création de compte utilisateur 6](#_bookmark15)
   2. [Connexion / Authentification 6](#_bookmark18)
   3. [Effectuer une transaction 6](#_bookmark18)
      1. [Effectuer un virement 6](#_bookmark19)
      2. [Effectuer un versement 6](#_bookmark20)
      3. [Effectuer un retrait 6](#_bookmark17)
   4. [Consultation du solde 7](#_bookmark18)
   5. [Historique des transactions 7](#_bookmark18)
3. [Spécification des besoins non fonctionnels 7](#_bookmark21)
   1. [Ergonomie de l’interface utilisateur 7](#_bookmark18)
   2. [Performance 7](#_bookmark18)
   3. [Fiabilité 7](#_bookmark18)
   4. [Faciliter de maintenance 7](#_bookmark18)
   5. [Disponibilité 7](#_bookmark18)
   6. [Compatibilité 7](#_bookmark18)
4. [Présentation des cas d’utilisation 7](#_bookmark22)
   1. [Présentation des acteurs 7](#_bookmark23)
   2. [Description des cas d’utilisation 8](#_bookmark24)
   3. [Diagramme des cas d’utilisation global 9](#_bookmark26)
5. Conclusion 9

[Chapitre 3 : Conception du système](#_bookmark27)

1. [Introduction 11](#_bookmark28)
2. [Modélisation dynamique 11](#_bookmark29)

2.1. [Diagrammes de séquences 11](#_bookmark30)

2.2. [Diagrammes de collaboration 12](#_bookmark35)

2.3. [Diagrammes d’états 12](#_bookmark36)

* 1. [Diagrammes d’état-transition 13](#_bookmark37)
  2. [Diagrammes d’activité 14](#_bookmark38)

1. [Modélisation statique 15](#_bookmark39)
   1. [Diagramme de classes 15](#_bookmark40)
   2. [Modèle relationnel 15](#_bookmark41)
   3. [Dictionnaire de données 16](#_bookmark42)
   4. [Architecture de l’application 16](#_bookmark44)
      1. [Architecture logiciel 16](#_bookmark45)
      2. [Architecture matériel 17](#_bookmark46)
2. [Conclusion 17](#_bookmark47)

[Chapitre 4 : Réalisation du système](#_bookmark48)

1. [Introduction 19](#_bookmark49)
2. [Environnement de développement 19](#_bookmark50)
   1. [Environnement matériel 19](#_bookmark51)
   2. [Environnement logiciel 19](#_bookmark52)
3. [Principales interfaces graphiques 21](#_bookmark53)
4. [Conclusion 23](#_bookmark54)

[Conclusion générale 24](#_bookmark55)

[Bibliographie et Néographie 25](#_bookmark56)

**Liste des figures**

[Figure 1 : Diagramme des cas d’utilisation………………………………………………………..…9](#_Diagramme_des_cas)

[Figure 2 : Diagramme de séquences "Versement"………………………………………………….11](#_bookmark30)

[Figure 3 : Diagramme de collaboration "Versement"……………………………………………....12](#_Diagrammes_de_collaboration)

[Figure 4 : Diagramme d’états "Versement"………………………………………………………...12](#_Diagrammes_d’état)

[Figure 5 : Diagramme d’état-transition "Transaction"……………………………………………...13](#_Diagrammes_d’état-transition)

[Figure 6 : Diagramme d’activité…………………………………………………………………….14](#_Diagrammes_d’activité)

[Figure 7 : Diagramme de classes……………………………………………………………………15](#_Diagramme_de_classes)

[Figure 8 : Modèle relationnel………………………………………………………………………..16](#_Modèle_relationnel)

[Figure 9 : Diagramme de composant………………………………………………………………..17](#_Architecture_logicielle)

[Figure 10 : Diagramme de déploiement……………………………………………………………..17](#_Architecture_matérielle)

Figure [11](#_bookmark53) : Page de connexion………………………………………………………………………21

Figure 12 : Page d'accueil principale………………………………………………………………..22

Figure 13 : Effectuer un retrait………………………………………………………………………23

Figure 14 : Effectuer un virement…………………………………………………………………...23

Figure 15 : Historique des transactions……………………………………………………………...24

## Liste des tableaux

[Tableau 1 : Planning prévisionnel 4](#_Toc198579035)

[Tableau 2 : Description du cas d’utilisation « Effectuer un virement » pour l’acteur « Client » 8](#_Toc198579036)

[Tableau 3 : Dictionnaire de données 16](#_Toc198579037)

[Tableau 4 : Caractéristiques de l'ordinateur 13](#_Toc198579038)

Introduction générale

**Introduction générale**

La numérisation croissante des services bancaires a profondément transformé la manière dont les utilisateurs interagissent avec les établissements financiers. Aujourd’hui, la majorité des opérations bancaires qu’il s’agisse de consultations de solde, de virements, de paiements ou encore de gestion de compte s’effectue à distance, via des plateformes numériques sécurisées. Cette évolution, rendue possible par l’essor des technologies web et mobiles, répond à un besoin croissant d’accessibilité, de rapidité et de simplicité d’usage.

Dans ce contexte, nous avons entrepris le développement d’une application de gestion bancaire en ligne, à la fois intuitive, sécurisée et fonctionnelle. Cette application permet aux utilisateurs de créer leur propre compte, de s’y connecter en toute sécurité, et d’accéder à un espace personnel depuis lequel ils peuvent effectuer diverses opérations : versements, retraits, virements vers d’autres comptes, paiement de factures, ainsi que la consultation de leur solde et de l’historique détaillé de leurs transactions.

Le projet a été conçu dans une logique d’apprentissage appliqué, combinant à la fois les compétences acquises en analyse des besoins, modélisation UML, conception de bases de données relationnelles, développement backend avec Django, interface web avec HTML/CSS, et gestion des transactions sécurisées. Il répond non seulement à des objectifs pédagogiques clairs, mais aussi à des enjeux concrets en matière d’expérience utilisateur et de fiabilité des services bancaires numériques.

Notre démarche s’est appuyée sur une méthodologie rigoureuse, incluant une phase d’analyse des exigences fonctionnelles et non fonctionnelles, la conception détaillée du système à l’aide de diagrammes UML, la modélisation et la structuration de la base de données, suivies par une phase de réalisation technique et de validation des fonctionnalités implémentées.

À travers ce rapport, nous présenterons les différentes étapes de la réalisation de cette application, depuis sa conception initiale jusqu’à sa mise en œuvre finale, en mettant en évidence les choix techniques effectués, les difficultés rencontrées, ainsi que les perspectives d’amélioration envisageables.

*Rapport de PFA : Système de Gestion bancaire* 1

|  |  |
| --- | --- |
| **Chapitre 1** | **Présentation du cadre de projet** |

Chapitre 1 : Présentation du cadre de projet

###### Introduction

Ce chapitre a pour objectif de fournir une vue d’ensemble du projet réalisé dans le cadre de notre stage de perfectionnement. Il introduit le contexte général du projet, en expliquant les motivations à l’origine de sa mise en place, les objectifs poursuivis, ainsi que la démarche adoptée. Dans un monde où les services bancaires numériques occupent une place prépondérante, la nécessité de concevoir des outils de gestion simples, fiables et accessibles est de plus en plus importante. C’est dans cette optique que nous avons choisi de développer une application de gestion bancaire permettant à un client de créer un compte, d’y accéder en toute sécurité et de réaliser diverses opérations comme les versements, retraits ou virements. Ce chapitre présente également les outils utilisés pour la réalisation du projet ainsi que la méthode de travail adoptée tout au long de son développement.

###### Etude de l’existant

* 1. Solution proposée

Nous avons proposé de concevoir une application web complète permettant aux utilisateurs de créer un compte, se connecter, effectuer des opérations (versements, retraits, virements), et consulter l’historique de leurs transactions.

Cette solution vise à améliorer l’accessibilité, la rapidité et la fiabilité des services bancaires, tout en assurant la sécurité et la traçabilité des données. Elle présente l’avantage d’être simple à utiliser, centralisée et adaptable. Toutefois, elle peut être améliorée en intégrant des rôles multiples ou des niveaux de sécurité plus avancés.

Notre projet se distingue par sa réalisation complète, depuis la modélisation jusqu’au développement, ce qui en fait une solution originale, conçue sur mesure pour répondre aux besoins identifiés.

###### Choix de modèle de développement

Pour la réalisation de ce projet, nous avons adopté un modèle de développement en cascade, structuré autour de phases bien définies : analyse, conception, implémentation, tests et validation. Ce modèle a été choisi pour sa clarté et sa simplicité d’application, particulièrement adaptée à un projet pédagogique de taille moyenne réalisé par deux personnes. Il permet d’avancer de manière linéaire tout en validant chaque étape avant de passer à la suivante. Ce choix a favorisé une organisation méthodique, une bonne répartition des tâches et un suivi clair de l’évolution du projet.

Chapitre 1 : Présentation du cadre de projet

1. Planning prévisionnel

Tableau  : Planning prévisionnel

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Semaine**  **Etape** | **Février** | | | | **Mars** | | | | **Avril** | | | | **Mai** | | | |
| **1** | **2** | **3** | **4** | **1** | **2** | **3** | **4** | **1** | **2** | **3** | **4** | **1** | **2** | **3** | **4** |
| *Etude préalable* | **X** | **X** | **X** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| *Conception* |  |  | **X** | **X** | **X** | **X** |  |  |  |  |  |  |  |  |  |  |
| *Réalisation* |  |  |  |  | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** |  |  |
| *Test et Validation* |  |  |  |  |  |  |  |  |  |  |  |  | **X** | **X** | **X** | **X** |

###### Conclusion

À l’issue de ce premier chapitre, nous avons posé les bases du projet en exposant clairement son contexte, ses objectifs et la problématique à laquelle il répond. Nous avons également présenté les différentes technologies retenues pour sa réalisation, ainsi que la méthodologie adoptée pour structurer et encadrer le travail. Ces éléments introductifs permettent de mieux appréhender la suite du rapport, qui détaillera successivement l’analyse des besoins, la conception du système, sa mise en œuvre technique, ainsi que les tests réalisés et les résultats obtenus. Cette introduction générale constitue ainsi une étape essentielle pour comprendre la portée du projet et son intérêt pédagogique et technique.

|  |  |
| --- | --- |
| **Chapitre 2** | **Spécification des besoins** |

Chapitre 2 : Spécification des besoins

###### 

###### Introduction

###### Ce chapitre a pour objectif de formaliser les besoins liés à l’application de gestion bancaire que nous souhaitons développer. Il s’agit d’identifier précisément les fonctionnalités attendues par les utilisateurs, ainsi que les contraintes techniques et organisationnelles à respecter. L’analyse des besoins permet de définir les cas d’utilisation, les rôles des acteurs impliqués, et d’établir une base solide pour la phase de conception. Cette étape est primordiale car elle oriente toutes les décisions futures en matière d’architecture, de technologie et d’implémentation.

###### Spécification des besoins fonctionnels

Les besoins fonctionnels définissent les opérations que le système doit permettre à l'utilisateur d'effectuer. Ce sont les services de base qu’une application doit fournir qui sont :

* 1. Création de compte utilisateur
* Permettre aux nouveaux utilisateurs de s'inscrire via un formulaire sécurisé.
* Collecter les informations personnelles : nom, prénom, courriel, numéro de téléphone, mot de passe.
* Enregistrer le compte dans la base de données.
  1. Connexion / Authentification
* Permettre aux utilisateurs à se connecter à leurs comptes à l’aide d’un identifiant et d’un mot de passe.
* Erreurs de connexion (mot de passe erroné ou utilisateur non trouvé).
* Rediriger l'utilisateur vers son tableau de bord personnel après connexion.

**2.3.** Effectuer une transaction

2.3.1. Effectuer un virement

* Sélectionner un compte bénéficiaire.
* Saisir le montant et la description de l’opération.
* Vérifier que le solde est suffisant.
* Enregistrer la transaction et mettre à jour les soldes des deux comptes.

2.3.2. Effectuer un versement

* Saisir le montant à verser.
* Enregistrer l’opération et mettre à jour le solde du compte utilisateur.

2.3.3. Effectuer un retrait

* Saisir le montant à retirer.
* Vérifier que le solde est suffisant.
* Valider et enregistrer l’opération, mettre à jour le solde.

Chapitre 2 : Spécification des besoins

**2.4.** Consultation du solde

* Afficher le solde disponible en temps réel.
* Mettre à jour automatiquement après chaque opération.

**2.5.** Historique des transactions

* Afficher toutes les transactions déjà effectuées (virements, retraits et versements).
* Permettre des filtres par libellé de transaction.

###### Spécification des besoins non fonctionnels

Les besoins non fonctionnels sont liés à la manière dont le système doit fonctionner plutôt qu’à ce qu’il fait.

**3.1.** Ergonomie de l’interface utilisateur

* Navigation fluide entre les pages.
* Messages d’erreurs et de confirmation clairs.

**3.2.** Performance

* Temps de réponse rapide aux requêtes.
* Optimisation des requêtes vers la base de données.

**3.3.** Fiabilité

* Tolérance aux pannes (gestion des exceptions).
* Enregistre les erreurs graves à des fins de débogage.

**3.4.** Disponibilité

* Application accessible à tout moment (hébergement stable).

**3.5.** Faciliter de maintenance

* Code bien structuré pour faciliter les mises à jour.

**3.6.** Compatibilité

* Compatible avec les navigateurs modernes.
* Utilisation de standards web reconnus.

###### Présentation des cas d’utilisation

* 1. Présentation des acteurs

Les clients : les clients sont les principaux utilisateurs du système. Ils accèdent aux services bancaires via l'application.

Chapitre 2 : Spécification des besoins

Ils peuvent créer un compte, se connecter en toute sécurité, effectuer diverses transactions financières telles que des virements, des retraits, des dépôts et même consulter les informations de son compte.

Le client est au cœur de l’application : toutes les fonctionnalités sont conçues pour répondre à ses besoins de gestion bancaires, garantissant une expérience fluide, intuitive et sécurisée.

* 1. Description des cas d’utilisation

Les actions principales qu’un client peut effectuer lors de son utilisation du système sont :

* Créer un compte.
* Se connecter.
* Consulter le solde.
* Effectuer un virement.
* Effectuer un versement.
* Effectuer un retrait.
* Consulter l’historique des transactions.

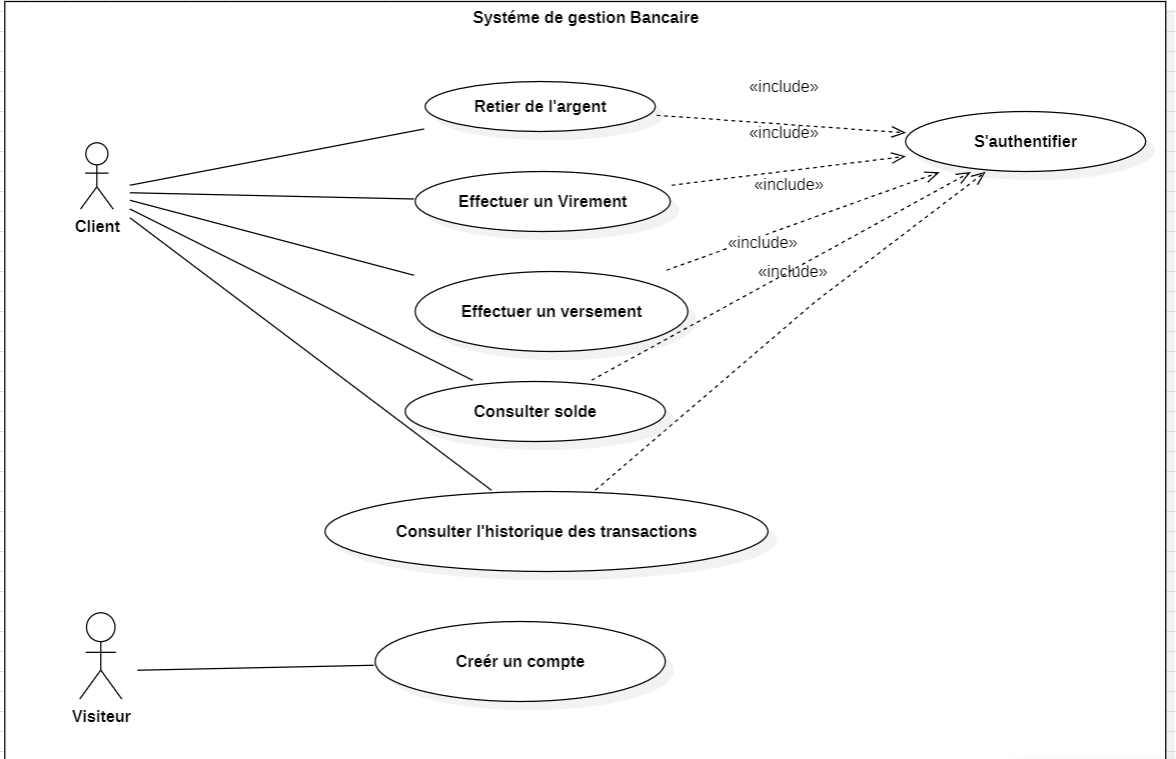
Tableau  : Description du cas d’utilisation « Effectuer un virement » pour l’acteur « Client »

|  |  |
| --- | --- |
| ***Cas n°*** | *Effectuer un virement* |
| ***Acteur*** *:* | *Client* |
| ***Objectif :*** | *Transférer de l’argent à un autre compte* |
| ***Pré-conditions :*** | *Le client doit être connecté et doit avoir un solde suffisant* |
| ***Post-conditions :*** | *Le montant est débité du compte du client connecté et est créditer dans le comte du client destinataire et le virement est enregistré.* |
| ***Scénario nominal :*** | *Tout d’abord on a la saisie des informations, ensuite vérification de la cohérence des infos, mise à jour des comptes et finalement la confirmation de la transaction.* |

Chapitre 2 : Spécification des besoins

* 1. Diagramme des cas d’utilisation global

Ce diagramme représente les interactions possibles entre les acteurs (Client et Visiteur) et le système de gestion bancaire. Il met en évidence les principales fonctionnalités offertes par l’application, telles que la création de compte, l’authentification, les opérations bancaires (versement, retrait, virement), la consultation du solde et l’historique des transactions. Les relations « include » indiquent que l’authentification est une étape nécessaire pour accéder aux fonctionnalités principales du systéme.



**Figure 1 : Diagramme des cas d’utilisation**

###### Conclusion

L’analyse menée dans ce chapitre a permis de dégager l’ensemble des exigences fonctionnelles et non fonctionnelles de l’application. Les cas d’utilisation ont été décrits et accompagnés de leurs diagrammes, permettant de clarifier les interactions entre le client et le système. Nous avons également présenté le modèle relationnel issu de la modélisation conceptuelle, ainsi que le dictionnaire de données précisant la structure des tables. Cette étape a posé les fondations sur lesquelles reposera la conception détaillée du système.

|  |  |
| --- | --- |
| **Chapitre 3** | **Conception du système** |

Chapitre 3 : Conception du système

###### Introduction

La phase de conception vise à transformer les besoins exprimés en une architecture claire et structurée, qui facilitera le développement de l’application. Dans ce chapitre, nous présentons les différentes représentations UML utilisées pour modéliser la structure et le comportement du système. À travers les diagrammes de cas d’utilisation, de classes, de séquence, d’activités, d’états-transitions, de composants et de déploiement, nous cherchons à garantir une compréhension globale du fonctionnement interne du système. Cette modélisation nous permet également d’anticiper les choix techniques à réaliser lors de l’implémentation.

###### Modélisation dynamique

* 1. Diagrammes de séquences

Ce diagramme illustre le déroulement chronologique des interactions entre le client et le système lors d’un versement d’argent. Il commence par la demande de saisie du montant, suivie de sa validation par le système, puis l’enregistrement de la transaction. Le diagramme montre également deux alternatives possibles : soit le versement est validé et les informations sont mises à jour, soit il est annulé. Cela permet de visualiser le processus complet, y compris les éventuelles décisions conditionnelles.

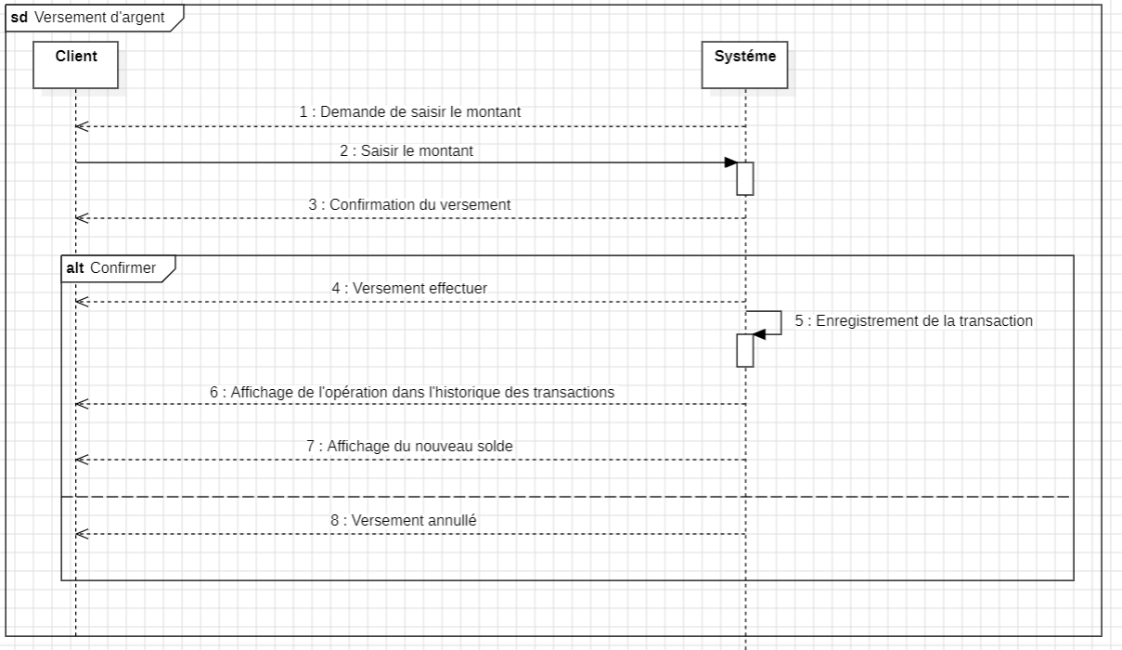


Figure : Diagramme de séquences "Versement"

* 1. Diagrammes de collaboration

Ce diagramme présente les objets impliqués dans l’exécution d’un versement et les messages échangés entre eux. Le client initie le versement via l’objet Transaction, qui envoie ensuite une requête au Compte pour créditer le montant. Ce diagramme met l’accent sur les **liens structurels** entre objets et leur collaboration pour accomplir une action métier précise, ici un versement.

Chapitre 3 : Conception du système

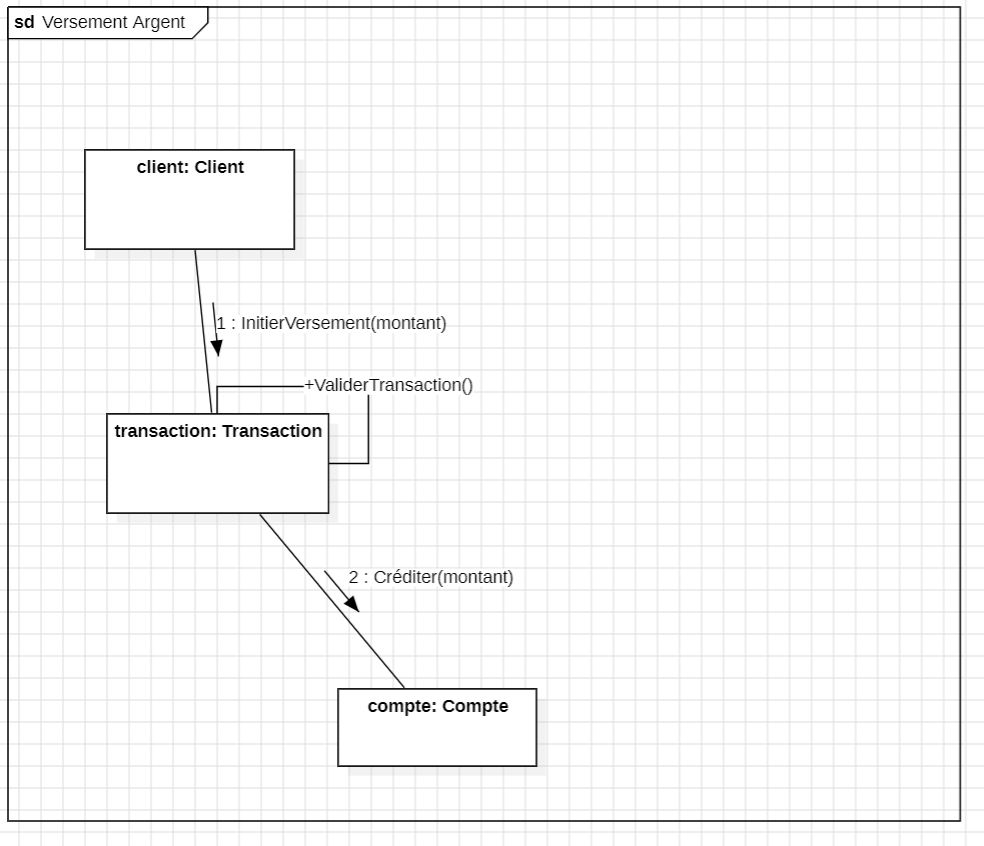


Figure : Diagramme de collaboration "Versement"

* 1. Diagrammes d’état

Ce diagramme décrit les différentes étapes qu’une opération de versement peut traverser dans le système bancaire. Il commence par l’initiation de la transaction, suivie d’une vérification du compte. Si cette vérification échoue, la transaction est immédiatement classée comme échouée. En revanche, si le compte est valide, le montant est crédité et la transaction est ensuite validée, menant à la fin du processus. Ce diagramme permet de représenter clairement les chemins possibles d’un versement en fonction de la réussite ou de l’échec des vérifications effectuées.

Une image contenant diagramme, texte, croquis, ligne

Le contenu généré par l’IA peut être incorrect.

Figure : Diagramme d’états "Versement"

Chapitre 3 : Conception du système

* 1. Diagrammes d’état-transition

Ce diagramme permet de visualiser les différents états d’une transaction bancaire et les transitions possibles entre ces états. La transaction débute par sa création, à la suite d’une action d’initialisation. Elle peut ensuite être annulée, menant directement à un état final d’annulation, ou bien être validée, ce qui la fait passer à l’état "en cours de traitement". Une fois dans cet état, deux issues sont possibles selon le déroulement de l’exécution : si tout se passe correctement, la transaction est marquée comme réussie, sinon elle est classée comme échouée en cas d’erreur. Ce diagramme met ainsi en évidence les différentes éventualités qui peuvent affecter une transaction tout au long de son traitement.

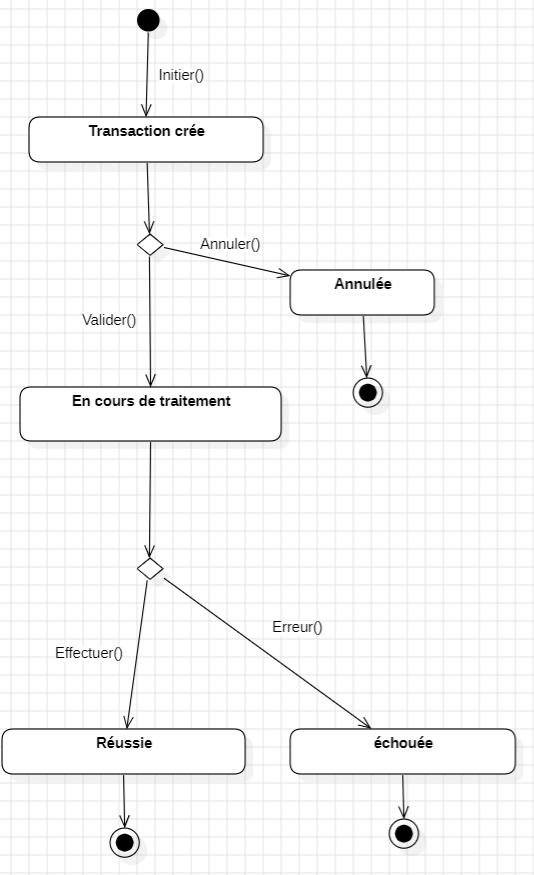


Figure : Diagramme d’état-transition "Transaction"

Chapitre 3 : Conception du système

* 1. Diagrammes d’activité

Ce diagramme représente les différentes actions réalisées par les intervenants du système, à savoir le client, le service des comptes et le service des transactions. Il débute avec la création d’un compte par le client, ce qui entraîne l’enregistrement du compte et l’attribution d’un identifiant dans le service concerné. Le client peut ensuite effectuer diverses opérations comme un virement, un retrait ou un versement. Chaque opération passe par une vérification du solde, qui peut conduire soit à l’annulation de l’opération en cas de solde insuffisant, soit à son approbation. Lorsqu’une opération est approuvée, elle est ensuite enregistrée dans le service des transactions. Le diagramme couvre aussi la consultation du solde, qui implique une recherche des données client pouvant aboutir à un affichage du solde ou à une erreur si les données sont introuvables. Ce diagramme illustre ainsi le déroulement global des activités selon les rôles impliqués, tout en mettant en évidence les conditions qui influencent la progression des processus.

Une image contenant texte, diagramme, Parallèle, Plan

Le contenu généré par l’IA peut être incorrect.

Figure : Diagramme d’activité

Chapitre 3 : Conception du système

###### Modélisation statique

* 1. Diagramme de classes

Ce diagramme représente la structure statique du système de gestion bancaire. Il modélise les trois classes principales de l’application : Client, Compte et Transaction.

La classe Client contient les informations personnelles de l’utilisateur et les méthodes liées à ses actions (s’authentifier, transférer, retirer…).

La classe Compte représente les comptes bancaires, avec les opérations de débit et de crédit.

La classe Transaction centralise toutes les opérations effectuées : versements, retraits, virements ou paiements.

Les associations entre ces classes illustrent que chaque transaction est effectuée par un client et liée à un compte.

Une image contenant texte, diagramme, ligne, Plan

Le contenu généré par l’IA peut être incorrect.

Figure : Diagramme de classe

Chapitre 3 : Conception du système

* 1. Modèle relationnel

Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.

Figure : Modèle relationnel

Le modèle relationnel ci-dessus montre les deux entités principales du système : compte\_client et transaction.

Chaque transaction est liée à un compte client via une clé étrangère, ce qui permet d’assurer la traçabilité de chaque opération.

* 1. Dictionnaire de données

Tableau : Dictionnaire de données

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nom de la colonne | Type de données | Taille | Obligatoire (Oui/Non) | Valeur par défaut | Valeurs autorisées | Clé | | Nom de la Table |
| Primaire | Etrangère |
| Id\_compte | INT | - | Oui | Auto-increment | Numérique | Oui | Non | Compte\_client |
| nom | VARCHAR | 50 | Oui | NULL | Texte | Non | Non | Compte\_client |
| prenom | VARCHAR | 50 | Oui | NULL | Texte | Non | Non | Compte\_client |
| email | VARCHAR | 100 | Oui | NULL | Format Email | Non | Non | Compte\_client |
| mot\_de\_passe | VARCHAR | 255 | Oui | NULL | Texte | Non | Non | Compte\_client |
| solde | DECIMAL | 10,2 | Oui | 0.00 | Numérique >=0 | Non | Non | Compte\_client |
| id\_transaction | INT | - | Oui | Auto-increment | Numérique | Oui | Non | Transaction |
| type\_transaction | ENUM | - | Oui | NULL | ‘versement’, ‘retrait’, ‘virement’. | Non | Non | Transaction |

* 1. Architecture de l’application
     1. Architecture logicielle

Ce diagramme représente l’architecture logicielle de l’application. Il illustre comment les différents composants du système interagissent entre eux : l’interface utilisateur communique avec les modules de gestion de compte, de transactions et d’authentification. Ces modules accèdent ensuite à la base de données pour lire ou enregistrer les informations. Cette séparation des responsabilités facilite la maintenance et l’évolutivité du système.

Chapitre 3 : Conception du système

Une image contenant diagramme, Plan, texte, Dessin technique

Le contenu généré par l’IA peut être incorrect.

Figure : Diagramme de composant

* + 1. Architecture matérielle

Ce diagramme représente l’architecture logicielle de l’application. Il illustre comment les différents composants du système interagissent entre eux : l’interface utilisateur communique avec les modules de gestion de compte, de transactions et d’authentification. Ces modules accèdent ensuite à la base de données pour lire ou enregistrer les informations. Cette séparation des responsabilités facilite la maintenance et l’évolutivité du système.

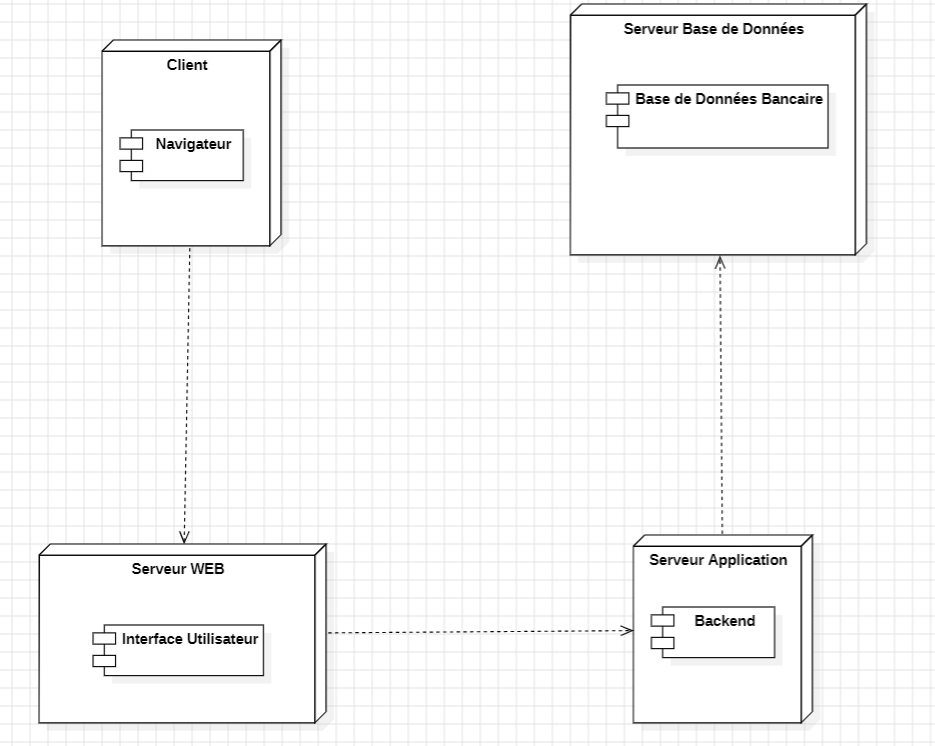


Figure : Diagramme de déploiement

Chapitre 3 : Conception du système

###### Conclusion

La phase de conception constitue une étape cruciale dans le développement de notre application de gestion bancaire. À travers ce chapitre, nous avons d'abord présenté l'architecture générale du système en mettant en évidence les différentes couches et leur interaction.

Divers diagrammes UML ont été élaborés pour illustrer de manière claire et structurée le fonctionnement interne du système : diagrammes de cas d’utilisation, de séquence, de collaboration, d’activité, d’état-transition, ainsi que le diagramme de classes. Ces représentations nous ont permis de modéliser les entités, les relations et les comportements attendus des objets tout au long de leur cycle de vie. Enfin, nous avons défini le modèle relationnel de la base de données et le dictionnaire de données pour garantir une organisation cohérente et efficace de l'information. Cette conception rigoureuse servira de base solide pour l'implémentation du système et contribue à anticiper les éventuels problèmes techniques ou fonctionnels.

|  |  |
| --- | --- |
| **Chapitre 4** | **Réalisation du système** |

Chapitre 4 : Réalisation du système

###### Introduction

###### Après avoir défini les besoins, conçu l’architecture du système et modélisé ses différentes composantes à l’aide d’outils UML, nous passons à la phase de réalisation. Cette étape marque le passage de la conception théorique à l’implémentation concrète du système. Elle consiste à traduire les diagrammes, modèles et spécifications en un code source fonctionnel, tout en respectant les contraintes techniques et les choix d’outils définis précédemment. Dans ce chapitre, nous présenterons l’environnement de développement utilisé, l’organisation du code, la structure de la base de données, ainsi que les principales fonctionnalités implémentées telles que la création de compte, la gestion des transactions et la consultation des informations utilisateur. Cette phase est essentielle pour concrétiser les objectifs fixés au début du projet.

###### Environnement de développement

* 1. Environnement matériel

Tableau : Caractéristiques de l'ordinateur

|  |  |
| --- | --- |
| **Nom de l’ordinateur** | LAPTOP-0FC93ON6 |
| **Système d’exploitation** | WINDOWS 11 Famille 64bits |
| **Fabricant du système** | LENOVO |
| **Modèle du système** | 82H8 |
| **Bios** | GGCN20WW |
| **Processeur** | 11TH Gen Intel® Core™ i5-1135G7 @ 2,40GHz (8CPUs) |
| **Mémoire** | 12288MB RAM |

* 1. Environnement logiciel

Le développement de notre application de gestion bancaire s’est appuyé sur plusieurs outils adaptés aux différentes étapes du projet, de la modélisation à la mise en œuvre technique.

* **Django :** Framework web Python utilisé pour développer la logique métier du système (backend), avec un ORM intégré facilitant l’interaction avec la base de données et un système d’authentification sécurisé.

Chapitre 4 : Réalisation du système

* **Django Template Language (DTL) :** Moteur de templates permettant de générer des pages HTML dynamiques en liant le backend aux données affichées côté client.
* **HTML & CSS :** HTML a été utilisé pour structurer les pages de l’interface utilisateur (formulaires, tableaux, etc.), tandis que le CSS a servi à améliorer l’apparence et l’ergonomie de l’application.
* **MySQL :** Système de gestion de base de données relationnelle utilisé pour stocker les informations sur les utilisateurs, comptes et transactions. Il a été choisi pour sa performance et sa compatibilité avec Django.
* **StarUML :** Outil de modélisation utilisé pour élaborer les différents diagrammes UML nécessaires à la conception du système (cas d’utilisation, classes, séquences, déploiement, etc.).

###### Principales interfaces graphiques

Cette interface représente la page d’authentification de l’application O&M BANK.

L’utilisateur doit saisir son identifiant et son mot de passe pour accéder à son espace personnel. Un lien en bas permet également de créer un nouveau compte. Le design est clair, moderne et professionnel, avec une image de fond représentant une agence bancaire, ce qui renforce l'identité visuelle de l'application.

Une image contenant texte, bâtiment, plein air, signalisation

Le contenu généré par l’IA peut être incorrect.

Figure : Page de connexion

Cette interface présente la page d'accueil du site web de O&M BANK après connexion. On peut y voir un message de bienvenue personnalisé "Bienvenue laila boujad !" en haut de la page. La plateforme affiche un récapitulatif des comptes avec la vente actuelle total de 21000,00 MAD.

L'interface comprend deux comptes distincts : un compte standard "Compte

Chapitre 4 : Réalisation du système

COMPT-3" avec une vente de 21000,00 MAD, et un compte épargne "Compte EPARGNE-3" nouvellement créé avec une vente de 0,00 MAD, comme l'indique la notification verte en haut "Compte épargne ajouté avec succès".

La navigation principale est située dans la barre bleue supérieure avec des options pour Accueil, Transactions, Historique, Contact et Déconnexion. Un bouton "Créer un compte épargne" est également visible dans la section principale. L'interface est épurée et fonctionnelle, permettant aux utilisateurs de gérer facilement leurs comptes bancaires en ligne.

Une image contenant texte, capture d’écran, conception, carte de visite

Le contenu généré par l’IA peut être incorrect.

Figure : Page d'accueil principale

Cette interface permet à un utilisateur connecté d’effectuer un retrait depuis l’un de ses comptes.

Il sélectionne le compte à débiter, saisit le montant à retirer, puis clique sur le bouton « Effectuer le retrait ».

L’interface est simple, centrée sur l’action, et facile à comprendre, ce qui permet une expérience utilisateur fluide.

Chapitre 4 : Réalisation du système

Une image contenant texte, capture d’écran, intérieur, conception

Le contenu généré par l’IA peut être incorrect.

Figure : Effectuer un retrait

L’interface ci-dessus représente le formulaire de virement de l’application bancaire. L’utilisateur peut sélectionner un compte source, ici un compte épargne nommé "EPARGNE-3", et spécifier un compte destinataire (par exemple, "COMPT-11"). Il peut ensuite saisir le montant à transférer, affiché ici comme "5000", avant de cliquer sur le bouton « Envoyer le Virement » pour valider l’opération. L’interface est claire, épurée, et suit une charte graphique en bleu clair avec un design moderne et responsive.

Une image contenant texte, capture d’écran, Police, conception

Le contenu généré par l’IA peut être incorrect.

Figure : Effectuer un virement

Chapitre 4 : Réalisation du système

Cette interface permet à un utilisateur connecté de consulter l’historique des différentes transactions effectuer sur ces comptes (standard et épargne).

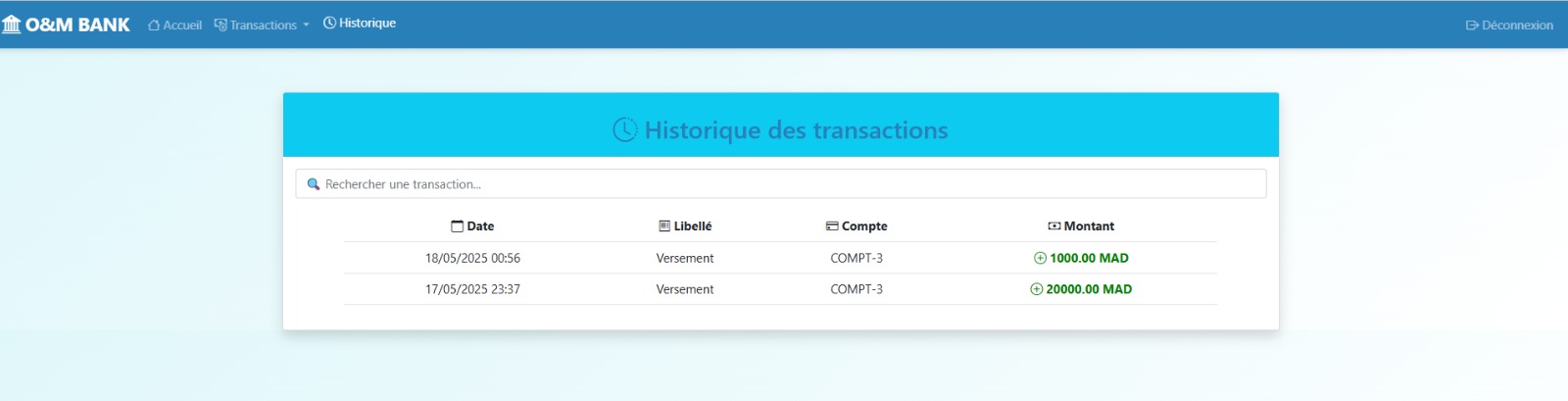


Figure : Historique des transactions

###### Conclusion

La réalisation de l’application de gestion bancaire a permis de mettre en œuvre les concepts étudiés dans les phases précédentes, en développant un système web fonctionnel, sécurisé et structuré. Toutes les fonctionnalités principales ont été intégrées avec succès, notamment la création de compte, l’authentification, les opérations bancaires (versement, retrait, virement), ainsi que l’affichage des soldes et de l’historique des transactions. L’architecture du système, basée sur Django, HTML/CSS et MySQL, s’est révélée efficace pour répondre aux besoins fonctionnels et non fonctionnels définis. Malgré quelques limitations techniques, cette étape a permis d’atteindre les objectifs principaux du projet, tout en posant les bases d’éventuelles améliorations futures.

*Conclusion générale*

### **Conclusion générale**

L’objectif principal de ce projet était de mettre en pratique tous nos connaissances acquises, en développant une application concrète répondant à un besoin réel. Dans ce cadre, le travail réalisé a porté sur la conception et la réalisation d’une application de gestion bancaire, permettant aux utilisateurs de créer des comptes, de se connecter à un espace personnel sécurisé, d’effectuer diverses opérations bancaires telles que les versements, retraits et virements, ainsi que de consulter l’historique des transactions. L’ensemble du développement a suivi une démarche structurée : analyse du besoin, modélisation UML, conception de la base de données relationnelle, et mise en œuvre des interfaces utilisateurs. Ce projet a permis de répondre efficacement aux problématiques posées au départ, notamment en ce qui concerne la gestion sécurisée des comptes et la traçabilité des opérations. Toutefois, un regard critique sur le travail accompli permet de relever certaines limites. Par exemple, l’extension vers des fonctionnalités plus poussées comme la génération de rapports détaillés ou l’intégration d’une API bancaire reste envisageable. En somme, ce projet a constitué une expérience enrichissante, tant sur le plan technique que professionnel, en consolidant nos compétences en développement logiciel, en gestion de base de données et en conception orientée objet.

*Bibliographie et Néographie*

### **Bibliographie et Néographie**

**Développement Front-End**

1. Duckett, J. (2011). HTML and CSS: Design and Build Websites. Wiley.

ISBN : 9781118008188

Disponible sur : https://www.wiley.com/en-us/HTML+and+CSS%3A+Design+and+Build+Websites-p-9781118008188

2. Flanagan, D. (2020). JavaScript: The Definitive Guide (7th ed.). O’Reilly Media.

ISBN : 9781491952023

Disponible sur : https://www.oreilly.com/library/view/javascript-the-definitive/9781491952023/

3. Bootstrap. (2024). Bootstrap Documentation.

Disponible sur : https://getbootstrap.com/docs/5.3/getting-started/introduction/

**Développement Back-End avec Django**

4. Holovaty, A., & Kaplan-Moss, J. (2009). The Definitive Guide to Django: Web Development Done Right (2nd ed.). Apress.

ISBN : 9781430219361

Disponible sur : https://link.springer.com/book/10.1007/978-1-4302-1937-8

5. Django Software Foundation. (2024). Django Documentation.

Disponible sur : https://docs.djangoproject.com/en/5.0/

**Base de Données – MySQL**

6. MySQL. (2024). MySQL 8.0 Reference Manual. Oracle Corporation.

Disponible sur : https://dev.mysql.com/doc/refman/8.0/en/

Sécurité Web

7. OWASP Foundation. (2023). OWASP Top Ten Web Application Security Risks – 2023.

Disponible sur : https://owasp.org/Top10/