

REPUBLIQUE DE COTE D'IVOIRE

UNION-DISCIPLINE-TRAVAIL

MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE NANGUI ABROGOUA

Unité Fondamentale de Recherche des Sciences Fondamentales et Appliquées

2020-2021



MÉMOIRE POUR OBTENTION DU DIPLÔME DE MASTER

Mention : INFORMATIQUE

Spécialité : GÉNIE INFORMATIQUE

THÈME

**CONCEPTION D'UNE APPLICATION DE TRANSFERT D'UNITÉ DE
COMMUNICATION POUR L'ENTREPRISE ONEMART**

Présenté par :

TUO ADAMA

Date de soutenance : xx octobre 2021

Composition du jury :

— Président :

— Membre :

— Membre :

— Membre :

Encadreur :

Dr ZEZE DJEDJE SYLVAIN,
Maitre-Assistant UNA

Remerciements

Tout d'abord ce travail ne serait pas aussi riche et n'aurait pas pu voir le jour sans l'aide de l'encadrement de **Dr Zézé Sylvain**, je vous remercie pour la qualité exceptionnelle de votre encadrement, votre rigueur et surtout votre disponibilité durant la préparation de ce mémoire.

Mon remerciement s'adresse à **Dr Tchimou N'Takpé** pour ses encouragements et ses précieux conseils.

J'adresse mes sincères remerciements à tous les professeurs, intervenants et toutes les personnes qui par leurs paroles, leurs écrits, leurs conseils et leurs critiques ont guidé mes réflexions tout au long de la rédaction de ce mémoire.

Je remercie mes très chers parents, **Kone Naminata** et **Tuo Zahana**, qui ont toujours été là pour moi. Je remercie surtout mon oncle **Kone Souleymane** pour ses encouragements sans fin tout au long de mon parcours scolaire.

À tous ces intervenants, je présente mes remerciements, mon respect et ma gratitude.

Dédicaces

A mes très chers parents **Tuo Zahana** et **Kone Naminata** qui ont toujours été là pour moi, qui m'ont soutenu et encouragé durant toutes ces années d'études. J'espère qu'ils trouveront dans ce travail toute ma reconnaissance et tout mon amour.

A mon cher frère **Tuo Kolo** et à mes sœurs : **Tuo Yoh abi**, **Yire Fatoumata** et **Barah Ramatou**.

A mes meilleurs amies

Je dédie ce mémoire.

Résumé

Faisant face à un système obsolète basé sur une architecture complexe, l'entreprise ONEMART retrouvait sa productivité altérée par le fait de la non-automatisation des tâches qu'elle effectuait. C'est dans cette optique que s'est déroulée le stage de notre mémoire de Master. En effet tout au long de ce stage nous avons eu à mettre une application Web de transferts et recouvrement d'argent à une plage de client (aussi appelé intermède) suivant un mode de paiement donnée (Crédit, comptant, bancaire, etc ...). Une fois l'opération (transfert ou recouvrement) effectuée, grâce à l'interopérabilité offerte par l'application, un système externe (nous avons aussi appelé validateur) récupère les informations de l'opération et la valide grâce à une syntaxe USSD.

Ainsi Afin de cadrer le développement de notre application et de répondre aux exigences de l'entreprise, nous avons élaboré un plan de conception grâce aux différents diagrammes UML. Vu les fonctionnalités offertes par l'application, elle a été développée en utilisant le framework **Laravel** et d'autres outils de développement (le système de gestion de base de donnée relationnelle **MySQL**, une bibliothèque JavaScript **AlpineJs**, etc...) pour faciliter le développement de l'application tout en assurant une bonne sécurité de celle-ci.

Mots clés : Interopérabilité, Intermèdes, Transferts, Recouvrements, Rechargements

Abstract

Faced with an obsolete system based on a complex architecture, ONEMART found its productivity altered by the fact of the non-automation of the tasks it was carrying out. It is in this perspective that the internship of our Master's thesis took place. In fact, throughout this throughout this training course we had to put a Web application of transfers and recovery of argent to a range of customer (also called intermediary) according to a given method of payment (Credit, cash, bank, etc ...). Once the operation (transfer or recovery) is done, thanks to the interoperability provided by the application, an external system (also called validator) retrieves the information of the operation and validates it thanks to a USSD syntax. Thus, in order to frame the development of our application and to meet the requirements of the company, we In order to frame the development of our application and to meet the requirements of the company, we have elaborated a design plan thanks to the various UML diagrams. Given the functionalities offered by the application, it was developed using the Laravel framework and other other development tools (the relational database management system MySQL , a JavaScript library AlpineJs, etc. ...) to facilitate the development of the application while while ensuring a good security of this one.

Key words : Interoperability, Interludes, Transfers, Recoveries, Reloads

Table des matières

Resumé	4
Liste des tableaux	ii
Table des figures	iv
Liste des abréviations	v
Introduction générale	1
1 CADRE GÉNÉRAL	2
1.1 ONEMART	2
1.2 Présentation de l'entreprise	2
1.3 Cadre du projet	2
1.3.1 Problématique	2
1.3.2 Objectifs du projet	3
1.3.3 Élaboration du cahier des charges	3
1.3.4 Planification de l'application	4
1.4 Conclusion	5
2 CONCEPTION ET ÉTUDE TECHNIQUE	6
2.1 Conception	6
2.2 Etude comparative entre UML et MERISE	6
2.2.1 Présentation de MERISE	6
2.2.2 Présentation de UML	6
2.2.3 Comparaison	7
2.2.4 Pourquoi le choix d'UML ?	7
2.3 Conception avec UML	7
2.3.1 Principaux diagrammes UML	7
2.3.2 Diagramme de cas d'utilisation	8
2.3.3 Diagramme de classe	16
2.4 Schéma relationnel	24
2.4.1 Modèle logique des données ou MLD	24
2.4.2 Modèle physique de donnée ou MPD	25

2.5	Outils de développement	27
2.5.1	Technologie front-end	27
2.5.2	Choix du framework back-end	29
2.5.3	Système de gestion de base de données	30
2.5.4	Avantages	31
2.5.5	Architecture structurel	31
2.5.6	Autre outils utilisés	33
3	PRÉSENTATION DE L'APPLICATION	34
3.1	Authentification	34
3.2	Liste des agents	35
3.3	Ajouter d'un agent	35
3.4	Les Rôles	36
3.5	Ajouter intermède	37
3.6	Liste des intermèdes	38
3.7	Les modes de paiements	39
3.8	Liste des types	39
3.9	Recouvrement	40
3.10	Transfert	41
3.11	Facture après une opération	42
3.12	Message des opérations	43
3.13	États des paiements	45
3.14	Conclusion	45
	Bibliographie	46

Liste des tableaux

2.1	Description de l'authentification	10
2.2	Description de cas d'utilisation : Gestion des compte	11
2.3	Description du cas d'utilisation : Gestion des intermèdes	12
2.4	Description du cas d'utilisateur : Gestion des numéros	13
2.5	Description du cas d'utilisateur : Gestion des opérations	15
2.6	Description de l'authentification	16
2.7	tableau de classe <i>Personnel</i>	20
2.8	tableau de classe <i>Intermed</i>	21
2.9	tableau de classe <i>Numéro</i>	21
2.10	tableau de classe <i>objectif</i>	21
2.11	tableau de classe <i>Opération</i>	22
2.12	tableau de classe <i>Opération</i>	22
2.13	tableau de classe <i>message</i>	22
2.14	tableau de classe <i>transfert</i>	23
2.15	tableau de classe <i>recouvrement</i>	23
2.16	tableau de classe <i>facture</i>	23
2.17	tableau de classe <i>ModePaiement</i>	23
2.18	tableau de classe <i>ouverture</i>	23
2.19	tableau de classe <i>agence</i>	24
2.20	tableau de classe <i>secteur</i>	24

Table des figures

2.1	Diagramme de cas d'utilisation : vue globale	9
2.2	Diagramme de cas d'utilisation : gestion des comptes	11
2.3	Diagramme de cas d'utilisation : gestion des comptes	12
2.4	Diagramme de cas d'utilisation : gestion des opérations	14
2.5	Diagramme de cas d'utilisation : gestion des messages	16
2.6	représentation d'une classe	17
2.7	multiplicité	18
2.8	association	18
2.9	diagramme de classe de l'application	20
2.10	Modèle physique des données	26
2.11	Logo HTML 5	27
2.12	Logo CSS 3	27
2.13	Logo Bootstrap 5	28
2.14	Logo AlpineJS	28
2.15	Logo de Laravel	29
2.16	Logo de Livewire	30
2.17	Logo de MySQL	31
2.18	MVC	32
2.19	MVC	33
2.20	Draw.io	33
3.1	Page d'authentification	34
3.2	Liste des agents	35
3.3	Ajouter un agent	36
3.4	Les rôles	36
3.5	Ajouter intermède	37
3.6	Ajout de numéros	37
3.7	38
3.8	Liste des intermèdes	38
3.9	Mode de paiement	39
3.10	Les types	40
3.11	Recouvrement	40
3.12	Details Recouvrement	41

3.13 Confirmation recouvrement	41
3.14 Transfert	42
3.15 Detail du transfert	42
3.16 Présentation de facture	43
3.17 Message des opérations	44
3.18 Message des transferts	44
3.19 Etat de paiement	45

Liste des abréviations

MERISE : *Méthode d'Étude et de Réalisation Informatique pour les Systèmes d'Entreprise*

UML : *Unified Modeling Language*

VPN : *Virtual Private Network*

PC : *Personal Computer*

VB : *Visual Basic*

API : *Application Programming Interface*

HTML : *Hyper Text Markup Language*

CSS : *Cascading Style Sheets*

MVC : *Model View Controller*

USSD : *Unstructured Supplementary Service Data*

Introduction générale

Depuis longtemps, La technologie n'a cessé de faire évoluer la productivité de milliers d'entreprises presque dans tous les secteurs d'activités.

Le passage de la mécanique aux domaines de l'informatique, de l'électronique de la domotique a révolutionné la vie journalière de l'être humain. Aujourd'hui, vu l'intérêt croissant de vouloir gagner en temps, d'automatiser les tâches répétitives, cela a poussé petites, moyennes et grandes entreprises à chercher des solutions informatiques capables de répondre à leurs besoins.

Il y'a quelques années de cela, l'entreprise ONEMART disposait d'une application legacy qui permettait d'effectuer des **transferts** et **recouvrements** sur les différents numéros des ses clients qui sont appelés **intermédiares**. Cette application n'étant plus maintenable et ne pouvant pas interagir avec d'autre système externe (principe d'**interopérabilité**), nous avons décider de refaire l'architecte de l'application en partant d'une conception simple.

C'est dans ce cadre s'inscrit notre projet de fin d'études qui consiste à réaliser une application de transfert et de recouvrement pour une entreprise appelée **ONEMART**.

Ainsi notre objectif a été de partir d'une application legacy et développer une nouvelle application qui pourra s'adapter dans le temps et pourra apporter un gain de temps de considérable à l'entreprise.

Ainsi pour bien cadrer l'étude de ce stage, notre travail se présentera comme suite :

- En premier lieu nous allons présenter l'entreprise **ONEMART** et le cadre dans lequel se situera le projet.
- En second lieu nous allons faire la présentation des différentes phases de conception et l'étude technique du projet.
- Enfin nous présenterons l'application grâce aux différentes captures d'écran accompagnées de quelques détails.

Chapitre 1

CADRE GÉNÉRAL

1.1 ONEMART

1.2 Présentation de l'entreprise

ONEMART est une société expérimentée dans la commercialisation des produits et services de la téléphonie mobile sur le marché national et bénéficiant d'un personnel hautement qualifié. Onemart dispose d'un réseau propre avec plusieurs points de vente mais aussi d'un portefeuille clientèle conséquent. Depuis janvier 2010, elle est le distributeur exclusif (franchisé) d'atlantique télécom CI, société de droit ivoirien propriétaire d'un réseau de radiotéléphonie cellulaire exploité sous la marque Moov.

ONEMART assure ainsi l'exclusivité de la distribution des produits et des services de Moov (Kits, recharges physiques et électroniques, portables et autres services après vente) dans les zones géographiques suivantes : Yopougon, Dabou, Sikensi, Tiassale, N'douci, Jacqueville, Grand-lahou, N'zianoua,.

1.3 Cadre du projet

Dans le cadre de notre mémoire nous nous sommes concentrés sur une des branches de l'entreprise qui est le rechargement des clients qui sont aussi appelés intermédiaires.

1.3.1 Problématique

Dans les années antérieures, l'entreprise ONEMART disposait d'une application VB déjà compilée avec toutes les plages de numéros dont elle disposait. Ainsi lors du passage des numérotations à dix chiffres, l'entreprise faisait face à un très gros problème qui était de mettre à jour les numéros des intermédiaires vu que le code source de l'application n'était plus disponible.

A son siège, l'entreprise disposait d'un PC central câblé en local avec un téléphone d'ancienne génération (Sony Ericsson) équipé d'une puce **emaster** qui récupérait par câble les opérations effectuées (Recouvrement et transfert) sur le PC pour les valider. Ainsi, pour faire des rechargements et transferts depuis un point de vente distant, l'on devait se connecter par VPN (Virtual Private Network) pour avoir accès au PC central. En effet, puisque le VPN fait transiter la connexion de l'utilisateur par un serveur distant, ce qui rajoute une étape intermédiaire lors de la transmission des informations. Ainsi l'on fait face à un débit de transmission moins stable.

Aussi sur le point sécuritaire, il est important de noter que les opérations se faisaient manuellement sachant bien que les montants de transfert étaient très élevés ce qui pouvait être très problématique du moment où l'on peut être exposé aux problèmes suivants : Se tromper sur le montant de transfert, transférer l'argent d'un intermède à un autre.

1.3.2 Objectifs du projet

Face aux problèmes cités ci-dessus, notre application aura pour objectifs de :

- Mettre en place une application WEB car en plus d'être accessible sur toutes les plateformes (peu importe le système d'exploitation de l'utilisateur), elle est simple et nécessite aucune installation.
- Que de passer par VPN (Virtual Private Network) pour avoir accès au PC central, l'application sera accessible peu importe la situation géographique de l'utilisateur.
- Fournir une interopérabilité entre l'application et d'autre système externe. C'est à dire rendre accessible par API (Application Programming Interface) les informations de transfert et recouvrement après chaque opération pour qu'un autre système puissent les récupérer et les valider par **USSD**.
- Offrir plusieurs modes de paiement aux intermèdes lors des transferts : comptant, crédit, bancaire, etc...
- Fournir une facture après chaque opération.

1.3.3 Élaboration du cahier des charges

Dans cette section, nous allons définir les différentes charges que notre application devra respecter lors de la conception. L'application aura pour charge :

- Créer un point d'entrée par lequel les données seront accessibles par un autre système externe.
- Persister les messages de transferts et recouvrement

- Sauvegarder les historiques de transfert.
- Gérer la liste des intermédiaires en ayant la possibilité de la modifier, ajouter et supprimer sans oublier de leur attribuer des numéros.
- Gestion du mode de paiement (paiement par carte bancaire, crédit ou comptant).
- Élaboration d'une facture après chaque opération avec la possibilité de la télécharger.
- Gérer l'état des opérations :
 - état initié : Lorsque l'opération vient d'être effectuée.
 - cours d'exécution : Lorsque l'application mobile récupère le message d'opération pour la valider.
 - exécuté : Lorsque l'opération est effectuée par l'application.
 - annulé : Lorsque l'opération est annulée.

1.3.4 Planification de l'application

Que l'on soit chef d'équipe ou responsable de sa propre activité, bien planifier ses projets est essentiel pour être efficace. Cela permet d'organiser son temps dépendamment du travail à réaliser, et de garantir son efficacité sur le long terme. Ainsi la planification d'une application passe par plusieurs étapes (qui peut dépendre d'une application à une autre) telle que :

- Étape n° 1 : **analyse fonctionnelle et définition des objectifs** : Cette partie consistera à rechercher et à caractériser les fonctions offertes par notre application pour satisfaire les différents besoins du client.
- Étape n° 2 : **conception détaillée** : La phase de conception détaillée donne lieu à la rédaction du cahier des charges opérationnel. C'est elle qui précisera les différents éléments de dimensionnement du projet.[1] C'est aussi dans cette étape qu'entre la modélisation du système avec des méthodes de modélisations comme MERISE.
- Étape n° 3 : **développement du projet** : C'est dans cette partie qu'entre en jeu la partie technique de l'application. Cette étape exige la maîtrise d'au moins un langage de programmation.

- Étape n° 4 : **Phase de tests** : C'est l'ensemble des tests (tests unitaires, tests d'intégration et tests de validation) qui permettront de retrouver les erreurs moins évidentes qui n'ont pas été détectées pendant la phase de développement.
- Étape n° 5 : **recette** : Permet de s'assurer que l'application développée correspond bien aux exigences fixées par le client.
- Étape n° 6 : **mise en production** : Déploiement de l'application.
- Étape n° 7 : **maintenance** : On entend par maintenance, l'ensemble des modifications mises en place après la mise en œuvre de l'application en production afin de corriger les bogues, améliorer les performances ou encore l'adapter à une modification de son environnement.[1]

1.4 Conclusion

Dans ce chapitre nous avons présenté **ONEMART** ainsi que ses différentes activités. Nous avons cadré le projet sur lequel tient notre mémoire en définissant la problématique, les objectifs, le cahier des charges et enfin le plan sur lequel se déroulera le développement de l'application. Le prochain chapitre sera dédié à la conception et à l'analyse des besoins de l'entreprise.

Chapitre 2

CONCEPTION ET ÉTUDE TECHNIQUE

2.1 Conception

Face à leur grandeur, le développement des systèmes d'information devient de plus en plus complexe. Prévoir les fonctionnalités d'un système d'information devient alors moins de en moins évidentes, c'est ainsi qu'entre en jeu la phase de conception. La phase de conception nécessite des outils permettant de mettre en place un modèle sur lequel on va s'appuyer pour réaliser notre application.

2.2 Etude comparative entre UML et MERISE

2.2.1 Présentation de MERISE

La méthode Merise (Méthode d'étude et de réalisation informatique pour les systèmes d'entreprise) est née à la fin des années 1970 en France, avec pour objectif de définir une démarche de conception de S.I. Le principe de base repose sur la séparation des données et des traitements. L'organisation des données semble plus pérenne que la définition des traitements, qui évolue en fonction de l'évolution des métiers, des fonctions et des utilisateurs. La méthode Merise intègre trois dimensions appelées cycles : le cycle d'abstraction, le cycle de vie et le cycle de décision. Le cycle de vie décrit les phases du projet de construction du S.I. du schéma directeur à la réalisation.[2]

2.2.2 Présentation de UML

UML est un langage de modélisation très complet, qui couvre de nombreux aspects du développement des logiciels, comme les exigences, l'architecture, les structures et les comportements.

Depuis sa normalisation, en 1997, UML a fortement évolué, passant d'un langage peu formel, principalement destiné à la documentation, à un langage suffisamment précis pour que des applications puissent être générées à partir des modèles. Cette évolution vers une plus grande précision a cependant créé une césure entre les tenants du « tout modèle », qui demandent toujours plus

de formalisme, et les développeurs, qui apprécient UML pour sa capacité à capturer en quelques dessins les grandes lignes d'une application.

2.2.3 Comparaison

MERISE est une méthode d'analyse et de réalisation des systèmes d'information qui est élaborée en plusieurs étapes : schéma directeur, étude préalable, étude détaillée et la réalisation.

Alors qu'UML, est un langage de modélisation des systèmes standard, qui utilise des diagrammes pour représenter chaque aspect d'un système c'est-à-dire : statique, dynamique en s'appuyant sur la notion d'orienté objet qui est un véritable atout pour ce langage.

2.2.4 Pourquoi le choix d'UML ?

Merise est encore tout à fait valable pour :

- La modélisation des données en vue de la construction d'une base de données relationnelles, la modélisation des processus métiers d'un système d'information automatisé en partie par du logiciel,
- La formalisation des besoins utilisateur dans la cadre de cahier des charges utilisateur, en vue de la conception d'un logiciel adapté.

UML est idéal pour :

- Concevoir et déployer une architecture logiciel développée dans un langage orienté objet (PHP, Java, C++, VB.Net,...).
- Pour modéliser les données (le modèle de classe réduit sans méthodes et stéréotypé en entités), mais avec des lacunes que ne présentait pas l'entité relation de Merise.
- Pour modéliser le fonctionnement métier (le diagramme d'activité et de cas d'utilisation) qui sont des formalismes très anciens qu'avait, en son temps, amélioré Merise...

Après cette étude comparative, il est certes que nous allons adopter UML comme langage de modélisation puisque nous allons utiliser le concept de l'orienté objet ainsi en PHP comme langage pour développer notre application.

2.3 Conception avec UML

2.3.1 Principaux diagrammes UML

UML dispose à sa version actuelle (2.5.1) de treize diagrammes qui sont regroupés en deux grandes catégories tels que les diagrammes structurels et les diagrammes de comportements.

Diagramme structurels

Ces diagrammes permettent de visualiser, spécifier, construire et documenter l'aspect statique ou structurel du système d'information. Voici quelques diagrammes utilisés couramment :

- Diagramme de classe : Ce diagramme représente la description statique du système en intégrant dans chaque classe la partie dédiée aux données et celle consacrée aux traitements. C'est le diagramme pivot de l'ensemble de la modélisation d'un système.[3]
- Diagramme de composant : Ce diagramme représente les différents constituants du logiciel au niveau de l'implémentation d'un système.[4]
- Diagramme de paquetage : Ce diagramme donne une vue d'ensemble du système structuré en paquetage. Chaque paquetage représente un ensemble homogène d'éléments du système (classes, composants. . .).[5]

Diagramme de comportements

Les diagrammes comportementaux modélisent les aspects dynamiques du système, c'est à dire les différents éléments qui sont susceptibles de subir des modifications.

2.3.2 Diagramme de cas d'utilisation

Un cas d'utilisation est une unité cohérente représentant une fonctionnalité visible de l'extérieur. Un cas d'utilisation modélise donc un service rendu par le système.[6]

Implémentation :

Acteurs :

- **Administrateur** : Acteur ayant pour rôle de gérer les comptes des autres utilisateurs de l'application.
- **Agent(Personnel)** : Agent de l'entreprise ayant accès à l'application depuis un point de vente agréé.
- **Valideur** : système externe qui récupère les messages des opérations effectuées et les valide par USSD. Le valideur peut être une application mobile ou un autre système.

Identification des cas d'utilisation :

- Authentification
- Gestion des comptes
- Gestion des intermèdes
- Gestion des numéros
- Gestion des opération
- Gestion des messages

Implémentation

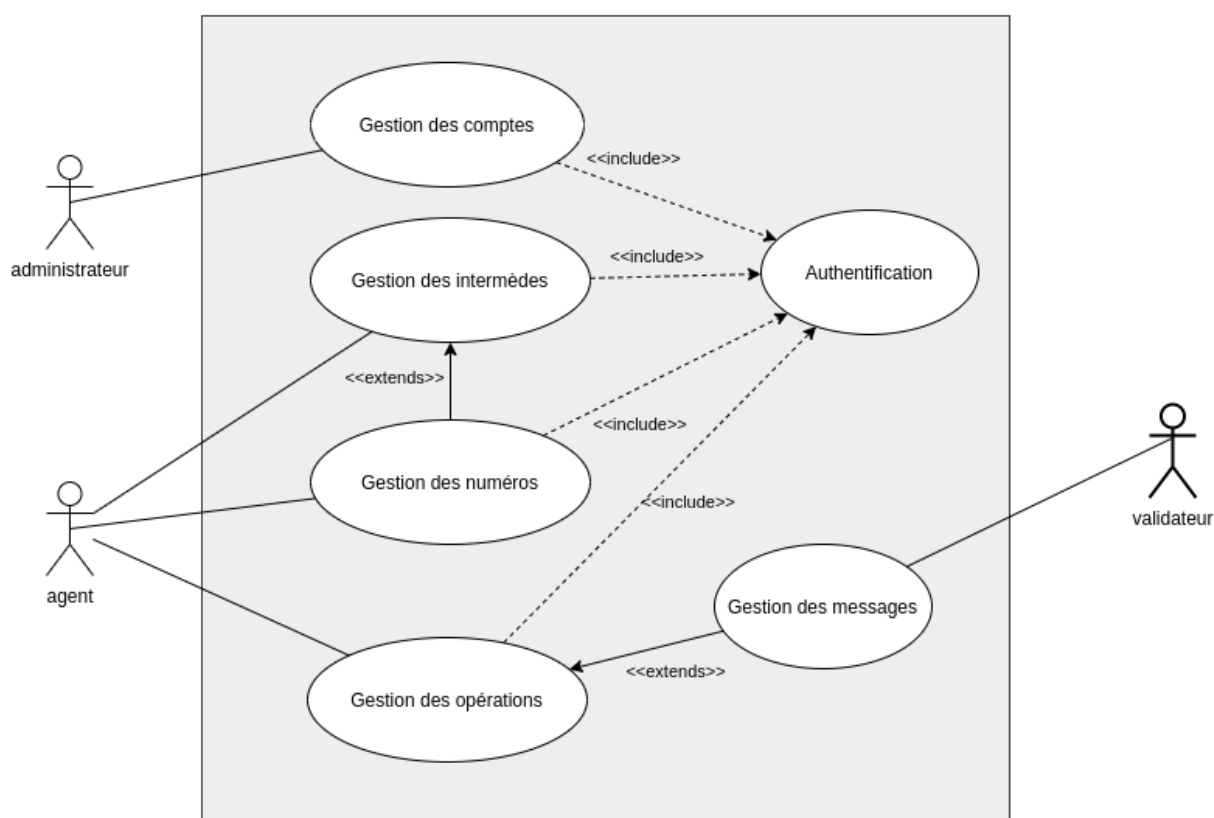


FIGURE 2.1 – Diagramme de cas d'utilisation : vue globale

Étude détaillée des cas d'utilisation

Authentification

	Description
Acteur principal	Agent de l'entreprise
Objectif	Se connecter pour avoir accès aux différentes fonctionnalités de l'application
Pré-conditions	Les informations sur l'agent doivent exister dans la base de données.
Déclencheur	Lancement de la page d'authentification
Scénario nominal	Affichage du formulaire de connexion contenant l'email et le mot de passe. L'agent renseigne ses identifiants et le système à son tour vérifie l'authenticité de ces informations. Si ces informations sont correctes alors l'agent est dirigé vers la page d'accueil.
Extensions	Si les identifiants envoyés sont incorrectes alors un message d'erreur lui est envoyé.

TABLE 2.1 – Description de l'authentification

Gestion des comptes

Dans cas d'utilisation, il revient à l'administrateur du système de créer les comptes des différents agents du système et leur attribuer un rôle à certaines pages de l'application en fonction de leurs rôles.

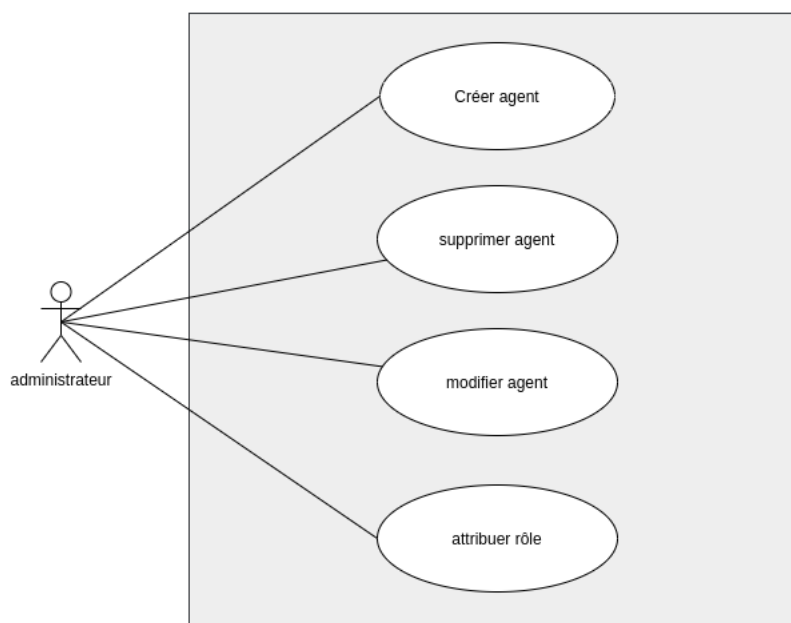


FIGURE 2.2 – Diagramme de cas d'utilisation : gestion des comptes

	Description
Acteur principal	Administrateur
Objectif	Enregistrer les agents de l'entreprise dans la base de donnée et attribuer à chacun d'eux un rôle.
Pré-conditions	Etre authentifié et avoir le rôle admin
Déclencheur	Lancement de la page d'authentification
Scénario nominal	Affichage du formulaire de connexion contenant l'email et le mot de passe. L'utilisateur saisit ses identifiants et le système à son tour vérifie l'authenticité de des données entrées. Si les informations entrées sont correctes alors l'agent est dirigé vers la page d'administration.
Extensions	Si les informations entrées sont incorrectes alors un message d'erreur lui est envoyé.

TABLE 2.2 – Description de cas d'utilisation : Gestion des compte

Gestion des intermèdes

Dans ce cas d'utilisateur, la tâche revient au personnel agent de gérer tout ce qui concerne les intermèdes.

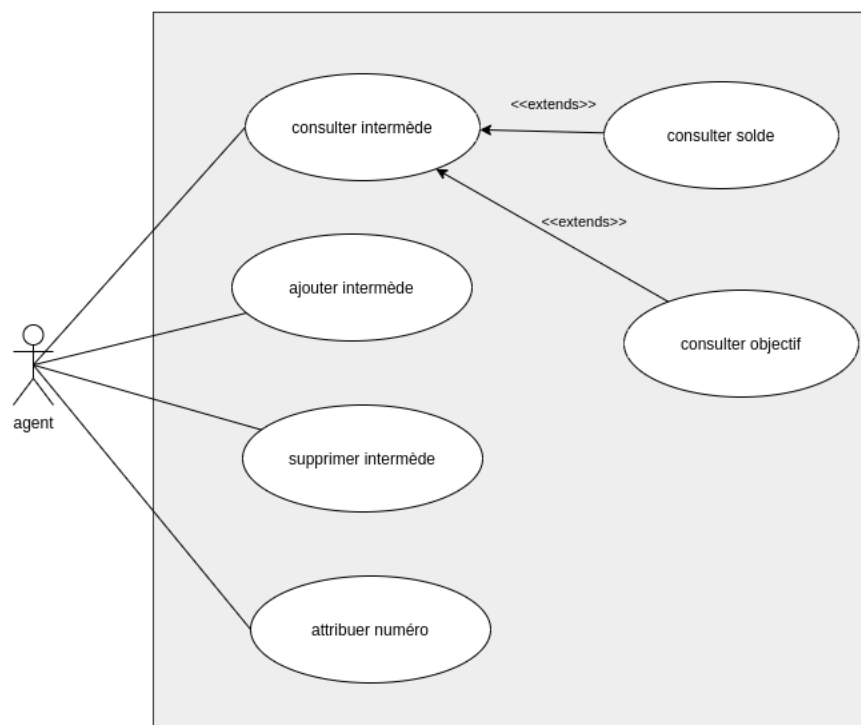


FIGURE 2.3 – Diagramme de cas d'utilisation : gestion des comptes

	Description
Acteur principal	Agent de l'entreprise
Objectif	Avoir la possibilité de gérer la liste des intermèdes dans le but de pouvoir l'étendre, réduire et mettre à jour des informations les concernant .
Pré-conditions	Être authentifié
Description	Une fois l'agent connecté, il a accès aux différentes pages qui lui permettront de gérer les intermèdes

TABLE 2.3 – Description du cas d'utilisation : Gestion des intermèdes

Gestion des numéros

	Description
Acteur principal	Agent de l'entreprise
Objectif	Mettre à jour les numéros des intermédiaires, créer de nouveaux numéros.
Pré-conditions	Avoir au moins un intermédiaire existant.
Déclencheur	Lancement de la page d'authentification.
Scénario nominal	Choisir l'intermédiaire. Saisir le numéro. Attribuer numéro. Enregistrer un numéro. Mettre à jour numéro. Supprimer numéro.
Extensions	Si aucun intermédiaire n'existe alors il sera impossible d'enregistrer un nouveau numéro.

TABLE 2.4 – Description du cas d'utilisateur : Gestion des numéros

Gestion des opérations

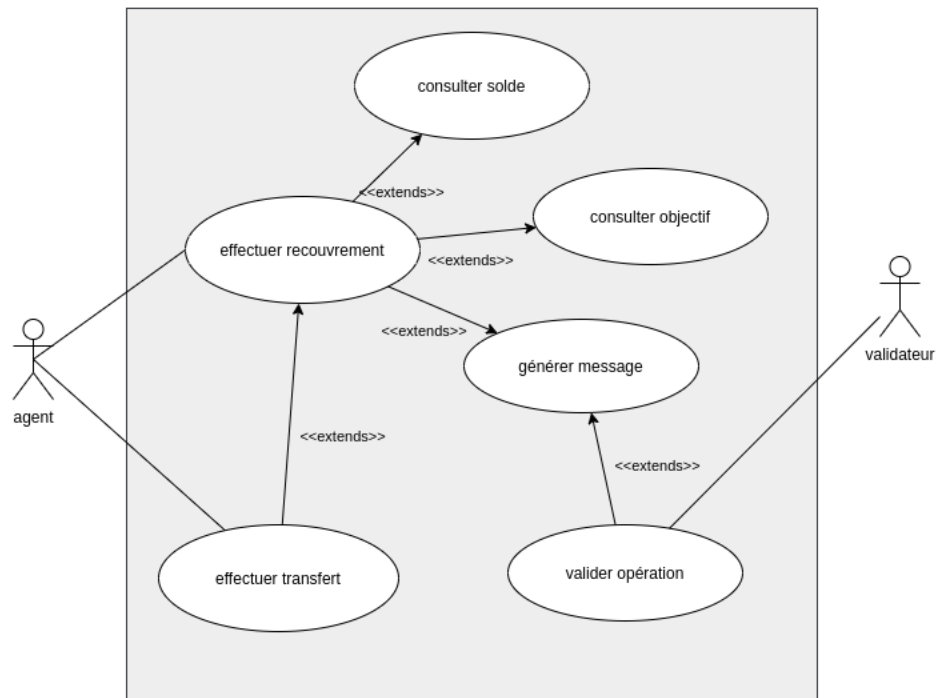


FIGURE 2.4 – Diagramme de cas d'utilisation : gestion des opérations

	Description
Acteur principal	Agent de l'entreprise et validateur
Objectif	Effectuer recouvrement(Dépôt) et transfert d'argent sur les numéros des intermédiaires
Pré-conditions	Être authentifié
Scénario nominal	<p>Sélectionner le grâce à son code ou son nom. Saisir le montant à transférer.</p> <p>Choisir le mode de paiement. Si le mode de paiement choisit correspond à un paiement bancaire, alors un nouveau champ apparaît pour saisir la référence bancaire. Dans le cas d'un recouvrement ou dépôt tous les modes de paiement de paiement sont autorisés sauf le mode de paiement à crédit.</p> <p>Lancer l'opération.</p> <p>Un message approuvant le qui l'opération a été effectuée est généré.</p> <p>Le validateur récupère le message et valide le opération de manière concrète grâce à la syntaxe appropriée par exemple il effectue : *413*NUMERO*MONTANT*00000# sachant bien que le NUMERO et le MONTANT sont contenus dans le message récupéré. Une fois l'opération validée concrètement, le validateur renvoie reçu. Impression de la facture.</p>
Extensions	Si les informations envoyées sont incorrectes alors un message d'erreur est affiché

TABLE 2.5 – Description du cas d'utilisateur : Gestion des opérations

Gestion des messages

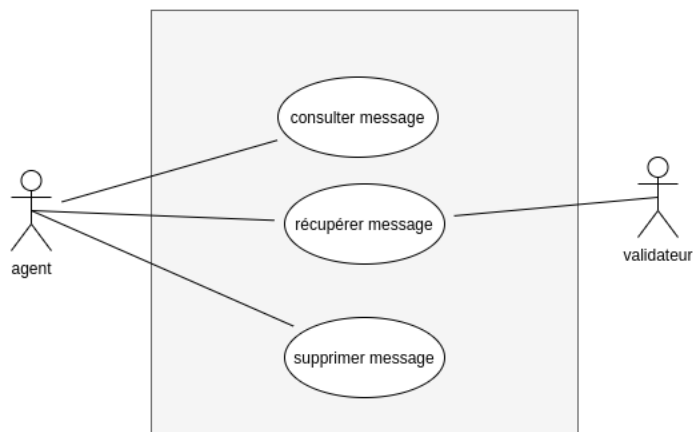


FIGURE 2.5 – Diagramme de cas d'utilisation : gestion des messages

	Description
Acteur principal	Agent et validateur
Objectif	Traiter les messages générés après chaque opération.
Pré-conditions	. Être authentifié
Déclencheur	Opération effectuée
Scénario nominal	<p>Une fois qu'une opération est effectuée, un nouveau message est généré avec l'état initié.</p> <p>Une fois que le message est récupéré, son état du message passe à l'état 'en cours d'exécution'.</p> <p>Enfin, lorsque l'opération est finalement validée, alors l'état du message passe à clôturer</p>

TABLE 2.6 – Description de l'authentification

2.3.3 Diagramme de classe

Définition de classe

Une classe est un ensemble de données et de fonctions regroupées dans une même entité. Une classe est une description abstraite d'un objet. Les fonctions permettant de manipuler la classe sont appelées méthodes. Instancier une classe consiste à créer un objet sur son modèle.

Représentation d'une classe

Une classe est représentée par rectangle subdiviser en trois compartiments dans laquelle :

- Le premier compartiment contient le nom de classe
- Le deuxième contient la liste des attributs de classe ainsi que leur visibilité (public, protégée ou privée).
- Le troisième compartiment représente la liste des méthodes permettant de manipuler les différents attributs de classe.

En gros une classe se présente comme le montre la figure suivante :

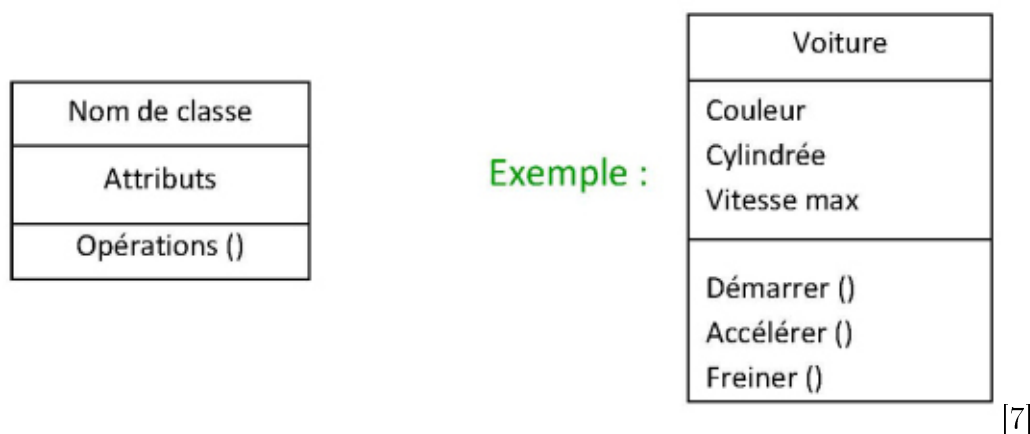


FIGURE 2.6 – représentation d'une classe

Les attributs

Les attributs représentent les données qui caractérisent l'objet. Les attributs sont aussi définis par leurs types (entier, chaîne de caractère, caractère, etc.).

Les méthodes

Les méthodes d'un objet caractérisent son comportement, c'est-à-dire l'ensemble des actions (appelées opérations) que l'objet est à même de réaliser.

Notion de multiplicité

La multiplicité définit le nombre d'instances de l'association pour une instance de la classe. La multiplicité est définie par un nombre entier ou un intervalle de valeurs, elle est aussi la traduction d'une règle de gestion.

1	Un et un seul
0..1	Zéro ou un
N ou *	N (entier naturel)
M..N	De M à N (entiers naturels)
0..*	De zéros à plusieurs
1..*	De 1 à plusieurs

FIGURE 2.7 – multiplicité

Les associations

Les associations permettent de préciser les relations qui peuvent exister entre deux ou plusieurs objets. L'association se fait entre classe et non entre les instances. Lorsqu'une association est définie entre deux classes, cela signifie que les objets instances de ces deux classes peuvent être reliés entre eux.

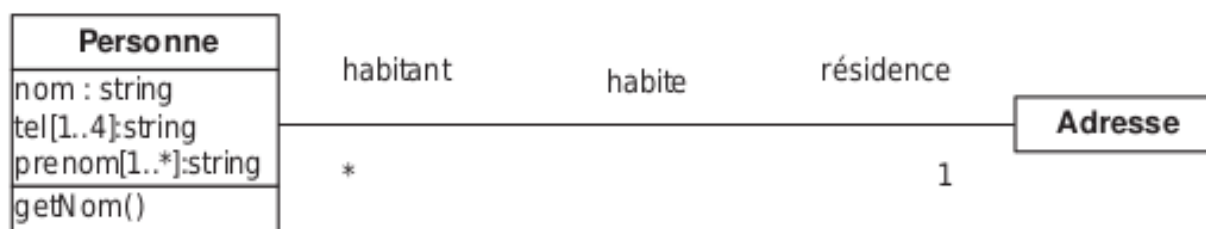


FIGURE 2.8 – association

implémentation

Identification des classes :

Personnel(ID :integer, nom :string, prenom :string, email)

Intermed(ID :integer, code :string, nom :string, solde :integer)

Numero(ID :integer, numero :string)

Objectif(ID :integer, mois :string, objectif :integer, bilan :integer)

Operation(ID :integer, montant :integer, reference :string, date)

Transfert(ID :integer, numero :string)

Secteur(ID :integer, nom :string)

Agence(ID :integer, nom :string)

Statut(ID :integer, nom :string)

Etat(ID :integer, nom :string, commentaire :text)

Message(ID :integer, sms :text, date :Date)

Facture(ID :integer, numeroFac :string, date :Date)

Mode de paiement(ID :integer, nom :string)

Ouverture(ID :integer, jour :string, code :string)

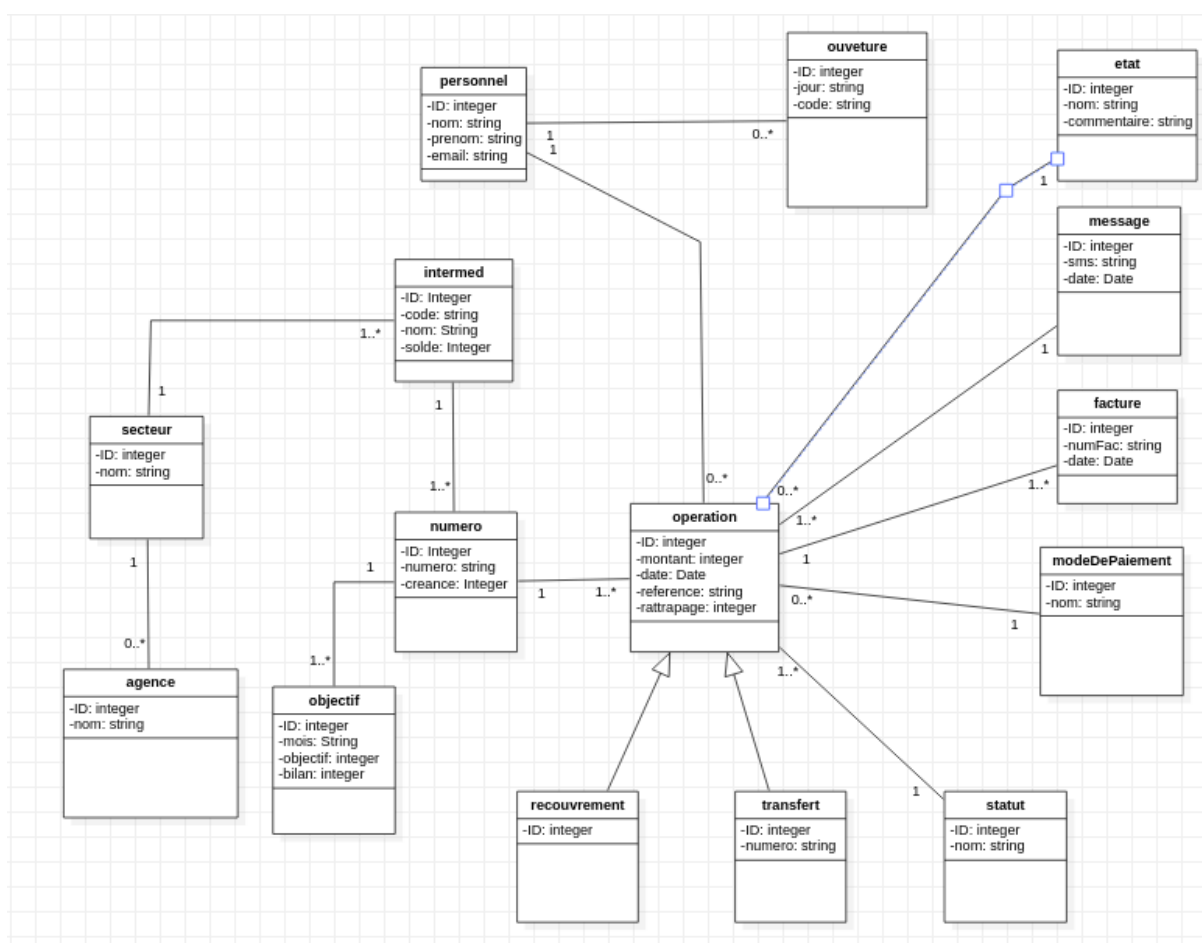


FIGURE 2.9 – diagramme de classe de l'application

Description des classe

Classe Peronnell : représente les effectuant les opérations			
	Nom	Description	Type
Attributs	ID	code permettant d'identifié le personnel	Numérique
	nom	désigne le nom personnel	chaîne de caractère
	prénom	désigne le prénom de personnel	chaîne de caractère
	email	adresse électronique de l'intermed	chaîne de caractère

TABLE 2.7 – tableau de classe *Personnel*

Classe Intermed : représente les effectuant les opérations			
	Nom	Description	Type
Attributs	ID	code permettant d'identifié l'intermed	numérique
	code	code unique permettant d'identifier l'intermed	chaîne de caractère
	nom	désigne le nom l'intermède	chaîne de caractère
	solde	désigne le solde l'intermède	numérique

TABLE 2.8 – tableau de classe Intermed

Classe Numéro : numéro appartenant à un intermède			
	Nom	Description	Type
Attributs	ID	permet d'identifier le numéro	numérique
	numéro	numéro de l'intermède	chaîne de caractère
	créance	créance de l'intermède sur ce numéro	numérique

TABLE 2.9 – tableau de classe *Numéro*

Classe Objectif : représente les effectuant les opérations			
	Nom	Description	Type
Attributs	ID	identifiant de l'objectif d'un numéro	numérique
	mois	mois dans lequel l'objectif à été effectué	chaîne de caractère
	objectif	objectif du numéro	chaîne de caractère
	bilan	bilan de l'objectif	numérique

TABLE 2.10 – tableau de classe *objectif*

Classe Opération : désigne les opérations pouvant être par le personnel			
	Nom	Description	Type
Attributs	ID	identifiant de l'opération	numérique
	montant	montant à transféré ou à déposer	numérique
	date	date à laquelle l'opération s'est effectuée	numérique
	référence	chaîne de caractère pouvant identifier l'opération	chaîne de caractère
	rattrapage	entier permettant de spécifier si l'opération s'est bien déroulée	entier

TABLE 2.11 – tableau de classe *Opération*

Classe État : définie l'ensemble des statut que peut prendre une opération			
	Nom	Description	Type
Attributs	ID	code permettant d'identifié l'intermède	numérique
	nom	nom du statut qui peut prendre les valeurs suivantes : <i>initié</i> , <i>en cours d'exécution</i> , <i>exécuté</i> , <i>annulé</i> et <i>supprimé</i>	chaîne de caractère

TABLE 2.12 – tableau de classe *Opération*

Classe Message : message de transfert après qu'une opération soit effectuée			
	Nom	Description	Type
Attributs	ID	identifiant du message	numérique
	sms	message de transfert	texte
	date	date à laquelle le message a été reçu	date

TABLE 2.13 – tableau de classe *message*

Classe transfert : classe fille de la classe opération spécialisée dans le transfert			
	Nom	Description	Type
Attributs	ID	permet d'identifier le transfert effectué	numérique
	numéro	numéro sur lequel le transfert sera effectué	chaîne de caractère

TABLE 2.14 – tableau de classe *transfert*

Classe recouvrement : classe spécialisée dans le recouvrement			
	Nom	Description	Type
Attributs	ID	permet d'identifier le recouvrement effectué	numérique

TABLE 2.15 – tableau de classe *recouvrement*

Classe Facture : facture délivrée après une opération			
	Nom	Description	Type
Attributs	ID	identifiant permettant de retrouver la facture	numérique
	numFac	numéro unique permettant d'identifier la facture	chaîne de caractère
	date	date à laquelle la facture a été délivrée	chaîne de caractère

TABLE 2.16 – tableau de classe *facture*

Classe ModePaiement : représente les effectuant les opérations			
	Nom	Description	Type
Attributs	ID	identifiant unique attribué à un mode paiement	numérique
	nom	nom du mode de paiement	chaîne de caractère

TABLE 2.17 – tableau de classe *ModePaiement*

Classe ouverture : représente les effectuant les opérations			
	Nom	Description	Type
Attributs	ID	identifiant des jours d'ouverture	numérique
	jour	jour d'ouverture	chaîne de caractère
	code	code d'ouverture	chaîne de caractère

TABLE 2.18 – tableau de classe *ouverture*

Classe agence : définie l'agence			
	Nom	Description	Type
Attributs	ID	identifiant permettant d'identifier l'agence	numérique
	nom	désigne le nom de l'agence	chaîne de caractère

TABLE 2.19 – tableau de classe *agence*

Classe Secteur : secteur au lequel appartient l'agence			
	Nom	Description	Type
Attributs	ID	identifiant du secteur	numérique
	nom	nom du secteur	chaîne de caractère

TABLE 2.20 – tableau de classe *secteur*

2.4 Schéma relationnel

2.4.1 Modèle logique des données ou MLD

Définition

Le modèle logique des données consiste à décrire la structure de données utilisée sans faire référence à un langage de programmation. Il s'agit donc de préciser le type de données utilisées lors des traitements. Ainsi, le modèle logique est dépendant du type de base de données utilisé.

Liste des tables

PERSONNEL(ID, nom, prenom, email)

SECTEUR(ID, nom)

AGENCE(ID, #secteur_id, nom)

INTERMED(ID, code, nom, solde)

NUMERO(ID, #intermed_id, numero, creance)

objectif(ID, #numero_id, mois, objectif, bilan)

MODEDEPAIEMENT(ID, nom)

STATUT(ID, nom)

ETAT(ID, nom)

OPERATION(ID, #personnel_id, #numero_id, #modepaiement_id, #statut_id, #etat_id, montant, date, reference, rattrapage)

OUVERTURE(ID, #personnel_id, jour, code)

TRANSFERT(ID, #personnel_id, #numero_id, #modepaiement_id, #statut_id, #etat_id, rattrapage, montant, reference, date)

RECOUVREMENT(ID, #personnel_id, #modepaiement_id, #statut_id, #etat_id, rattrapage, montant, reference, date)

FACTURE(ID, #operation_id, numFac, date)

MESSAGE(ID, #operation_id, sms, date)

2.4.2 Modèle physique de donnée ou MPD

Définition

Un modèle de données physique est un modèle spécifique à une base de données, qui représente des objets de données relationnelles (par exemple, tables, colonnes, clés principales et externes) et leurs relations.

Représentation du MPD

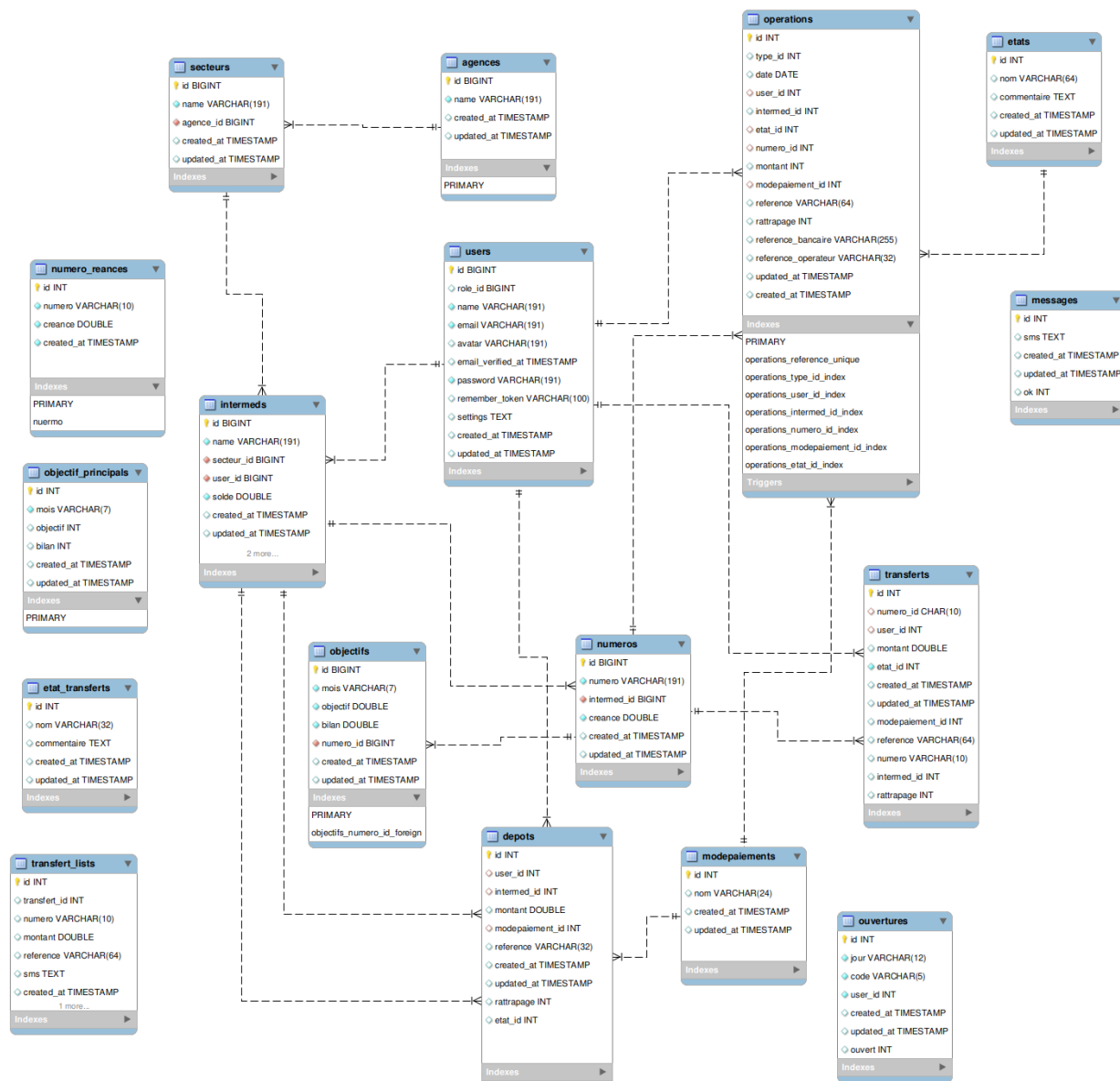


FIGURE 2.10 – Modèle physique des données

2.5 Outils de développement

2.5.1 Technologie front-end

HTML 5

L'HyperText Markup Language, HTML, désigne un type de langage informatique descriptif. Il s'agit plus précisément d'un format de données utilisé dans l'univers d'Internet pour la mise en forme des pages Web. Il permet, entre autres, d'écrire de l'hypertexte, mais aussi d'introduire des ressources multimédias dans un contenu.

Développé par le W3C (World Wide Web Consortium) et le WHATWG (Web Hypertext Application Technology Working Group), le format ou langage HTML est apparu dans les années 1990. Il a progressivement subi des modifications et propose depuis 2014 une version HTML5 plus aboutie.

L'HTML est ce qui permet à un créateur de sites Web de gérer la manière dont le contenu de ses pages Web va s'afficher sur un écran, via le navigateur.[8]



FIGURE 2.11 – Logo HTML 5

CSS 3

Les feuilles de styles (en anglais "Cascading Style Sheets", abrégé CSS) sont un langage qui permet de gérer la présentation d'une page Web. Le langage CSS est une recommandation du World Wide Web Consortium (W3C), au même titre que HTML ou XML. Aujourd'hui nous sommes à la version 3 d'où CSS 3.



FIGURE 2.12 – Logo CSS 3

Bibliothèque CSS - Bootstrap 5

Bootstrap est une bibliothèque CSS permettant de donner une apparence sublime aux sites web. L'une de ces forces est son côté responsive qui permet au site de s'adapter sur les grands et petits écrans (mobile)



FIGURE 2.13 – Logo Bootstrap 5
[9]

Bibliothèque JavaScript - AlpineJS

Alpine est un outil robuste et minimal pour composer des comportements directement dans notre code HTML. Il apporte une réactive flexible comme d'autre bibliothèque JS (React par exemple).

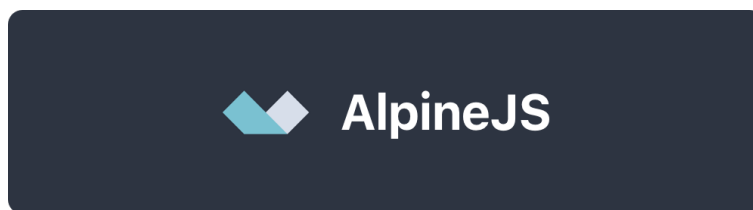


FIGURE 2.14 – Logo AlpineJS
[10]

2.5.2 Choix du framework back-end

Laravel



FIGURE 2.15 – Logo de Laravel
[11]

Dans cette application, nous avons décidé de nous tourner vers l'un des framework PHP le plus populaire et le plus utilisé : **Laravel**.

En effet Laravel est le framework PHP open source le mieux noté sur GitHub. Fondé en 2011 par Taylor Otwell, Laravel utilise le pattern MVC et est orienté objet.[12]

Que de réinventer la roue, Laravel se base sur des technologies existantes.

Laravel se base sur **Symfony** pour créer son système de routage. De même pour l'envoi des mails Laravel utilise la bibliothèque **SwiftMailer**.

Avantage

- Il est facile à installer et présent chez tous les hébergeurs.
- Basé sur l'architecture MVC(Model View Controller)qui donne une meilleure organisation au niveau du code et avec une hiérarchie de dossiers très bien fait.
- Grande communauté et beaucoup de documentation
- Possibilité de faire tests unitaires afin de garantir qu'il n'y a pas de bogues ou d'exceptions l'application
- Développement plus rapide grâce aux helpers déjà disponible.

Livewire



FIGURE 2.16 – Logo de Livewire
[13]

Présentation

Livewire est un package Laravel, qui aux permet développeur Laravel de créer des applications Web réactives en utilisant Laravel Blade comme langage de templating sans écrit la moindre ligne de code.

Avantages

- Créer des application réactive en restant dans le confort Laravel sans écrit la moindre ligne de code JavaScript.
- Crée facilement des composants réutilisables
- Plusieurs fonctions (ou encore **helper**) disponible pour rendre encore plus facile sa prise en main.

2.5.3 Système de gestion de base de données

MySQL



FIGURE 2.17 – Logo de MySQL
[14]

Et surtout MySQL, qui est un Système de Gestion de Bases de Données Relationnelles (abrégé SGBDR), c'est-à-dire un logiciel qui permet de gérer des bases de données, et donc de gérer de grosses quantités d'informations. Il utilise pour cela le langage SQL. Il s'agit d'un des SGBDR les plus connus et les plus utilisés.

2.5.4 Avantages

- Grâce à sa popularité, MySQL dispose d'une grande communauté, ce qui implique qu'on peut facilement trouver de l'aide en cas de difficulté.
- il est totalement open source et gratuit.
- il est en plus multi-threadé et multi-utilisateurs.
- Ses performances sont excellentes

2.5.5 Architecture structurel

MVC

Modèle-vue-contrôleur ou MVC est un motif d'architecture logicielle destiné aux interfaces graphiques lancé en 1978 et très populaire pour les applications web. Le motif est composé de trois types de modules ayant trois responsabilités différentes : les modèles, les vues et les contrôleurs.

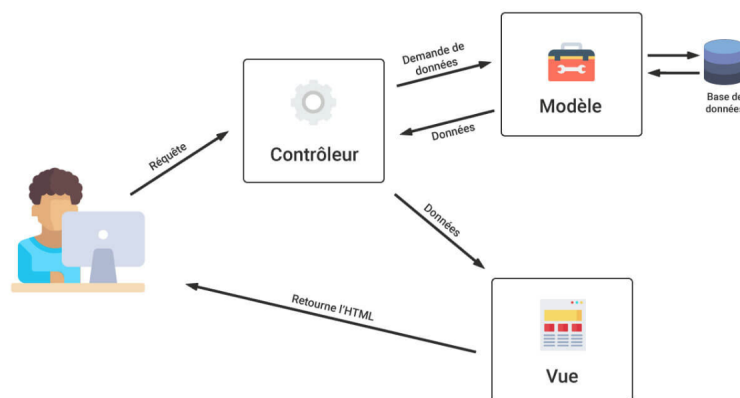


FIGURE 2.18 – MVC
[15]

Model

cette partie gère les données de l'application. Son rôle est d'aller récupérer les informations « brutes » dans la base de données, de les organiser et de les assembler pour qu'elles puissent ensuite être traitées par le contrôleur. On y trouve donc entre autres les requêtes SQL[16]. Dans une application laravel le Model est géré par Eloquent qui est Object Relational Mapping.

View

Cette partie se concentre sur l'affichage. Elle ne fait presque aucun calcul et se contente de récupérer des variables pour savoir ce qu'elle doit afficher. On y trouve essentiellement du code HTML mais aussi quelques boucles et conditions PHP très simples.

Controller

Cette partie gère la logique du code qui prend des décisions. C'est en quelque sorte l'intermédiaire entre le modèle et la vue : le contrôleur va demander au modèle les données, les analyser, prendre des décisions et renvoyer le texte à afficher à la vue.

2.5.6 Autre outils utilisés

MySQL Workbench



FIGURE 2.19 – MVC
[17]

MySQL Workbench est un logiciel de gestion et d'administration de bases de données MySQL.

Draw.io



FIGURE 2.20 – Draw.io
[18]

Draw.io est une application gratuite en ligne, accessible via son navigateur (protocole https) qui permet de dessiner des diagrammes ou des organigrammes. Cet outil vous propose de concevoir toutes sortes de diagrammes, de dessins vectoriels, de les enregistrer au format XML puis de les exporter.[19]

Chapitre 3

PRÉSENTATION DE L'APPLICATION

3.1 Authentification

Pour contrôler l'accès aux ressources dans notre application, nous avons mis en place un système d'authentification qui permettra des informations sur chaque personne connectée.



The screenshot displays the login interface for 'Onemart'. At the top, the logo consists of a globe icon and the text 'Onemart la clé de la distribution'. Below this, the form includes fields for 'E-MAIL' and 'PASSWORD', a 'REMEMBER ME' checkbox, and a blue 'Login' button. The background is a collage of images: a globe, a map, and two people. A copyright notice 'Copyrights © All Rights Reserved by VZ Technology' is visible at the bottom.

E-MAIL

E-mail

PASSWORD

Password

☐ REMEMBER ME

Login

Copyrights © All Rights Reserved by VZ Technology

FIGURE 3.1 – Page d’authentification

3.2 Liste des agents

Dans cette page on peut voir la liste des agents autorisés à accéder à l’application.

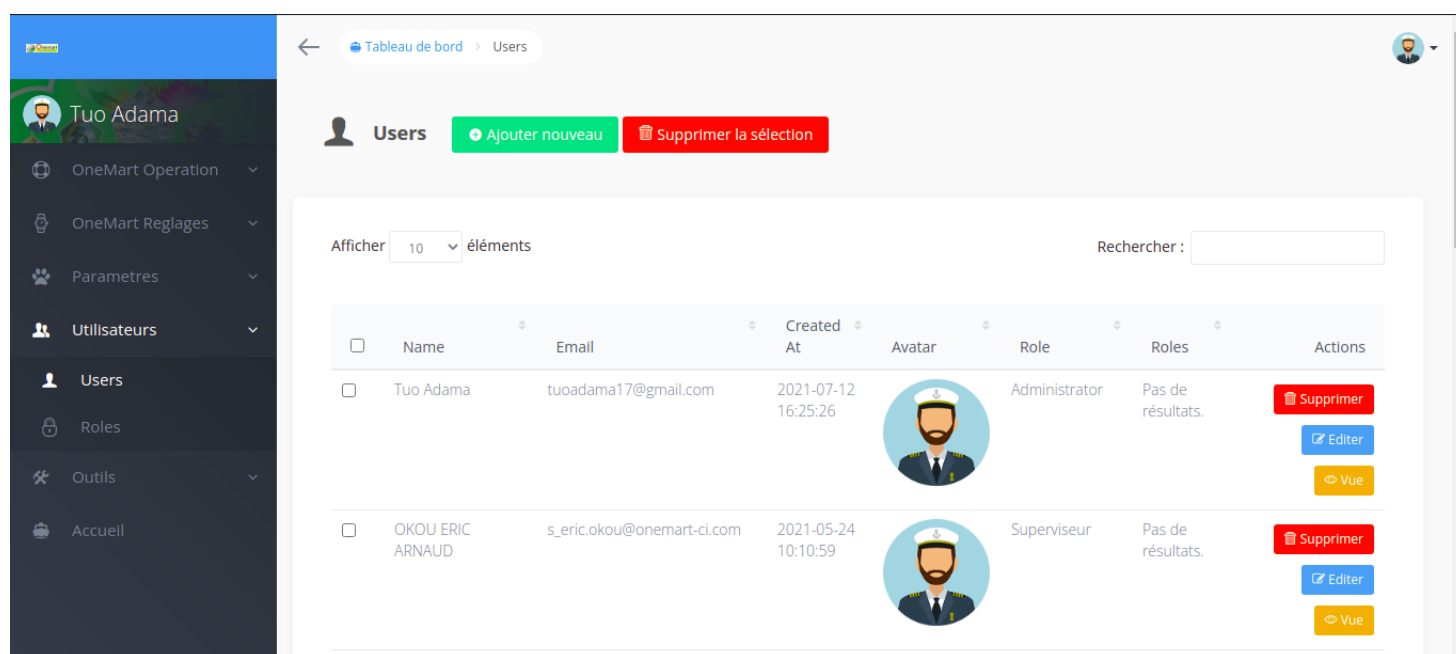


FIGURE 3.2 – Liste des agents

3.3 Ajouter d’un agent

Page permettant ajouter des agents tout en ayant la possibilité de leurs attribuer des rôles.

FIGURE 3.3 – Ajouter un agent

3.4 Les Rôles

Pour définir une restriction à l'accessibilité aux ressources de l'application, on attribue un ou plusieurs rôle à un utilisateur.

	Name	Display Name	Actions
<input type="checkbox"/>	OneMart Admin	Administration OneMart	<button>Vue</button> <button>Editor</button> <button>Supprimer</button>
<input type="checkbox"/>	Caisse	Caissiers, Caissières	<button>Vue</button> <button>Editor</button> <button>Supprimer</button>
<input type="checkbox"/>	Superviseur	Superviseur	<button>Vue</button> <button>Editor</button> <button>Supprimer</button>
<input type="checkbox"/>	admin	Administrator	<button>Vue</button> <button>Editor</button> <button>Supprimer</button>
<input type="checkbox"/>	user	Normal User	<button>Vue</button> <button>Editor</button> <button>Supprimer</button>

FIGURE 3.4 – Les rôles

3.5 Ajouter intermède

État civil

Etat civil	Les numeros	Dossier	Finalisation
Nom <input type="text" value="GSM"/>	Prenom <input type="text" value="UNITS DEALER 2 BEL"/>	Société <input type="text" value="SOCIÉTÉ"/>	Secteur <input type="text" value="YOP SIPOREX"/>
Localisation <input type="text" value="LOCALISATION"/>	Telephone <input type="text" value="TELEPHONE"/>	Superviseur <input type="text" value="GNAKAN GERMAIN KOUASSI"/>	<input type="button" value="Suivant"/>

FIGURE 3.5 – Ajouter intermède

Les numéros


Etat civil	Les numeros	Dossier	Finalisation
<p>Etat civil</p> <p>Code : 2</p> <p>Nom : GSM UNITS DEALER 2 BEL</p> <p>Secteur : YOP SIPOREX</p> <p>Numeros</p> <p>0103190877 </p>	<p>Ajout de numeros</p> <p><input type="text"/></p>	<p><input type="button" value="Suivant"/></p>	

FIGURE 3.6 – Ajout de numéros

Photo de profil de l'intermède

FIGURE 3.7

3.6 Liste des intermèdes

Dans cette on affiche tous les intermèdes avec leur solde.

Code	Nom et prenom	Numeros	Secteur	Solde	Superviseur	
1	GSM UNITS DEALER	0102555978	YOP SIPOREX	3000000	GNAKAN GERMAIN KOUASSI	Modifier Transfert Rvt
2	GSM UNITS DEALER 2 BEL	0103190877	YOP SIPOREX	0	GNAKAN GERMAIN KOUASSI	Modifier Transfert Rvt
3	KONE SEYDOU KADER	0102582982	YOP MOSSIKRO SANTE 3	0	NGORAN KOUAKOU YVES ROLAND	Modifier Transfert

FIGURE 3.8 – Liste des intermèdes

3.7 Les modes de paiements

Cette section concerne les modes de paiements. Nous avons mis en place un système qui pourra s'adapter dans le temps car elle permettra d'ajouter si on le veut d'autre mode de paiement.

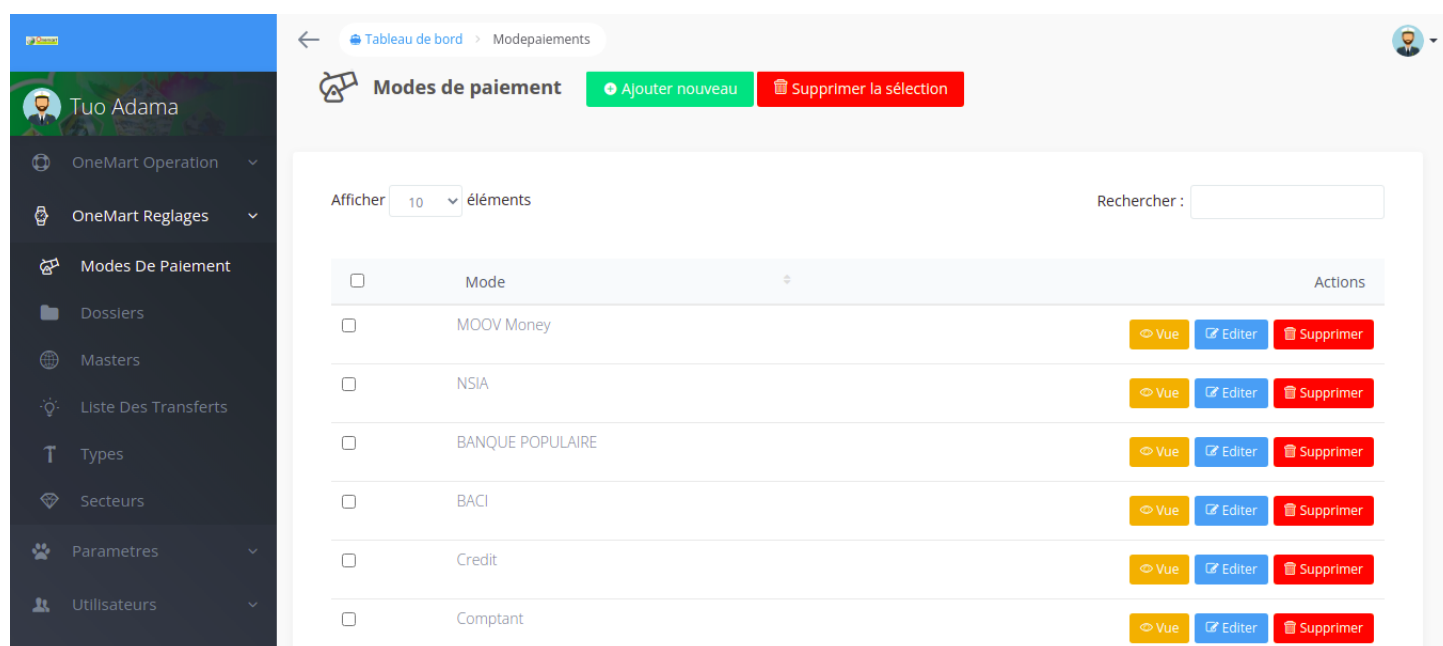


FIGURE 3.9 – Mode de paiement

3.8 Liste des types

Dans cette partie également on liste l'ensemble d'opérations que peut effectuer l'agence pour que plus tard on puisse ajouter d'autre nouvelle fonctionnalité.

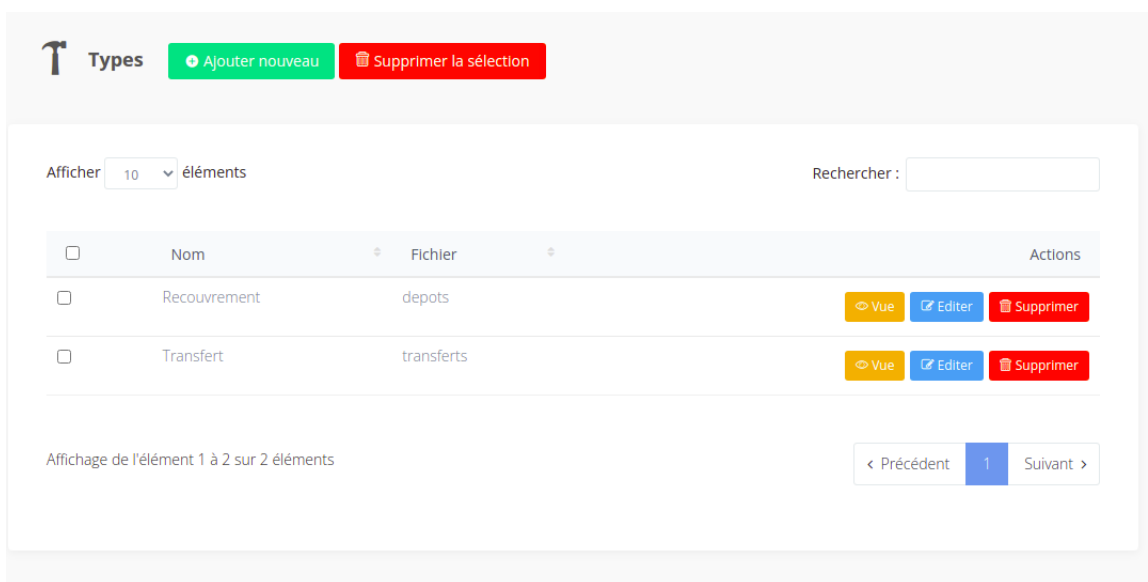


FIGURE 3.10 – Les types

3.9 Recouvrement

Le recouvrement ou encore dépôt se sert des informations présentes sauf que le mode de paiement à crédit n'est pas accessible.

The screenshot shows a 'Recouvrement' form. At the top, there's a header with a 'Recouvrement' title and a back arrow. Below this, a 'Filtre' section contains a 'Code' input field with the value '10' and an 'Intermed' dropdown menu showing 'KOUAKOU KOUADIO CHRISTOPHE'. The main form area has a 'Recouvrement' section with a text input field containing 'KOUAKOU KOUADIO CHRISTOPHE'. To the right of this field are two input fields: 'Montant' with the value '30000' and 'Mode de paiement' with a dropdown menu showing 'Comptant'.

FIGURE 3.11 – Recouvrement

FIGURE 3.12 – Details Recouvrement

FIGURE 3.13 – Confirmation recouvrement

3.10 Transfert

Bien que l'interface de transfert soit similaire à celle du recouvrement, dans le l'opération de transfert on a accès à tous les modes de paiements.

FIGURE 3.14 – Transfert

FIGURE 3.15 – Detail du transfert

3.11 Facture après une opération

Après chaque opération, une facture est délivré à l'intermède. Cette facture se présente comme suite :

Onemart
 01 BP 11980 Abidjan 01
 27 23 45 73 09 / 01 01 02 45 55
 info@onemart-ci.com
 2021/10/06 09:55

N°Transfert-20211006-393

N°Client 2

GSM UNITS DEALER 2 BEL

RECU

E-intermed	Montant
HT	100 122
TVA	22 878
TTC	123 000

Payé et Livré

Tot. Créance 30 000 :

Caisse
 CAISSE-25
 TUO ADAMA

Votre objectif de Octobre est :0
Vous êtes à 0 soit -%
-SOYEZ DANS LE MOOV-

FIGURE 3.16 – Présentation de facture

3.12 Message des opérations

Une fois qu'une opération est effectuée, on a un message qui est généré.

<input type="checkbox"/>	SMS	Ok	Actions
<input type="checkbox"/>	Vous avez transfere 200 000 Fcfa au numero 0140269317. Votre solde actuel est 178 374 718 Fcfa. Ref 01787016875	OK	Editier Supprimer Vue
<input type="checkbox"/>	Vous avez transfere 420 000 Fcfa au numero 0142880309. Votre solde actuel est 191 639 718 Fcfa. Ref 01786431259	OK	Editier Supprimer Vue
<input type="checkbox"/>	Vous avez transfere 200 000 Fcfa au numero 0140269344. Votre solde actuel est 192 059 718 Fcfa. Ref 01786393875	OK	Editier Supprimer Vue
<input type="checkbox"/>	Vous avez transfere 420 000 Fcfa au numero 0142880309. Votre solde actuel est 191 639 718 Fcfa. Ref 01786431259	OK	Editier Supprimer Vue
<input type="checkbox"/>	Vous avez transfere 500 000 Fcfa au numero 0140399381. Votre solde actuel est 192 259 718 Fcfa. Ref 01786390399	OK	Editier Supprimer

FIGURE 3.17 – Message des opérations


<div>  Liste des Transferts </div> <div> Ajouter nouveau Supprimer la sélection </div>						
Afficher		10	éléments		Rechercher : <input type="text"/>	
<input type="checkbox"/>	Transfert Id	Numero	Montant	Reference	Sms	Actions
<input type="checkbox"/>	26	0103778778	130	01749366060	Vous avez transfere 130 Fcfa de credit vers le numero 0103778778. Votre solde actuel est 6 830 Fcfa. Ref 01749366060	Supprimer Vue Editier
<input type="checkbox"/>	25	0103778778	120	01749364420	Vous avez transfere 120 Fcfa de credit vers le numero 0103778778. Votre solde actuel est 6 960 Fcfa. Ref 01749364420	Supprimer Vue Editier

FIGURE 3.18 – Message des transferts

3.13 États des paiements

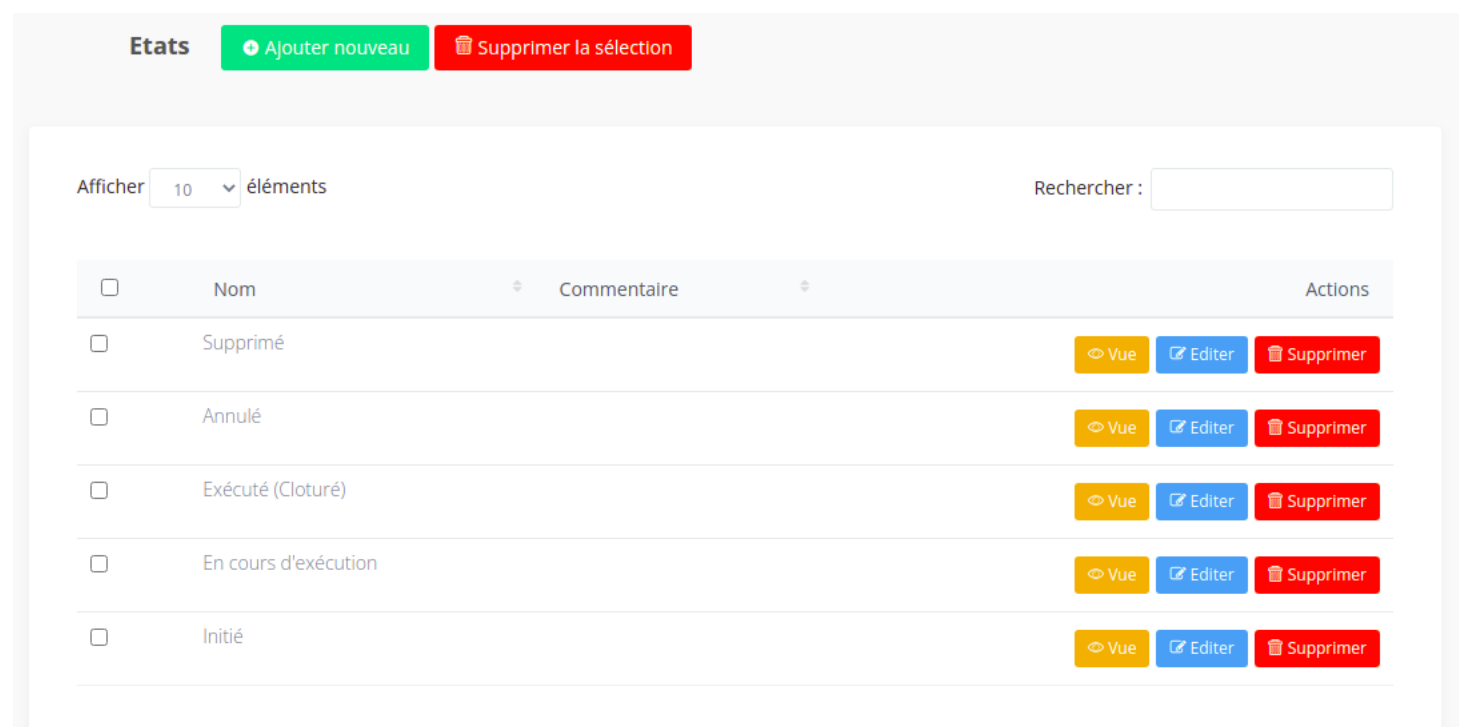


FIGURE 3.19 – Etat de paiement

3.14 Conclusion

Dans ce chapitre nous avons exploré les grandes parties de notre application grâce aux différentes captures d'écrans et quelque détails pour plus de compréhension.

Conclusion générale

Dans le monde du développement informatique, pour mettre en place une application qui puisse être maintenable, il est important de suivre un chemin logique qui part de la conception jusqu'au choix des outils techniques pour la réalisation du projet.

C'est à partir de ce raisonnement que nous nous sommes posés de nombreuses questions quant au choix de la méthode de modélisation, de l'outil de développement, etc, pour mener à bien la réalisation de ce projet. Ce projet a été inspiré d'une application legacy (une vieille application) qui ne pouvait plus être maintenue par le fait de la non disponibilité de son code source et aussi de l'architecture sur laquelle elle reposait. C'est à partir de ce fait, que nous nous sommes fixés comme objectif de fournir une application qui bien évidemment basée sur une architecture simple pourra s'adapter aux besoins de l'entreprise et fournir plus de fonctionnalités dans le but de faciliter les différentes activités menées au sein de l'entreprise ONEMART.

La réalisation de cette application nous a été bénéfique sur plusieurs points. Elle nous a permis de comprendre le sens réel de la pratique dans le domaine informatique car grâce à la pratique, nous avons touché à tout ce qui est API Rest, faire le tour du développement WEB et sans oublier la structuration de code que nous avons tirée de l'architecture MVC.

En réalité, en partant de la conception jusqu'à la réalisation, le développement de cette application nous a également permis de nous perfectionner en améliorant nos connaissances en programmation informatique et en conception. Il est aussi important de noter que grâce à ce projet, nous avons compris et mis en œuvre le déroulement d'un cycle de vie d'un logiciel.

A l'issue de notre stage, nous d'autres idées d'amélioration nous sont venues en tête qui consiste à la mise en application d'une application mobile pour chaque agent l'entreprise.

C'est ainsi que nous nous posés la question principale suivante :

A l'aide des points d'entrées définies, comment mettre en place une application mobile pouvant effectuer les opérations de rechargements et transferts sans disposée de la puce e-master ?

Bibliographie

- [1] Samantha Mur. Comment gerer un projet informatique? <https://www.appvizer.fr/magazine/operations/gestion-de-projet/gestion-projet-informatique>. 18/12/2021.
- [2] www.universalis.fr. Merise. https://www.universalis.fr/encyclopedie/systemes-informatiques-conception-architecture-et-urbanisation-des-systemes-d-informatique/#i_49502.
- [3] memoireonline. Diagramme de classe. https://www.memoireonline.com/01/17/9565/m_Application-de-gestion-commerciale-des-produits-alimentaires13.html.
- [4] memoireonline. Diagramme de composant. https://www.memoireonline.com/01/17/9565/m_Application-de-gestion-commerciale-des-produits-alimentaires13.html.
- [5] memoireonline. Diagramme de paquetage. https://www.memoireonline.com/01/17/9565/m_Application-de-gestion-commerciale-des-produits-alimentaires13.html.
- [6] Developpez. Diagramme de cas d'utilisation. <https://laurent-audibert.developpez.com/Cours-UML/?page=diagramme-cas-utilisation>.
- [7] moodle.campusafrika. Exemple diagramme de classe. https://moodle.campusafrika.gos.orange.com/pluginfile.php/1497/mod_resource/content/2/Le%C3%A7on6/web/res/image_12.jpg.
- [8] Journal du net. Hypertext markup langage. <https://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1203255-html-hypertext-markup-langage-definition-traduction/>.
- [9] memoireonline. Logo de bootstrap 5. <https://www.pngaaa.com/detail/3801772>.
- [10] miro.medium.com. Logo de alpinejs. https://miro.medium.com/max/1400/1*KIJIInxXLBElYBcQaEclz6g.png.
- [11] laravel News. Logo de laravel. <https://laravelnews.imgix.net/images/laravel-featured.png?ixlib=php-3.3.1>.
- [12] Webilio. Desc laravel. <https://webilio.xyz/2020/04/17/pourquoi-utiliser-le-framework-laravel/>.
- [13] Livewire. Logo livewire. <https://laravel.sillo.org/wp-content/uploads/2020/10/Capture-62.png>.
- [14] MySQL. Logo mysql. <https://upload.wikimedia.org/wikipedia/fr/thumb/6/62/MySQL.svg/1200px-MySQL.svg.png>.

- [15] Blogdummi. Logo mvc. <https://blogdummi.fr/wp-content/uploads/2019/01/schema-general-architecture-mvc.png>.
- [16] Openclassroom. Model de l'architecture mvc. <https://openclassrooms.com/fr/courses/4670706-adoptez-une-architecture-mvc-en-php/4678736-comment-fonctionne-une-architecture-mvc>.
- [17] Mysql. Logo mysql workbench. <https://img.utdstc.com/icon/f6f/11c/f6f11c75fda63dd454fa5db9610a77cfd6752be4db11010f2e4252551a4abccd:200>.
- [18] DrawIo. Logo drawio. <https://store-images.s-microsoft.com/image/apps.1409.13851527096222888.2b60149a-04a5-4578-a6b2-d7b7377332d5.c22d8e97-4d44-4304-9bd2-55f9d29c0f82?mode=scale&q=90&h=200&w=200&background=%23464646>.
- [19] www.tice-education.fr. description de drawio. <https://www.tice-education.fr/tous-les-articles-et-ressources/articles-internet/819-draw-io-un-outil-pour-dessiner-des-diagrammes-en-ligne>.