Ministère de l'Enseignement Supérieur, de la Recherche et de l'Innovation

Université Assane Seck de Ziguinchor

UFR Sciences et Technologies

Département Informatique

# Mémoire de fin d'études

Pour l'obtention du diplôme de Master
Mention : Informatique
Spécialité : Génie Logiciel

**Sujet**

## *Univ-Money, un porte-monnaie électronique pour la gestion des transactions au sein de l'Université*

Présenté et soutenu par : M. Mor MBOUP

Encadré par : Dr. Ibrahima DIOP

Présenté le 08/12/2021, devant un jury composé

- Pr. Youssou FAYE (**Président**)

- Dr. Ibrahima DIOP (**Encadreur**)

- M. Bassirou DIENE (**Rapporteur**)

- Pr. Youssou DIENG (**Examinateur**)

**Année Universitaire 2020 - 2021**

# DEDICACES

Je dédié ce mémoire à mon grand frère **Mamadou MBOUP**, Enseignant.

# REMERCIMENTS

*En préambule à ce mémoire nous remercions tout d'abord* ***ALLAH (SWT)*** *le Tout Puissant et Miséricordieux, qui nous a donné la force et la patience d'accomplir ce modeste travail et nous prions sur son* ***Prophète MUHAMMAD*** *qu'****ALLAH*** *lui accord la paix et le salut ainsi qu'à sa* ***Famille*** *et ses* ***Compagnons****.*

*Je n'oublie pas mon guide spirituel* ***Cheikh Ahmadou Bamba Khadim Rassoul****, qu'****ALLAH*** *l'agrée, pour ses enseignements de la quintessence de la vie et ses orientations dans l'obéissance au Souverain Juge.*

*Notre profonde gratitude et sincères remerciements vont à l'endroit de notre encadreur* ***Dr Ibrahima DIOP****, pour nous avoir confié ce travail, pour son suivi, sa disponibilité, ses orientations, ses conseils et ses encouragements durant toute la période de réalisation de ce mémoire. Que Dieu l'assiste tout au long de ses projets.*

*Nous remercions* ***Pr. Youssou FAYE*** *d'avoir accepté de présider notre jury ainsi que les autres membres de ce jury -* ***Pr. Youssou DIENG*** *et* ***M. Bassirou DIENE*** *- d'avoir accepté d'évaluer ce modeste travail.*

*Ces remerciements vont aussi au corps professoral et administratif de l'Université Assane SECK de Ziguinchor, plus particulièrement celui du département d'Informatique, pour la richesse et la qualité de leurs enseignements et qui déploient de grands efforts pour assurer à leurs étudiants une formation actualisée.*

*Je remercie mon père* ***Isma Mboup****, qui a toujours veillé sur moi, pour ses encouragements, son aide, sa protection, ses prières et ses conseils tout au long de mes études qu'****ALLAH*** *lui accord une longue vie. Ainsi qu'à ma* ***Feue Maman Déthié SECK****, celle qui m'a donné la vie, celle qui a toujours veillé sur moi pour sa protection, ses orientations et ses conseils que le Bon Dieu l'accueille dans Son Paradis Céleste.*

*Je n'oublie pas mes frères et sœurs pour leurs soutiens et leurs encouragements et généralement à toute ma famille. Je ne remercierai jamais assez mon grand frère* ***Mamadou MBOUP*** *qui, par la volonté de Dieu, si je suis arrivé à ce niveau c'est grâce à lui ; qu'****ALLAH*** *le guide, le protège et lui accorde une longue vie.*

*Je profite de cette occasion pour rendre hommage à :*

- ***Feue Déthié SECK,*** *ma Mère ;*
- ***Feue Ndèye Fatou SOW,*** *défunte épouse de mon grand frère Mamadou MBOUP ;*
- ***Feu Mamadou Petit COULIBALY,*** *mon camarade de promotion de Master ;*
- ***Feu Pape Mamour DIOUF,*** *condisciple et camarade de formation ;*
- ***Feu Alpha SANE,*** *camarade de formation.*

*Qu'ils reposent en paix.*

*Enfin, nous adressons nos plus sincères remerciements à tous nos proches, condisciples et amis, qui nous ont toujours encouragé au cours de la réalisation de ce mémoire.*

*Merci à vous qui lisez ce mémoire.*

# RESUME

L'Université Assane Seck de Ziguinchor (UASZ) est une institution publique en pleine croissance et le nombre de ses étudiants ne cesse d'augmenter. Chaque année on oriente des milliers d'étudiants dans cette Université par conséquent les services rendus aux étudiants deviennent de plus en plus difficiles à gérer. Ces difficultés se manifestent par des files d'attentes d'étudiants pour le paiement des frais d'inscription, l'achat des tickets de restauration, réception des bourses. On peut aussi noter des grèves et des journées sans tickets, quand ces difficultés prennent de l'ampleur.

L'Université a montré sa volonté à résoudre les problèmes liés aux services rendus aux étudiants et ainsi les mettre dans les meilleures conditions au sein du campus en intégrant le paiement mobile par le biais de ces partenaires tels que : Orange Money, Free Money, Wizall Money, Proximo, SesaPay etc. On note, malgré cette volonté de l'UASZ, qu'au lieu que les problèmes se résolvent définitivement, ils se sont déplacés vers d'autres tels que :

- paiement non harmonisé, car effectué avec plusieurs partenaires qui ont des règles de gestion différentes ;
- paiement de frais de services par les étudiants, comme c'est le cas pour le service PAYTECH de paiement des frais d'inscription ;
- non généralisation du paiement mobile dans tous les services au sein de l'Université ;

La solution que nous avons proposée est le système Univ-Money, qui est un porte-monnaie électronique basé sur l'API-Money pour la gestion des transactions estudiantines au sein de l'Université. Elle permettra aux étudiants de recevoir leurs bourses pédagogiques, de payer leurs frais d'inscription, de payer les mensualités de leur logement, d'acheter leurs tickets de restauration à partir de leur porte-monnaie accessible en ligne et par leur téléphone mobile.

Nous avons fait appel à la méthodologie 2TUP pour le processus de développement de notre application et l'approche « contract-first » pour la définition et la modélisation de nos services web. En ce qui concerne la structuration et le stockage des données, nous avons utilisé le Système de Gestion de Bases de Données Relationnelles (SGBDR) MySQL. L'implémentation en JavaScript avec son runtime Node.js nous a permis de mettre en place un système souple, sécurisée, fiable, facile à utiliser.

# TABLES DES MATIERES

# LISTE DES FIGURES

# LISTE DES TABLEAUX

# LISTE DES ABREVIATIONS

**UASZ:** Université Assane SECK de Ziguinchor

**TIC:** Technologies d'Informations et de Communication

**ENT:** Espace Numérique de Travail

**API:** Application Programming Interface

**REST:** REpresentational State Transfer

**UML:** Unified Modeling Language

**2TUP:** 2 Track Unified Process

**URI:** Uniform Ressource Identifier

**CUR:** Centre Universitaire Régional

**UFR:** Unité de Formation et de Recherche

**BU:** Bibliothèque Universitaire

**CROUS/Z:** Centre Régional des Œuvres Universitaires et Sociales de Ziguinchor

**GAB:** Guichet Automatique Bancaire

**INE:** Identité Nationale d'Etudiant

**WSDL:** Web Services Description Language

**JSON:** JavaScript Object Notation

**YAML:** Yet Another Markup Language

**USSD:** Unstructured Supplementary Services Data

**EME**: Etablissement de Monnaie Electronique

**PME**: Porte-monnaie Electronique

# Introduction générale

Les Universités sénégalaises en général et l'Université Assane Seck de Ziguinchor en particulier sont en pleine mutation vers l'automatisation des services rendus aux étudiants dans l'Université, se caractérisant par une marche vers les Technologies de l'Information et de la Communication (TIC).

De plus en plus, l'Université grandit, la gestion des services rendus aux étudiants devient une lourde tâche. Cet état de fait implique qu'il y ait des plateformes informatiques afin de mieux gérer et d'avoir une vue globale sur l'ensemble des services aux étudiants. De ce fait, l'UASZ adopte l'une des utilisations technologiques les plus répandues à travers le monde et qui tend à se répandre dans notre pays qui est le paiement en ligne, que nous retrouvons dans la plupart des sites web ainsi que des applications mobiles, pour le paiement des inscriptions et le paiement des bourses pédagogiques. Et ce pour mieux conduire sa politique d'innovation et sa stratégie de gestion des services aux étudiants en répondant aux besoins croissants d'amélioration de la gestion de ces services.

L'Université a montré sa volonté de résoudre les problèmes liés aux services rendus aux étudiants et ainsi les mettre dans les meilleurs conditions au sein du campus en intégrant le paiement mobile par le biais de ces partenaires tels que : PAYTECH et Ecobank qui sont aussi en partenariat avec des moyens de paiement tels que : Orange Money, Free Money, Wizall Money, Proximo, SesaPay,… pour faciliter les paiements. On note, malgré cette volonté de l'Université, que la tentative de résolution de ces problèmes a engendré d'autres problèmes tels que :

- Le paiement non harmonisé, car effectué avec plusieurs partenaires qui ont des règles de gestion différentes ;
- Le paiement de frais de services par les étudiants ;
- Le paiement d'intérêt par l'Université ;
- Non généralisation du paiement mobile dans tous les services au sein de l'Université.

C'est dans ce sens que nous avons senti le besoin de mise en place d'une solution qui permettra une exécution plus rapide dans la gestion des services aux étudiants et cela sans frais de commissions pour les étudiants ni d'intérêts à payer par l'Université.

Conscient de l'importance de notre projet initié par nous-mêmes, nous avons décidé de mettre sur pied un système de paiement électronique propre à l'Université.

Ainsi est né le projet de mise en place du système Univ-Money qui est un porte-monnaie électronique basé sur l'API-Money pour la gestion des transactions étudiantes au sein de l'Université. Univ-Money permettra aux étudiants de recevoir directement leurs bourses pédagogiques dans leur porte-monnaie numérique ou de créditer le solde de ce dernier chez un distributeur ou un partenaire. Ainsi, une fois les comptes crédités, les étudiants peuvent acheter leurs tickets pour les repas et payer leur logement et inscription dans les applications des partenaires d'Univ-Money (un client restaurant, un client logement, un client inscription) et les applications mises à la disposition des étudiants implémentant les services qu'offre Univ-Money comme « créer de compte étudiant », « débiter le solde du compte d'un étudiant », « créditer le solde du compte d'un étudiant », et « consulter le solde du compte d'un étudiant ».

Le travail qui a été fait est décrit dans la suite de ce présent document qui est organisé en cinq principaux chapitres notamment :

- **Chapitre I : le contexte justificatif** expose les problèmes liés aux services rendus aux étudiants à l'UASZ.
- **Chapitre II : état de l'art sur le porte-monnaie électronique** qui permet de faire une synthèse de la monnaie électronique.
- **Chapitre III : capture et analyse des besoins fonctionnels** consacrés au recensement et à l'analyse des besoins fonctionnels du système Univ-Money.
- **Chapitre IV : capture des besoins techniques et conception** fera mention du recensement des besoins techniques, de la conception préliminaire, de la conception générique et de la conception détaillée du système.
- **Chapitre V : implémentation des services d'Univ-Money** qui fera l'objet d'une présentation des outils que nous avons utilisés pour la production du système Univ-Money et la manière dont les services ont été conçus.
- **Chapitre VI : utilisation des services et réalisation d'un prototype** qui permettent de montrer de façon détaillée l'utilisation des services d'Univ-Money et de réaliser un prototype répondant aux besoins réels de l'Université.

Après ces chapitres, nous avons une conclusion et des perspectives du projet étudié.

# Chapitre I : le contexte justificatif

Ce chapitre décrit, de manière très détaillée, le contexte pour lequel le mémoire est réalisé. On présentera d'abord l'UASZ, ensuite nous décrirons les services offerts aux étudiants puis les bourses pédagogiques de l'Etat pour les étudiants, après nous parlerons évidemment de certaines difficultés rencontrées afin d'améliorer les services rendus aux étudiants et nous terminerons par poser la problématique de notre mémoire.

## I. L'Université Assane Seck de Ziguinchor

Dans les années 2000 l'idée était d'abord de créer un Centre Universitaire Régional (CUR). Toutefois, avec la volonté politique liée à l'élargissement de la carte universitaire et au regard des besoins et des potentialités de la région Sud, l'institution fut créée en tant que Université publique.

Pédagogiquement ouverte en février 2007, l'Université de Ziguinchor est depuis mars 2014 baptisée Université Assane SECK. C'est par le décret numéro 2008-537 du 22 mai 2008 portant création et organisation que l'Université est reconnue comme institution publique.

Une des idées maitresses était de promouvoir la destination Sud, de créer une institution ouverte sur la société et capable d'intégrer dans sa démarche les besoins de la région. C'était aussi un prétexte pour le Sénégal d'ouvrir une passerelle sur la sous-région en répondant au besoin de formation.

Avec trois UFR (l'UFR Sciences et Technologie, l'UFR Sciences Economiques et Sociales et l'UFR des Lettres, Arts et Sciences Humaines), l'Université de Ziguinchor a donc démarré avec 257 étudiants, 20 enseignants-chercheurs et 30 personnels administratifs, techniques et de service.

En 2011, une nouvelle UFR a été ajoutée : il s'agit de l'UFR des Sciences de la Santé. Chacune des UFR dispose d'un service chargé de la pédagogie géré par un chef de service pédagogique, et de départements qui sont gérés par des chefs de départements. Ces départements proposent des formations qui sont gérées par des responsables de formation. Pour

la bonne gestion de ces quatre entités (UFR, services pédagogiques, départements et formations) l'UASZ dispose :

- d'un Personnel Administratif, Technique et de Service (PATS),
- d'un Personnel Enseignant et de Recherche (PER),
- et d'un Personnel Administratif, Technique et de Service Fonctionnaire (PATSF).

De nombreux centres et services communs sont notés et ils ne cessent d'augmenter d'année en année. Ainsi, on distingue :

- la Direction de la Formation Ouverte et à Distance (DFOAD) ;
- la Direction de l'Informatique et des Systèmes d'Information (DISI) ;
- la Bibliothèque Universitaire (BU) ;
- la Direction du service aux Etudiant (DSE) ;
- la Direction des Affaires Financières (DAF) ;
- la Direction Centrale de la Scolarité (DCS) ;
- la Direction de la Gestion du Patrimoine et de Maintenance (DGPM) ;
- la Direction de l'Environnement et de la Sécurité (DES) ;
- la Direction des Ressources Humaines (DRH) ;
- le Centre des Œuvres Universitaires et Sociales (CROUS/Z) ;

Les sept premiers départements étaient celui de Mathématiques, de Physique, de Chimie, d'Informatique, de Géographie, de gestion et la filière d'Informatique appliquée.

Selon les données recueillies à la scolarité, le Tableau 1 ci-après donne des informations détaillées sur l'évolution des étudiants de 2007 à 2018 dans chaque UFR selon l'année et selon le genre.

En 2020, le nombre d'étudiants est passé à sept mille trois-cent vingt-deux (7322). Le nombre estimatif des étudiants est de neuf mille cinq cent (9500) étudiants en 2021.

Université Assane Seck de Ziguinchor ne cesse de grandir, son nombre d'étudiants ne cesse d'augmenter. Ce qui nécessite une bonne organisation pour rendre les services aux étudiants dans les conditions les meilleures favorisant la réussite de ces derniers.

*Tableau 1 : Récapitulatif de l'évolution du nombre d'étudiants de l'UASZ de 2007 à 2018*

| ANNÉE | UFR SES | | | UFR ST | | | UFR LASHU | | | UFR 2S | | | TOTAL |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | F | H | TOTAL | F | H | TOTAL | F | H | TOTAL | F | H | TOTAL | |
| 2006 - 2007 | 24 | 96 | 120 | 18 | 119 | 137 | | | 0 | | | 0 | 257 |
| 2007 - 2008 | 174 | 433 | 607 | 34 | 360 | 394 | 52 | 127 | 179 | | | 0 | 1180 |
| 2008 - 2009 | 331 | 813 | 1144 | 61 | 550 | 611 | 97 | 207 | 304w | | | 0 | 2059 |
| 2009 - 2010 | 462 | 1080 | 1542 | 90 | 675 | 765 | 133 | 303 | 436 | | | 0 | 2743 |
| 2010 - 2011 | 558 | 1178 | 1736 | 122 | 760 | 882 | 171 | 342 | 513 | | | 0 | 3131 |
| 2011 - 2012 | 575 | 1219 | 1794 | 151 | 850 | 1001 | 191 | 389 | 580 | 19 | 26 | 45 | 3420 |
| 2012 - 2013 | 578 | 1281 | 1859 | 182 | 956 | 1138 | 194 | 434 | 628 | 32 | 54 | 86 | 3711 |
| 2013 - 2014 | 545 | 1210 | 1755 | 236 | 1032 | 1268 | 210 | 446 | 656 | 59 | 77 | 136 | 3815 |
| 2014 - 2015 | 493 | 1045 | 1538 | 242 | 1031 | 1273 | 204 | 432 | 636 | 76 | 109 | 185 | 3632 |
| 2015 - 2016 | 441 | 937 | 1378 | 267 | 1036 | 1303 | 179 | 392 | 571 | 97 | 133 | 230 | 3482 |
| 2016 - 2017 | 462 | 838 | 1300 | 299 | 1090 | 1389 | 180 | 389 | 569 | 125 | 152 | 277 | 3535 |
| 2017 - 2018 | 583 | 1024 | 1607 | 361 | 1229 | 1590 | 235 | 531 | 766 | 151 | 205 | 356 | 4319 |

Dans la section qui suit, nous présentons les services rendus aux étudiants à l'UASZ.

## II. Les services aux étudiants

L'Université Assane Seck de Ziguinchor rend, à des degrés divers, des services à ces étudiants visant à améliorer son taux de réussite. Notamment, elle offre aux étudiants des services pour :

- s'inscrire et suivre une formation dans une UFR ;
- se loger au campus ;
- manger au restaurant ;
- se soigner au centre médical ;
- avoir accès à une documentation à la BU ;
- être évalué et disposer d'un diplôme, qui lui ouvre une porte au monde professionnel ;
- …

Dans cette partie nous présentons quelques services que l'Université Assane Seck offre à ses étudiants.

## 1. Inscription des étudiants aux formations de UASZ

L'Université propose plusieurs formations. Chaque étudiant suit une formation donnée en s'inscrivant administrativement et pédagogiquement. Cependant, il doit payer ses frais d'inscription relatifs au niveau où il s'est inscrit. Grâce à sa bourse ou une autre source financière, l'étudiant peut payer son inscription auprès de l'agent comptable de l'UASZ ou bien en ligne. L'agent comptable dispose de différentes ressources informatiques. En termes d'équipements informatiques, elle dispose d'un ordinateur de bureau qui lui permet de renseigner un paiement donné et d'une imprimante pour l'impression des reçus à remettre aux étudiants. En termes de logiciels, l'Université a mis en place une plateforme permettant aux étudiants de payer leur inscription en ligne via la passerelle de paiement PAYTECH qui regroupe différents moyens de paiement notamment Orange Money, Free Money et Visa et MasterCard. Le montant à payer pour chaque étudiant est de 25.000 FCFA pour la Licence, 50.000 FCFA pour le Master et 75.000 FCFA pour le Doctorat.

## 2. La restauration des étudiants

C'est un service de restauration collective destinée aux étudiants qui permet à tout un chacun de prendre ses repas à des prix subventionnés. Pour prendre son repas l'étudiant doit payer un ticket au niveau des vendeurs de ticket situés dans les pavillons. Le ticket est à 100 FCFA pour le déjeuner et le diner et à 50 FCFA pour le petit déjeuner. L'étudiant donne de l'argent liquide au vendeur en échange d'un ticket bleu pour le petit déjeuner et rouge pour le déjeuner et le diner. Avec l'augmentation des étudiants chaque année, la prise du repas est de plus en plus difficile pour les étudiants. En effet, à chaque heure de repas les étudiants sont obligés de se mettre en rang et vu le nombre d'étudiants, on note des files d'attentes très longues pour entrer dans le restaurant ainsi que pour acheter des tickets.

## 3. Le logement des étudiants

C'est un service qui permet d'accueillir et d'orienter les étudiants au sein de l'Université, les mettant ainsi dans des conditions favorables pour suivre les enseignements. Le campus social dispose de plusieurs pavillons. Chacun de ces pavillons est réparti en chambres de deux lits

chacune, des toilettes et une salle pour regarder la télévision. En début d'année, le comptable matière met à la disposition des chefs de pavillons des matelas, des draps et des couvertures pour qu'ils soient distribués aux étudiants. L'attribution des lits ou codification est la procédure qui permet d'octroyer les lits aux étudiants bénéficiaires des œuvres sociales. Cette attribution de lits se fait par ordre de mérite et une partie des lits est réservée pour les cas sociaux. La commission de codification est composée du chef de services d'hébergement, des délégués d'étudiants et des chefs de service pédagogique de chaque UFR. Les bénéficiaires ont l'obligation de verser une somme de 12.000 FCFA au moment de la codification dont 5.000 FCFA pour les frais de caution, 6.000 FCFA pour les mensualités entrée et sortie et 1.000 FCFA pour le guide d'étudiant. Ainsi, chaque bénéficiaire a droit à une clé, un matelas, deux draps, une couverture, une chaise et une table. Chaque mois les étudiants bénéficiaires payent chacun 3.000 FCFA de mensualité.

Pour le paiement des logements l'étudiant donne de l'argent en espèces au représentant de CROUS/Z qui à son tour lui remet un reçu de paiement prouvant qu'il a payé les frais de son logement. Il ne dispose pas d'application pouvant réduire la charge de travail et le temps passé pour la codification.

### 4. La prise en charge médicale

C'est un service qui permet à l'étudiant de se soigner lorsqu'il tombe malade ou de se faire consulter en cas de besoin. Elle assure un suivi sanitaire des étudiants à l'intérieur de l'Université.

Une fois à l'infirmerie, l'étudiant paie son ticket de consultation à 50FCFA en donnant de l'argent liquide au caissier en échange d'un ticket de consultation. Après consultation l'étudiant va au niveau de la pharmacie pour payer son ordonnance qui lui est éventuellement prescrite.

La plupart de ces services nécessitent un paiement de la part de l'étudiant, d'où l'intérêt de disposer d'une bourse d'étudiant, qui finance la vie de l'étudiant à l'Université. La partie qui suit présente les bourses des étudiants.

## III. Les bourses des étudiants

L'Etat du Sénégal, par le Ministère de l'Enseignement Supérieur, de la Recherche et de l'Innovation, spécifiquement par la direction des bourses, octroie des allocations d'études à

savoir bourses, aides, subventions et indemnités aux étudiants remplissant les conditions requises. Les critères d'attribution ont été reconduits conformément au décret n°2014-963 du 12 août 2014 et à l'arrêté 00532 du 15 janvier 2015 [1].

Le Tableau 2 suivant résume les types de bourses, les critères d'attribution de chaque type de bourses.

*Tableau 2 : Liste des attributions des bourses pédagogiques aux étudiants*

| Types de bourses | Critères d'attribution de la bourse | Montant |
|---|---|---|
| **Bourses entières** | Bacheliers de l'année en cours, inscrits dans un établissement supérieur :<br>• ayant obtenu la mention Assez-bien ;<br>• admis par concours ou par sélection ou par test dans les écoles et instituts ;<br>• boursiers de la Coopération | 40.000 FCFA / mois |
| **Demi-bourse** | • Attribuée à tout bachelier des séries S1, S3, S4, S5, T1, T2 et F6 non attributaire d'une bourse d'excellence ou entière.<br>• Pour les bacheliers des séries S2, S2A, G, STEG ayant une moyenne au baccalauréat de 10.50.<br>• Pour les bacheliers des séries L1, L2, L', LAR, LA ayant une moyenne au baccalauréat de 10.90 | 20.000 FCFA / mois |
| **Bourses sociales** | Elles sont octroyées à des étudiants :<br>• Orphelins ou de parents indigents ;<br>• Handicapés physiques ;<br>• Atteints de maladies chroniques | 60.000 FCFA / an |

| | | |
|---|---|---|
| **Aides sociales** | Elles sont attribuées une fois dans l'année aux étudiants non boursiers | 60.000 FCFA / an |
| **Subventions mémoires** | Elles sont octroyées aux étudiants qui rédigent leur mémoire de Master | 100.000 FCFA |

Au niveau l'UASZ, la direction des bourses est représentée par l'antenne des bourses qui reçoit les dossiers des étudiants, les traite avec Microsoft Excel et les envoie à la direction des bourses pour l'établissement des états de paiement.

C'est maintenant que la direction des bourses envoie la liste au département chargé des paiements des bourses de l'institut bancaire Ecobank. Ensuite le chef de ce département traite les listes qui seront envoyées aux différents systèmes de paiement existant y compris ceux de l'Ecobank (Ecobank mobile, Orange money, Free Money, Wizzal Money,…). C'est ainsi que chaque étudiant perçoit sa bourse via un moyen de paiement.

# IV. Les difficultés rencontrées dans les services rendus aux étudiants

Les difficultés rencontrées dans les services rendus aux étudiants sont diverses. Dans cette partie on décrit certaines difficultés rencontrées dans le fonctionnement des services rendus aux étudiants.

## 1. Lenteurs dans le traitement des candidatures de bourses

Au fur et à mesure que l'Université grandisse, le nombre d'étudiant augmente et par conséquence le traitement des dossiers des étudiants prend beaucoup plus de temps. Ce qui va impacter sur la durée nécessaire pour faire le virement des bourses dans les comptes des étudiants. Ce qui fait aussi que le paiement des bourses des étudiants accuse de plus en plus des retards.

## 2. Insécurité dans le traitement des candidatures de bourses

Le fait d'utiliser un fichier Excel ne garantit pas une sécurité totale dans le traitement des candidatures de bourses. En effet, l'agent qui traite les dossiers peut perdre les données ou enregistrer des étudiants qui ne sont pas habilités à avoir des bourses. Le document peut-être

modifié à tout moment, c'est-à-dire qu'une personne tiers peut même ajouter des noms sur la liste sans aucune trace.

## 3. Problèmes liés aux transactions de paiements des bourses d'étudiants

Dans le cadre du paiement des bourses il y a souvent des retards de paiement ce qui engendre parfois des perturbations sur le calendrier universitaire par des grèves ou parfois même sanctionné par des journées sans tickets à la restauration. Avec le paiement des bourses effectué via une banque on assiste à de très longues files d'attentes des étudiants devant les quelques GAB de la banque pour percevoir leurs bourses.

### a. Contrainte de délai de disponibilité de l'argent liquide

Le ministère des Finances fait le contrôle et met à la disposition de la direction des bourses les fonds pour le paiement des bourses. Les fonds sont récoltés à partir des impôts, des taxes, des dons etc. Après avoir récolté des fonds, il s'en suit la distribution de ces derniers vers les différentes entités de l'Etat par ordre de priorité. Ce qui ne garantit pas toujours la disponibilité des fonds destinés au paiement des bourses. Toutefois quand il y a indisponibilité d'argent liquide au niveau du ministère des Finances qui attribue les fonds à la direction des bourses, le paiement des bourses, des aides et des subventions accusera un retard à cause du manque d'argent instantané.

### b. Plusieurs moyens de paiement pour le paiement des bourses

La direction des bourses fait ses états de paiement validés par le ministère des Finances et ça passe par un processus assez long jusqu'à arriver à Ecobank qui fait le paiement. L'Etat du Sénégal paie environ 150.000 étudiants. Ecobank, ne pouvant pas payer en un seul jour pour 150.000 étudiants, à trouver de partenaires comme Orange Money, Free Money, Wizall Money, Proximo, SesaPay...

### c. Frais de paiement des transferts des paiements de bourses

Le paiement en ligne des bourses n'est pas sans frais. En effet, le paiement des bourses via les moyens de paiement (Ecobank mobile, Orange money, Free Money, Wizzal Money…) nécessite de payer des commissions par la direction des bourses aux différents opérateurs. Et vu le nombre d'étudiants boursiers dans toutes les Universités du Sénégal classés par niveau

(Licence ou Master ou Doctorat) et selon le type de bourse obtenu (cf. Tableau 2), la somme d'argent dépensée en guise de commissions est estimée à des centaines de millions.

### 4. Lenteurs dans la réception des bourses

D'après les enquêtes que nous avons menées, nous avons souligné que la procédure de paiement adoptée est très longue. En effet, après avoir traité les dossiers des étudiants au niveau de l'Université, l'antenne des bourses doit envoyer la liste obtenue à la direction des bourses afin d'établir les états de paiement. Et ceci est fait dans toutes les Universités du Sénégal. La direction des bourses fait ses états de paiement validés par le ministère des Finances et ça passe par un processus assez long jusqu'à arriver à Ecobank qui fait le paiement. L'Etat du Sénégal paie environ 150.000 étudiants. Ecobank, ne pouvant pas payer en un seul jour pour 150.000 étudiants, la banque a trouvé des partenaires comme Orange Money, Free Money, Wizall Money, Proximo, SesaPay etc. C'est là où se trouvent les problèmes. L'étudiant peine à recevoir le code pour être payer et parfois il ne sait pas qui paie car les opérateurs peuvent changer. Parmi les opérateurs aussi il peut y avoir certains qui n'ont pas un montant élevé [2].

### 5. Des grèves et des journées sans ticket

Les lenteurs que nous avons énumérées plus haut sont fréquemment des motifs de grèves ou de journées sans ticket au niveau des restaurants de l'Université. Quand les étudiants ne reçoivent pas leurs bourses à temps ils mettent des stratégies de grèves en place. Il s'agit de prendre leurs repas sans payer de tickets ou de descendre dans les rues pour affronter les forces de l'ordre. Ce qui est encore un problème dont on œuvre pour y remédier.

### 6. Problèmes liés aux transactions de paiement des services rendus par l'Université aux étudiants

Dans le contexte actuel de l'Université on note chaque début d'année des files d'attentes interminables d'étudiants devant le bureau de l'agent comptable pour le paiement des frais de scolarité. Ce même constat est fait au niveau de la restauration notamment au moment de l'achat des tickets et au moment d'entrer dans les salles à manger. C'est le même cas avec la gestion manuelle des paiements des frais de logement social avec des registres de paiement. On ne dispose pas de données réelles sur la restauration. Par conséquent, un étudiant peut prendre deux fois le même repas sans être repéré.

### a. Plusieurs moyens de paiement pour le paiement des frais d'inscriptions

En termes de logiciels, l'Université a mis en place une plateforme permettant aux étudiants de payer leurs inscriptions en ligne via la passerelle de paiement PAYTECH qui regroupe différents moyens de paiement notamment Orange Money, Free Money et les cartes Visa et MasterCard.

### b. Frais de paiement des transferts lors des paiements des frais d'inscription

Le paiement en ligne des inscriptions ne se fait pas gratuitement. En effet, le paiement des inscriptions en ligne via la plateforme ENT engendre des frais de paiement d'un montant de 1.5% du montant à payer relatif au niveau (Licence ou Master ou Doctorat) où l'étudiant s'est inscrit. Ce qui fait que certains étudiants vont au niveau de l'agence comptable pour payer leurs inscriptions à cause de la cherté des frais de paiement. Le Tableau 3 ci-après est une répartition des étudiants par niveau d'étude en 2018.

*Tableau 3 : répartition des étudiants par niveau d'étude en 2018*

| Niveau | Nombre d'étudiants | Frais d'inscription |
|---|---|---|
| **Licence 1** | 1577 | 25.000 FCFA |
| **Licence 2** | 877 | 25.000 FCFA |
| **Licence 3** | 810 | 25.000 FCFA |
| **Master 1** | 335 | 50.000 FCFA |
| **Master 2** | 514 | 50.000 FCFA |
| **Doctorat 1** | 105 | 75.000 FCFA |
| **Doctorat 2** | 48 | 75.000 FCFA |
| **Doctorat 3** | 53 | 75.000 FCFA |

Ce paiement en ligne des frais d'inscription n'est pas sans coût pour les étudiants. Ainsi, d'après le tableau ci-dessus on peut estimer à 2.092.500 FCFA la somme des commissions payées par les étudiants pour l'effectivité des inscriptions en ligne en 2018 à l'Université Assane Seck de Ziguinchor. Ce qui permet de dire qu'en 2021, la somme des commissions payées par les étudiants est aux environs de 10.000.000 FCFA.

### 7. Des rangs interminables pour acheter des tickets de restauration, s'inscrire ou codifier pour une chambre

On note aussi des files d'attentes interminables au niveau des différents lieux où on achète des tickets de restauration, au niveau de l'agence comptable pour le paiement des frais d'inscription et au moment de payer les frais de logements sociaux. Ce sont des difficultés, parmi tant d'autres auxquelles nous allons proposer une solution.

La gestion des services rendus aux étudiants par les structures administratives et sociales de l'Université et ceux rendus à ces derniers par la direction des bourses nécessitent une amélioration. En effet, l'étudiant qui reçoit sa bourse via un moyen de paiement agréé, est obligé de retirer son argent en liquide pour aller payer soit des tickets de restauration, soit ses frais médicaux ou bien son logement et éventuellement ses frais d'inscription.

Le défi est grand pour rendre ce système plus flexible et moins contraignant pour les autorités en charge de la gestion de l'enseignement supérieur. Dans la section suivante, nous présentons la problématique de ce mémoire et une solution pour révolutionner la gestion des transactions entre étudiants, Université et Direction des bourses.

## V. Problématique du sujet

L'efficacité et la rapidité des moyens de paiement mobiles (Ecobank mobile, Orange money, Free Money, Wizzal Money ou PAYTECH qui regroupent différents moyens de paiement tels que cités précédemment mais aussi les cartes Visa et MasterCard) ont influencé le paiement des bourses d'étudiants et le paiement des frais d'inscriptions dans les Universités du Sénégal. Ce qui fait de l'Université un écosystème où cohabite tout un ensemble de moyens de paiements avec des tarifications des commissions aussi différentes de par leur mode de fonctionnement.

Les frais de commissions payés par l'Université dans son ensemble pourraient augmenter si on essaie de généraliser le paiement dans tout le fonctionnement de l'Université à savoir dans le paiement des services rendus aux étudiants dans le campus (Restauration, logement, soin médical, …).

Deux besoins :

- Disposer d'un seul moyen de paiement (des services et des bourses) des transactions bien adapte à l'espace universitaire, pour ainsi :

    - unifier les paiements par un système appartenant à l'enseignement supérieur ;
    - supprimer les commissions ;
    - disposer des données sur les transactions universitaires.
- Réduire les contraintes liées à la disponibilité d'argent liquide au début de chaque mois pour payer les bourses.

Notre problématique est de concevoir et d'implémenter un Système composé d'un calpé nommé Univ-Money et d'un ensemble de services web, qui permettra d'implémenter différents types d'applications web ou mobile de gestion des transactions entre les étudiants et les structures universitaires pour :

- Le paiement des frais de scolarité ;
- Le paiement des frais de logement social ;
- L'achat des tickets de restauration ;
- Le paiement des frais médicaux ;
- …

Univ-Money permettra aussi la gestion des transactions de la direction des bourses vers les étudiants pour :

- Le paiement des bourses ;
- Le paiement des aides ;
- Le paiement des subventions ;

Avec son calpé numérique crédité, par une bourse ou personnellement dans un point de recharge Univ-Money, un étudiant peut payer ses frais de scolarité, son logement à l'Université et s'approvisionner de tickets (petit déjeuner, déjeuner et diner) pour la restauration, le tout avec son téléphone mobile sans avoir à faire des rangs interminables.

Ainsi avec sa carte d'étudiant ou avec une application installée dans son téléphone, contenant la photo de l'étudiant, le numéro INE (Identité National de l'Etudiant) en QRCODE, l'étudiant sera reconnu par les applications de :

- Gestion de la scolarité afin de payer ses frais d'inscription ;
- Gestion de logement pour le paiement des frais de logement ;
- Gestion du restaurant pour l'achat de ticket ;

- Gestion médical pour pouvoir payer ses frais médicaux.

# VI. Objectifs et besoins généraux

## 1. Objectifs

Le projet s'inscrit dans l'initiative de proposer des services variés et adaptés au fonctionnement des structures administratives et sociales de l'Université ainsi qu'au fonctionnement de la direction des bourses. Il s'agit entre autres de :

- Proposer une meilleure gestion transparente des transactions entre les étudiants et les services qui composent le monde universitaire
- Faciliter les transactions entre la direction des bourses et les étudiants
- Rendre plus pratique le paiement des services au sein de l'Université
- Disposer d'un outil de paiement personnalisé à l'Université
- Se doter d'un outil conforme aux besoins actuels et ouverts aux perspectives d'évolutions futures des activités de l'Université.

## 2. Contraintes à respecter

En termes de paiement de services, les aspects les plus importants sont la sécurité, la fiabilité et la rapidité des transactions ainsi que la disponibilité du système compte tenu du caractère temps réel des traitements de transactions. En effet, de nombreux risques menacent tout système qui gère des informations sensibles comme celles des paiements de services. Il est donc nécessaire de mettre en place un système couvert d'une importante politique de sécurité d'accès ainsi que des mesures de protection des données et des matériels en exploitation.

Un système de paiement indisponible peut engendrer une perturbation du calendrier universitaire.

Une autre contrainte à respecter est la prise en compte des ressources matérielles et humaines.

Il faudra aussi mettre en place un établissement financier dépendant de la direction des bourses pour permettre la distribution de monnaie électronique.

## 3. Enjeux du système à mettre en place

La mise en place du système de paiement spécifique à l'Université contribuera à la bonne gestion du système éducatif. Notamment dans le domaine social ainsi que dans le domaine administratif. En effet, l'association d'un compte money-universitaire à un étudiant présente plusieurs avantages. D'une part, il permettra à la direction des bourses d'avoir un lien beaucoup plus facile avec l'Université pour effectuer les paiements des bourses. Ainsi les procédures de paiement seront beaucoup plus optimales et réduira un certain nombre de problèmes liés au paiement des bourses, notamment les omissions ainsi que les retards de paiement. D'autre part, au sein de l'Université, il permettra de réduire des files d'attentes constatées lors des paiements des frais d'inscription et des achats de tickets pour la restauration.

De plus, un système interne permettra de réduire les frais de commissions cités plus haut.

## 4. Quelques fonctionnalités attendues du système

Le système regroupe un ensemble de fonctionnalités. Parmi elles on peut citer :

- Consultation de solde
- Créditer un compte
- Débiter un compte
- Transfert d'argent
- Création de compte pour les étudiants
- Création de compte pour les partenaires
- Création de compte pour les distributeurs

Pour atteindre nos objectifs, il est indispensable de suivre un processus de développement de notre système.

## 5. Processus de développement du système

Le processus de développement décrit une approche du développement logiciel. Il définit une séquence d'étapes, en parties ordonnées, qui concourent à l'obtention d'un système logiciel ou à l'évolution d'un système existant.

En termes de projets, il est d'une importance capitale d'opter et de pratiquer une méthodologie de gestion de projet adaptée au contexte dans lequel on se trouve. Le choix de la méthode idéale

a été porté sur le processus unifié parce qu'il intègre le 2 Track Unified Process (2TUP). Ce dernier en plus d'apporter des réponses aux problèmes de changement continuel imposé aux systèmes d'informations des entreprises, renforce le contrôle des capacités d'évolution et de correction du système tout en permettant de mieux prendre en compte les besoins des futurs utilisateurs. D'ailleurs, 2 Track signifie littéralement que le processus suit deux chemins. Il s'agit de la branche fonctionnelle et de la branche technique, qui sont les deux axes de changements imposés aux systèmes d'information. A l'issue des évolutions des modèles fonctionnels et de l'architecture technique, la réalisation du système consiste à fusionner les résultats des deux branches. La Figure 1 suivante est une représentation graphique de cette méthode.

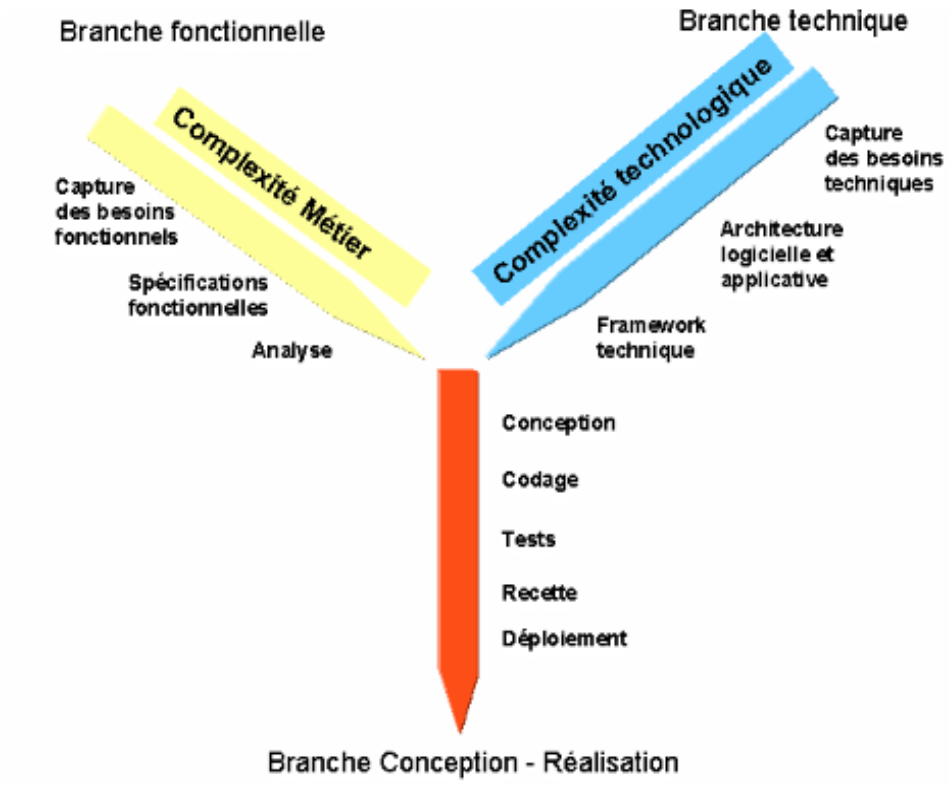


*Figure 1 : Représentation graphique de la méthodologie 2TUP.*

Les paragraphes précédents ont montré l'importance de l'application d'une méthode de développement pour la réussite d'un projet de mise en place d'un système d'information informatique. A travers eux, l'efficacité de la méthode 2TUP pour la réalisation d'un système d'information a été démontrée. Les paragraphes qui suivent expliciteront les besoins fonctionnels du notre système d'information.

Pour la conception de nos APIs nous avons décidé d'utiliser l'approche du contrat d'abord ou « contract firts » en anglais qui permet de concevoir des APIs à partir d'un contrat.

# Chapitre II : Etat de l'art sur le porte-monnaie électronique

Selon l'Instruction n°008-05-2015 régissant les conditions et modalités d'exercice des activités des émetteurs de monnaie électronique (*Article 8*), à l'exception des banques et des établissements financiers de paiement habilités par la loi portant réglementation bancaire, aucune structure ou établissement ne peut exercer des activités d'émission de monnaie électronique, sans avoir été dûment agréé ou autorisé préalablement par la Banque Centrale.

Sur les huit (8) initiatives d'émission de Mobile Money au Sénégal, seulement deux (2) acteurs sont indépendants des banques en ce qui concerne la licence d'émetteur de Monnaie électronique délivrée par la Banque Centrale des Etats de l'Afrique de l'Ouest. Les deux EMEs sont des initiatives des compagnies télécoms Orange et Free qui sont agréées à travers leurs spin-off Ofms et Mobile Cash qui émettent et distribuent respectivement Orange Money et Free Money. Les autres initiatives sont des partenariats entre des banques, des fintechs et des compagnies télécoms [3].

Pour comprendre la notion de porte-monnaie électronique, il faut passer par la définition de plusieurs concepts importants tels que :

- **Etablissement émetteur** : Les établissements émetteurs sont les banques, les établissements financiers de paiement, les systèmes financiers décentralisés dûment autorisés et les établissements de monnaie électronique.
- **Etablissement financier de paiement** : Ce sont les établissements financiers de paiement au sens de l'Instruction n° 011-12/2010/RB relative au classement, aux opérations et à la forme juridique des établissements financiers à caractère bancaire.
- **Etablissement de monnaie électronique** : Un établissement de monnaie électronique est toute personne morale, autre que les banques, les établissements financiers de paiement et les systèmes financiers décentralisés, habilités à émettre des moyens de paiement sous forme de monnaie électronique et dont les activités se limitent à : l'émission de monnaie électronique ; la distribution de monnaie électronique.
- **Emission de monnaie électronique** : Il s'agit de l'émission d'unités de valeurs électroniques en contrepartie de fonds reçus.

- **Système Financier Décentralisé (SFD)** : Un Système Financier Décentralisé est une institution dont l'objet principal est d'offrir des services financiers à des personnes qui n'ont généralement pas accès aux opérations des banques et établissements financiers tels que définis par la loi portant réglementation bancaire et habilités aux termes de la loi portant réglementation des systèmes financiers décentralisés à fournir ces prestations.
- **Distributeur** : Un distributeur est une personne morale ou physique inscrite au Registre du Commerce et du Crédit Mobilier, ou système financier décentralisé, offrant à la clientèle, en exécution d'un contrat avec l'établissement émetteur, un service de distribution de monnaie électronique.
- **Sous-distributeur** : Un sous-distributeur est une personne morale ou physique ou le système financier décentralisé, offrant à la clientèle, en exécution d'un contrat avec le distributeur, sous la responsabilité de l'émetteur, un service de distribution de monnaie électronique.
- **Distribution de monnaie électronique** : La distribution de monnaie est l'ensemble des services de retrait d'espèces, de chargement et rechargement contre remise de monnaie fiduciaire ou scripturale, de paiement et de transfert d'argent liés à la monnaie électronique.
- **Monnaie électronique** : La monnaie électronique est une valeur monétaire représentant une créance sur l'établissement émetteur qui est stockée sous une forme électronique, y compris magnétique, émise sans délai contre la remise de fonds d'un montant qui n'est pas inférieur à la valeur monétaire émise et acceptée comme moyen de paiement par des personnes physiques ou morales autres que l'établissement émetteur.

# I. Porte-monnaie électronique

## 1. Définition

Le porte-monnaie électronique (PME) - ou portemonnaie électronique selon la réforme de l'orthographe de 1990 - est un dispositif qui peut stocker de la monnaie sans avoir besoin d'un compte bancaire et d'effectuer directement des paiements sur des terminaux de paiement [4].

Le porte-monnaie électronique, également dit portefeuille électronique (e-wallet en anglais), couvre deux réalités différentes :

- Le premier type de PME est la carte bancaire prépayée, qui peut stocker de la monnaie sans avoir besoin d'un compte bancaire et d'effectuer directement des paiements sur des terminaux de paiement. Elles ont été créées sur le même modèle que les cartes bancaires à puces (Visa, MasterCard, etc.) dont le but d'effectuer ses achats facilement à l'aide d'une carte stockant la monnaie plutôt que de transporter avec soi une certaine quantité de liquidité.
- Un PME peut aussi prendre la forme d'un dispositif de paiement lié directement à une application de téléphone mobile, à une adresse mail ou encore à un numéro de téléphone. Cela permet au créditeur de transférer de l'argent depuis son compte sans avoir besoin des coordonnées bancaires du bénéficiaire.

Ces deux types de PME se démarquent donc des moyens de paiements traditionnels tels que le virement par leur facilité d'utilisation, la sécurité et l'exécution rapide des transferts. Ces transferts peuvent se faire à différentes échelles : entre deux entreprises, entre une société et un particulier ou encore entre des particuliers.

### 2. Utilisation

L'implémentation de ce nouveau système permettra de réduire la circulation des espèces dans l'Université ce qui diminue drastiquement les coûts de gestion que ce soit pour la Scolarité, le CROUS/Z (comptage, conservation, transport, etc.) ou pour la Direction des bourses (fonds de caisse à conserver, comptages, manipulations, transport, etc.). Plus particulièrement les étudiants se voient également moins encombrés et le risque de vol est fortement réduit.

Le PME permettra également aux étudiants d'effectuer des transactions en ligne (soit pour le paiement des frais d'inscription, pour l'achat de tickets pour la restauration, etc.) sur des sites.

Le PME permettra aussi à la Direction des bourses de faire les virements des bourses chaque mois et bien sûr avec le respect les délais de paiement vu le caractère numérique de ce moyen de paiement.

### 3. PME et paiement mobile

L'avènement de l'internet en 2000 et l'arrivée des smartphones un peu plus tard ont ouvert la perspective de création de nouveaux PME, car la possibilité d'avoir internet sur son appareil mobile permet alors de bénéficier des services offerts par les banques à portée de main et à tout

moment. Le paiement mobile est une notion apparue vers 2010 pour désigner les divers achats effectuables à partir d'un téléphone portable. Parmi les services alors proposés certains permettent au téléphone mobile de devenir un porte-monnaie électronique rechargeable et géré par le fournisseur de services du paiement. De nombreuses applications voient alors le jour et répondent à la description de PME, deux catégories peuvent alors être définies :

- La première regroupe les applications mobiles qui ont le même fonctionnement que la carte prépayée. Ces dernières reposent sur le stockage de monnaie soit depuis son compte bancaire ou un rechargement par carte bancaire sur diverses bornes (banques, bureaux de poste, partenaires).
- La deuxième est appelée le Mobile Money qui ne nécessite pas d'être souscrit à une banque, ni même d'accès à Internet. Il s'agit d'une technologie reposant sur les opérateurs téléphoniques pour effectuer les transactions souhaitées. Ces derniers font office d'intermédiaire pour le rechargement du PME et le transfert d'argent entre particuliers.

## 4. Architecture

Une application de porte-monnaie électronique ou numérique est une application qui permet d'effectuer des paiements numériques à partir d'un site web. Le composant logiciel assure la sécurité et un cryptage fort des données. Il existe de nombreux types de porte-monnaie électroniques et chacun a des modes de traitement des paiements différents.

Dans le cas d'Univ-Money nous avons une architecture composée des éléments matériels et logiciels suivants :

- Serveur calpé Univ-Money : qui permet la gestion des transactions sur le porte-monnaie.
- Serveur d'autorisation (API-money) : contrôle les informations de chaque traitement, l'état des porte-monnaie. Un contrôle sur la fraude peut également être effectué.
- Serveur HTTPS : ce sont les serveurs des différents systèmes qui sont mis en place pour faire des transactions sur le porte-monnaie électronique. Ils sont des applications web ou mobile.
- SSL : technologie de sécurité standard pour établir une connexion chiffrée entre un serveur web et un navigateur.

- HTTPS (HyperText Transfer Protocol Secure) : désigne la version sécurisée du protocole HTTP, un protocole de communication qui permet la liaison entre un client et un serveur pour le World Wide Web (www).
- ISO 8583 : Il s'agit de la norme de l'Organisation internationale de normalisation pour les systèmes d'échange de transactions électroniques initiées par les titulaires de cartes.

La Figure 2 suivante représente l'architecture technique du porte-monnaie électronique Univ-Money.

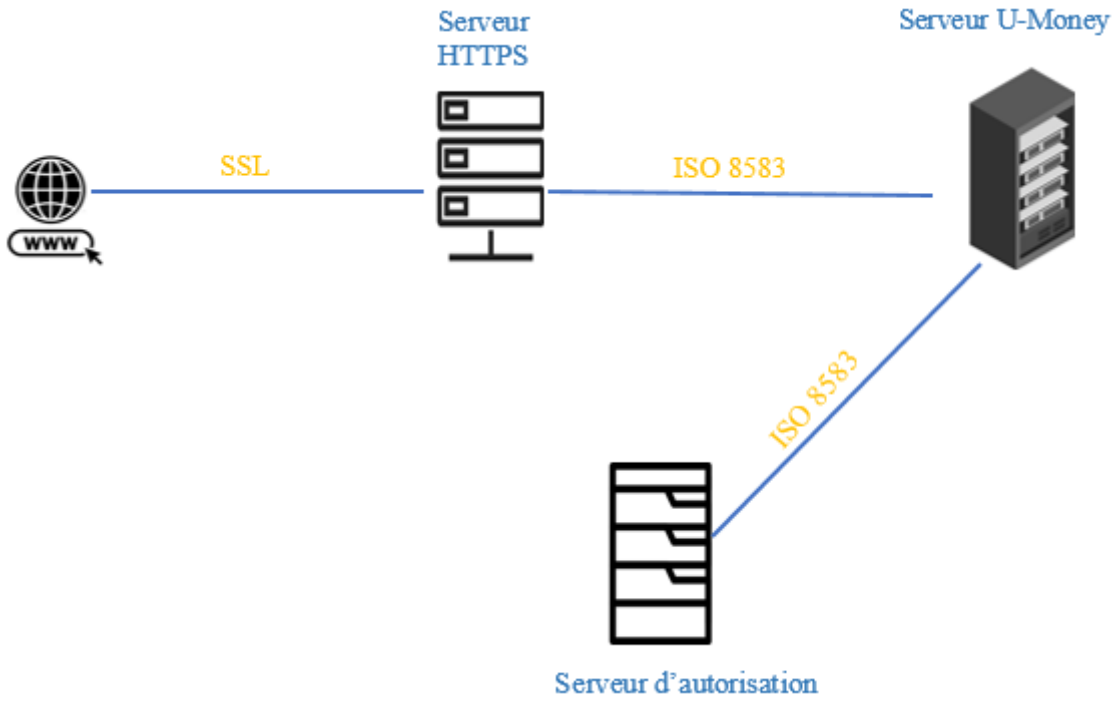


*Figure 2 : Architecture application porte-monnaie électronique*

Ainsi, la mise en place d'Univ-Money nécessite un partenariat avec un Etablissement Financier de Paiement. Car, n'étant pas agrée à distribuer de la monnaie électronique, Univ-Money doit utiliser la licence d'une banque ou d'un SFD qui est autorisée à émettre de la monnaie électronique pour ainsi devenir un prestataire techniques. D'où l'utilisation de l'API-money.

## II. API-money

W-HA est une société anonyme de droit français agréée en France en tant qu'Etablissement de Monnaie Electronique. API-money est une solution de paiement par monnaie électronique de W-HA, filiale à 100% du groupe Orange et agréé par l'Autorité de Contrôle Prudentiel et de Résolution (ACPR) en tant qu'Etablissement de Monnaie Electronique(EME). Elle désigne

l'interface de programmation applicative ou API (Application Programming Interface) mise à la disposition du partenaire par l'émetteur (W-HA) pour offrir une fonctionnalité de paiement ou d'encaissement par émission de monnaie électronique sur le système informatique du partenaire [5].

### 1. Partenaire

Un partenaire est une société offrant à ses utilisateurs, en qualité de mandataire de l'émetteur, la possibilité d'ouvrir un compte, d'acquérir de la monnaie électronique émise par l'émetteur par l'intermédiaire du système informatique du partenaire, et d'utiliser la monnaie électronique en vue de réaliser une transaction.

### 2. La création de compte

Le terme compte désigne le compte de monnaie électronique ouvert, sur demande du Partenaire, par l'émetteur dans ses livres au nom de l'utilisateur. L'API-money offre des fonctionnalités de création de compte. Il existe deux types de comptes :

- Le compte STANDARD pour les particuliers
- Le compte BUSSINESS pour les professionnels

### 3. Le transfert

Un transfert P2P (Point-à-Point ou Peer-to-Peer) est un transfert de monnaie électronique entre deux portefeuilles «EMONEY». En option, certains frais peuvent être perçus par le partenaire lors de l'opération.

### 4. L'encaissement par carte de crédit (cash-in)

L'encaissement par carte de crédit consiste à alimenter un porte-monnaie électronique en monnaie électronique en utilisant une carte de crédit en cours de validité et contenant un solde suffisant pour effectuer la transaction.

### 5. L'encaissement par virement bancaire (cash-in)

L'encaissement par virement bancaire consiste à alimenter un porte-monnaie électronique en monnaie électronique en utilisant un compte bancaire contenant un solde suffisant pour effectuer la transaction en fournissant un IBAN et un BIC valides.

### 6. Le retrait (cash-out)

Un retrait est une transaction visant à extraire de l'argent d'un portefeuille et à transférer de l'argent sur un compte bancaire. Pour effectuer un retrait d'un portefeuille, au moins un compte bancaire doit être ajouté au compte associé. Le retrait peut être effectué sur des portefeuilles appartenant à des comptes STANDARD et BUSINESS mais également sur des portefeuilles appartenant au partenaire lui-même (portefeuilles EMONEY ou FEES).

### 7. Transaction

Une transaction est une opération réalisée par un utilisateur-payeur au bénéfice d'un utilisateur-bénéficiaire en vue de l'acquisition, par ce dernier, d'un bien ou service. Cette opération de paiement prend la forme de l'acquisition de monnaie électronique auprès de l'émetteur et son transfert sur le compte de l'utilisateur-bénéficiaire désigné par l'utilisateur-payeur. Elle désigne toute transaction conclue par un utilisateur conformément aux conditions générales du dispositif informatique du partenaire et donnant lieu à une opération de paiement.

L'API-money offre plusieurs fonctionnalités à ses partenaires dont la création de compte, le transfert, l'encaissement par carte de crédit, l'encaissement par virement bancaire et le retrait. Ces services sont utilisés par Univ-Money pour la distribution de monnaie électronique de façon générale. Les deux types d'encaissement cités plus haut permettent à Univ-Money d'alimenter son porte-monnaie électronique afin de pouvoir distribuer de la monnaie électronique à ses utilisateurs. Cette distribution de la monnaie électronique fait appel à la fonctionnalité de transfert de monnaie électronique du compte d'Univ-Money vers les comptes de ses utilisateurs, il s'agit de l'opération communément appelé dépôt. Le propriétaire d'Univ-Money peut, en cas de besoins d'argent, utiliser la fonctionnalité retrait pour déplacer de la monnaie électronique de son compte API-money vers son compte bancaire associé à son compte API-money.

La partie qui suit fera l'objet de recensement et d'analyse des différents besoins fonctionnels du système Univ-Money en faisant ressortir les différents diagrammes UML.

# Chapitre III : Capture et analyse des besoins fonctionnels du système

Dans ce chapitre, nous parlerons d'abord de la capture des besoins fonctionnels et ensuite de l'analyse de ces besoins. Elles sont deux étapes d'un projet qu'on trouve dans la branche gauche du modèle en Y.

## I. Capture des besoins fonctionnels

La capture des besoins fonctionnels consiste à relever les parties indispensables à l'utilité et à l'utilisabilité du futur système. La capture des besoins fonctionnels est donc une étape importante et obligatoire dans le modèle 2TUP. Ainsi dans la suite de cette partie, nous identifions les acteurs et les axes fonctionnels du système pour ensuite montrer les relations qui les lient en s'appuyant sur des diagrammes de cas d'utilisation. Enfin nous décrirons les cas d'utilisation qui constituent les différentes fonctionnalités du système.

### 1. Identifications des acteurs du système

Un acteur est une entité externe qui interagit avec le système. Il peut être un utilisateur physique, un périphérique externe ou un système externe. Un acteur est généralement représenté à l'aide d'un bonhomme en dessous duquel est écrit son nom. Lorsque l'acteur n'est pas humain, par exemple pour un système logiciel ou un système physique, le bonhomme peut être remplacé par un pictogramme plus adapté, voir la Figure 3 suivante.


*Figure 3 : représentation graphique d'un acteur en UML.*

Dans le cas de notre sujet, nous avons identifié quatre acteurs, décrits dans le Tableau 4 suivant :

*Tableau 4 : Liste des Acteurs du système et leur description*

| Acteur | Description |
|---|---|
| **Client Etudiant** | Il s'agit du client web, du client mobile et du client USSD mis à la disposition des étudiants pour faire leurs transactions. |
| **Client distributeur** | C'est le système informatique mis en place pour les points de distribution de la monnaie électronique du Calpé Univ-Money. |
| **Client partenaire** | Il s'agit du client gestion logement, du client gestion scolarité, du client gestion restauration, du client gestion médicale et du client gestion des bourses qui sont des systèmes développés au niveau des différentes structures de l'Université. |
| **API-money** | Le système qui nous offre les services de paiement et d'encaissement de la monnaie électronique. Il autorise les opérations effectuées au niveau d'Univ-money |
| **Client admin** | Il permet de maintenir le système, de créer des comptes pour les utilisateurs et de gérer les services. |

## 2. Identifications des axes de fonctionnement du système

Les besoins fonctionnels sont les actions réalisées par le système en réponse aux demandes effectuées par les acteurs.

Les besoins fonctionnels auxquels notre système doit répondre ont été identifiés et illustrés dans le Tableau 5 suivant :

*Tableau 5 : Liste des services ou des fonctionnalités du système*

| Service ou Fonctionnalité | Acteur(s) |
|---|---|
| **Consulter solde** | Client Etudiant, Client partenaire, Client partenaire et Client distributeur |
| **Créditer un compte** | Client administrateur |

| | |
|---|---|
| **Débiter un compte** | Client administrateur |
| **Transférer de l'argent** | Client Etudiant, Client partenaire et Client distributeur |
| **Créer compte Etudiant** | Client distributeur, Client partenaire |
| **Créer compte Distributeur** | Client administrateur |
| **Créer compte Partenaire** | Client administrateur |

## 3. Diagrammes de cas d'utilisation

Le diagramme de cas d'utilisation permet de donner une vision globale du comportement fonctionnel du système. Il représente la structure des grandes fonctionnalités du système. Il définit la relation qui existe entre les utilisateurs et les cas d'utilisation. Un cas d'utilisation représente une unité discrète d'interaction entre un utilisateur et le système. Le diagramme de cas d'utilisation est composé par les utilisateurs qui sont appelés acteurs (actors) et les cas d'utilisation (use case). Le diagramme de cas d'utilisation de notre projet est décomposé en plusieurs diagrammes.

### a. Diagramme de cas d'utilisation pour le service de la création de compte

Le service de la création de compte peut être utilisé dans les trois cas suivants :

- soit pour créer des comptes étudiants ;
- soit pour créer des comptes distributeurs ;
- soit pour créer des comptes partenaires (ou les instances universitaires notamment Scolarité, Direction des bourses, CROUS/Z qui veulent intégrer Univ-Money dans leurs systèmes pour la gestion des transactions).

Les systèmes externes qui veulent utiliser ce service vont devoir soumettre une demande de création de compte au système Univ-Money. La demande est autorisée et exécutée au niveau du système d'API-money. Le diagramme de cas d'utilisation de la Figure 4 suivante représente la création des comptes dans le système Univ-Money.

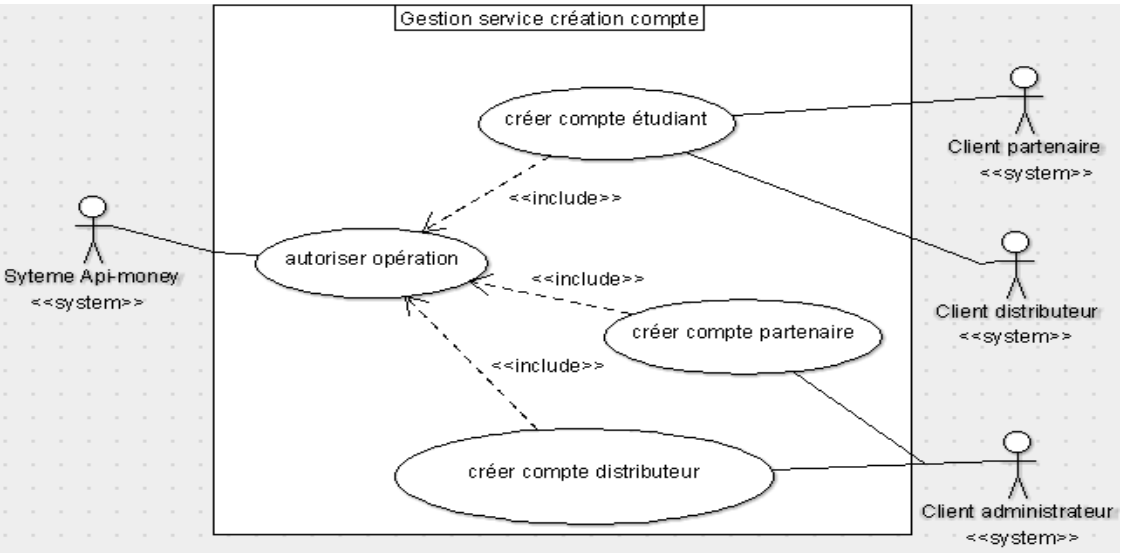


*Figure 4 : Diagramme de cas d'utilisation pour le service de la création de compte*

## b. Diagramme de cas d'utilisation pour les services de base

Le système Univ-Money fournit des services de transfert d'argent, de dépôt d'argent, de retrait d'argent et de consultation de solde, voir le diagramme de cas d'utilisation de la Figure 5 suivante. Les systèmes externes qui utilisent ces services soumettent des demandes pour faire des transactions au système d'Univ-Money. Le système API-money autorise la demande et exécute la transaction.

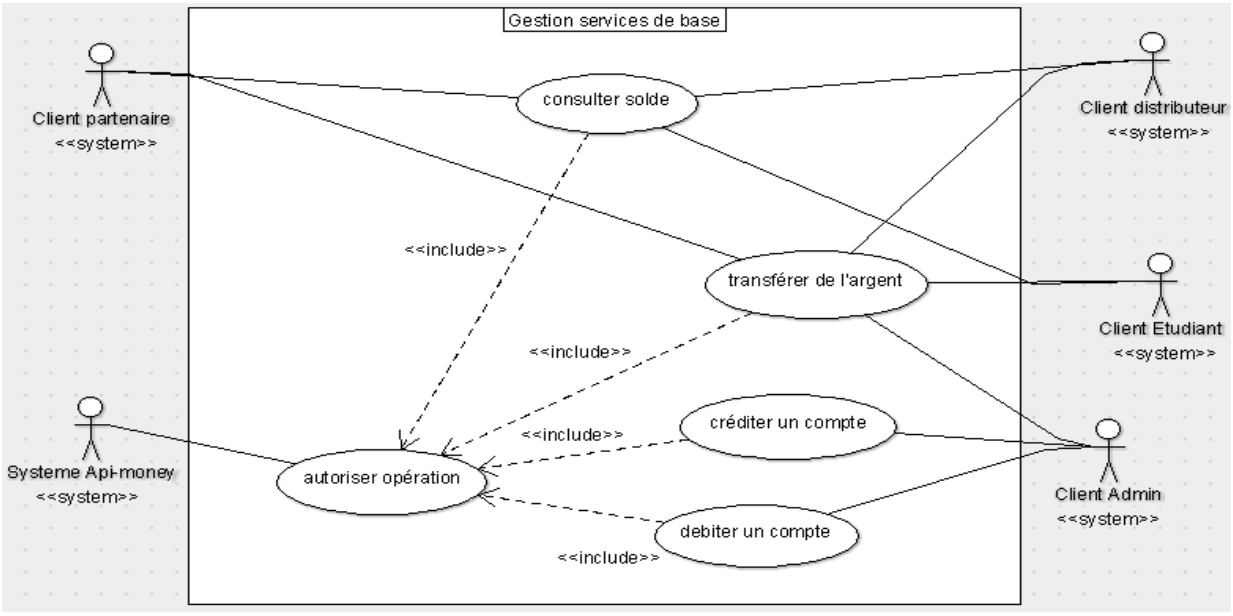


*Figure 5 : Diagramme de cas d'utilisation pour les services de base*

Dans la partie qui suit nous décrirons de manière plus détaillée les différents cas d'utilisation.

## 4. Description des cas d'utilisation du système

Le diagramme de cas d'utilisation décrit les grandes fonctions du système du point de vue des acteurs, mais n'expose pas de façon détaillée le dialogue entre les acteurs et les cas d'utilisation. De ce fait, il est important de rédiger une description textuelle qui permet de fournir un format de présentation textuelle qui est plus souple. Avec UML, on peut procéder comme le montre les tableaux *Tableau 6*, *Tableau 7*, *Tableau 8*, *Tableau 9*, *Tableau 10*, *Tableau 11* et *Tableau 12* suivants pour décrire chaque cas d'utilisation.

*Tableau 6 : Description du cas d'utilisation « créer compte étudiant »*

| **Nom** | **Créer compte étudiant** |
|---|---|
| **Résumé** | C'est une fonctionnalité qui permet de créer des comptes pour les étudiants |
| **Acteurs** | Client distributeur, client partenaire |
| **Précondition** | Etre partenaire d'Univ-Money en demandant une intégration de l'API dans son système afin d'avoir la documentation pour utiliser le service. |
| **Scenario nominal** | • L'utilisateur effectue une demande depuis son serveur vers le serveur d'Univ-Money via l'url **localhost:5000/api/create/account/standard**.<br>• Le serveur d'Univ-Money vérifie l'en-tête de la requête afin de pouvoir s'assurer que c'est le bon utilisateur et que les paramètres sont complets et correctes.<br>• Le serveur d'Univ-Money demande une autorisation au système d'API-money.<br>• Le serveur d'API-money autorise la demande<br>• Le serveur d'API-money crée le compte<br>• Le système enregistre les informations dans la base de données d'Univ-Money<br>• Le serveur d'Univ-Money renvoie une réponse pour faire savoir que le compte est créé avec succès. |
| **Exceptions** | Une exception est levée lorsqu'une demande est effectuée avec un en-tête d'autorisation incorrecte ou avec des paramètres incorrects. |
| **Post-condition** | message d'information de la création de compte |

*Tableau 7 : Description du cas d'utilisation « créer compte distributeur »*

| **Nom** | **Créer compte distributeur** |
|---|---|
| **Résumé** | C'est une fonctionnalité qui permet de créer des comptes pour les distributeurs de la monnaie électronique. |
| **Acteurs** | Client administrateur |
| **Précondition** | Etre partenaire d'Univ-Money en demandant une intégration de l'API dans son système afin d'avoir la documentation pour utiliser le service. |
| **Scenario nominal** | • L'utilisateur effectue une demande depuis son serveur vers le serveur d'Univ-Money via l'url **localhost:5000/api/create/account/distributeur**.<br>• Le serveur d'Univ-Money vérifie l'en-tête de la requête afin de pouvoir s'assurer que c'est le bon utilisateur et que les paramètres sont complets et correctes.<br>• Le serveur d'Univ-Money demande une autorisation au système d'API-money.<br>• Le serveur d'API-money autorise la demande<br>• Le serveur d'API-money crée le compte<br>• Le système enregistre les informations dans la base de données d'Univ-Money<br>• Le serveur d'Univ-Money renvoie une réponse pour faire savoir que le compte est créé avec succès. |
| **Exceptions** | Une exception est levée lorsqu'une demande est effectuée avec un en-tête d'autorisation incorrecte ou avec des paramètres incorrects. |
| **Post-condition** | message d'information de la création de compte |

*Tableau 8 : Description du cas d'utilisation « créer compte partenaire »*

| **Nom** | **Créer compte partenaire** |
|---|---|
| **Résumé** | Cette fonctionnalité est mise en place pour la création de compte au profit des structures de l'Université (Scolarité, CROUS/Z et Direction des bourses) qui veulent utiliser Univ-Money dans leurs systèmes d'information. |
| **Acteurs** | Client administrateur |
| **Précondition** | Etre partenaire d'Univ-Money en demandant une intégration de l'API dans son système afin d'avoir la documentation pour utiliser le service. |
| **Scenario nominal** | • L'utilisateur effectue une demande depuis son serveur vers le serveur d'Univ-Money via l'url **localhost:5000/api/create/account/partenaire**.<br>• Le serveur d'Univ-Money vérifie l'en-tête de la requête afin de pouvoir s'assurer que c'est le bon utilisateur et que les paramètres sont complets et correctes.<br>• Le serveur d'Univ-Money demande une autorisation au système d'API-money.<br>• Le serveur d'API-money autorise la demande<br>• Le serveur d'API-money crée le compte<br>• Le système enregistre les informations dans la base de données d'Univ-Money<br>• Le serveur d'Univ-Money renvoie une réponse pour faire savoir que le compte est créé avec succès. |
| **Exceptions** | Une exception est levée lorsqu'une demande est effectuée avec un en-tête d'autorisation incorrecte ou avec des paramètres incorrects. |
| **Post-condition** | message de confirmation de la création de compte |

*Tableau 9 : Description du cas d'utilisation « créditer un compte »*

| **Nom** | **créditer un compte** |
|---|---|
| **Résumé** | C'est une fonctionnalité qui permet d'alimenter un porte-monnaie en monnaie électronique. Cette alimentation se fait par un virement de l'argent qui se trouve dans un compte bancaire vers le porte-monnaie électronique associé à ce compte bancaire. |
| **Acteurs** | client administrateur |
| **Précondition** | Etre partenaire d'Univ-Money en demandant une intégration de l'API dans son système afin d'avoir la documentation pour utiliser le service. |
| **Scenario nominal** | • L'utilisateur effectue une demande depuis son serveur vers le serveur d'Univ-Money via l'url **localhost:5000/api/operation/crediter**.<br>• Le serveur d'Univ-Money vérifie l'en-tête de la requête afin de pouvoir s'assurer que c'est le bon utilisateur et que les paramètres sont complets et correctes.<br>• Le serveur d'Univ-Money demande une autorisation au système d'API-money.<br>• Le serveur d'API-money autorise la demande<br>• Le serveur d'API-money crédite le compte<br>• Le système enregistre les informations dans la base de données d'Univ-Money<br>• Le serveur d'Univ-Money renvoie une réponse pour faire savoir que le compte est débité. |
| **Exceptions** | Une exception est levée lorsqu'une demande est effectuée avec un en-tête d'autorisation incorrecte ou avec des paramètres incorrects. |
| **Post-condition** | message de confirmation de la transaction |

*Tableau 10 : Description du cas d'utilisation « débiter un compte »*

| **Nom** | **débiter un compte** |
|---|---|
| **Résumé** | Cette fonctionnalité permet d'extraire de la monnaie électronique d'un compte et le mettre dans le compte bancaire qui lui est associé afin de pouvoir récupérer de l'argent en espèces. |
| **Acteurs** | Client administrateur |
| **Précondition** | Etre partenaire d'Univ-Money en demandant une intégration de l'API dans son système afin d'avoir la documentation pour utiliser le service. |
| **Scenario nominal** | • L'utilisateur effectue une demande depuis son serveur vers le serveur d'Univ-Money via l'url **localhost:5000/api/operation/debiter**.<br>• Le serveur d'Univ-Money vérifie l'en-tête de la requête afin de pouvoir s'assurer que c'est le bon utilisateur et que les paramètres sont complets et correctes.<br>• Le système vérifie que le montant est disponible dans le calpé<br>• Le serveur d'Univ-Money demande une autorisation au système d'API-money.<br>• Le serveur d'API-money autorise la demande<br>• Le serveur d'API-money débite le compte<br>• Le système enregistre les informations dans la base de données d'Univ-Money<br>• Le serveur d'Univ-Money renvoie une réponse pour faire savoir que le compte est débité. |
| **Exceptions** | Une exception est levée lorsqu'une demande est effectuée avec un en-tête d'autorisation incorrecte ou avec des paramètres incorrects. |
| **Post-condition** | message de confirmation de la transaction |

*Tableau 11 : Description du cas d'utilisation « transférer de l'argent »*

| **Nom** | **Transférer de l'argent** |
|---|---|
| **Résumé** | Cette fonctionnalité permet de faire des transferts de monnaie électronique entre les comptes Univ-Money. |
| **Acteurs** | Client Etudiant, Client Partenaire, Client Distributeur, Client Administrateur |
| **Précondition** | Etre partenaire d'Univ-Money en demandant une intégration de l'API dans son système afin d'avoir la documentation pour utiliser le service. |
| **Scenario nominal** | • L'utilisateur effectue une demande depuis son serveur vers le serveur d'Univ-Money via l'url **localhost:5000/api/operation/transfert**.<br>• Le serveur d'Univ-Money vérifie l'en-tête de la requête afin de pouvoir s'assurer que c'est le bon utilisateur et que les paramètres sont complets et correctes.<br>• Le système vérifie que le montant est disponible dans le calpé<br>• Le serveur d'Univ-Money demande une autorisation au système d'API-money.<br>• Le serveur d'API-money autorise la demande<br>• Le serveur d'API-money diminue et incrémente respectivement le solde du compte qui envoie et le solde du compte qui reçoit du montant de la transaction.<br>• Si le Client distributeur effectue le transfert vers un compte étudiant alors le solde de son compte commission est incrémenté de la valeur de la commission calculée pour cette transaction par le serveur d'API-money.<br>• Le système enregistre les informations dans la base de données<br>• Le serveur d'Univ-Money renvoie une réponse pour faire savoir que le transfert est effectif. |
| **Exceptions** | Une exception est levée lorsqu'une demande est effectuée avec un en-tête d'autorisation incorrecte ou avec des paramètres incorrects ou lorsque le solde du compte qui envoie est inférieur au montant du transfert |
| **Post-condition** | message de confirmation du transfert |

*Tableau 12 : Description du cas d'utilisation « consulter solde »*

| **Nom** | **Consulter solde** |
|---|---|
| **Résumé** | Ce cas d'utilisation permet de connaitre le solde d'un compte Univ-Money. |
| **Acteurs** | Client Etudiant, Client Partenaire, Client Distributeur, Client Administrateur |
| **Précondition** | Etre partenaire d'Univ-Money en demandant une intégration de l'API dans son système afin d'avoir la documentation pour utiliser le service. |
| **Scenario nominal** | • L'utilisateur effectue une demande depuis son serveur vers le serveur d'Univ-Money via l'url **localhost:5000/api/account/solde/{numero_cpt}**.<br>• Le serveur d'Univ-Money vérifie l'en-tête de la requête afin de pouvoir s'assurer que c'est le bon utilisateur<br>• Le serveur d'Univ-Money demande une autorisation au système d'API-money.<br>• Le serveur d'API-money autorise la demande<br>• Le serveur Univ-Money retourne le solde du compte<br>• Le serveur d'Univ-Money renvoie une réponse contenant le solde du compte |
| **Exceptions** | Une exception est levée lorsqu'une demande est effectuée avec un en-tête d'autorisation incorrecte. |
| **Post-condition** | message d'information sur le solde. |

Les cas d'utilisations décrits ci-dessus sont les plus important. En effet, en plus d'ajouter un utilisateur, un compte ou une opération, le système permet de modifier, de supprimer et de voir le contenu de chaque enregistrement présent dans la base de données.

Ce chapitre nous a permis d'exposer le rôle de la capture des besoins fonctionnels du processus 2TUP. Les diagrammes de cas d'utilisation nous ont permis de préciser l'étude du contexte du système en décrivant les différentes façons dont disposeront les acteurs pour utiliser le futur système.

Dans la partie qui suit nous allons faire l'analyse des besoins fonctionnels en montrant les différents événements qui permettront aux utilisateurs de bien profiter du système.

# II. Analyse des besoins fonctionnels du système

L'analyse des besoins fonctionnels est la phase qui permet de mettre en évidence ce que va réaliser le futur système en terme de métier. Cette phase s'appuie sur des séquences spécifiques d'actions et d'interactions entre les acteurs du système. Elle utilise donc les diagrammes de séquences qui mettent l'accent sur la chronologie des envois de message et les diagrammes d'activités qui définissent avec précision les traitements qui ont cours au sein du système. Les fonctionnalités du système étant assez nombreuses, nous allons donc présenter dans cette partie, les diagrammes de séquences et les diagrammes d'activités de quelques fonctionnalités.

## 1. Le cas d'utilisation « créer compte étudiant »

Cette fonctionnalité permet d'associer à chaque étudiant un compte Univ-Money pour pouvoir bénéficier des services offerts par le système. Elle est composée de plusieurs activités. Le client qui veut utiliser Univ-Money dans son système fait d'abord une demande d'intégration du système Univ-Money afin de disposer de la documentation et des outils pour utiliser Univ-Money. Ensuite l'utilisateur fait une requête depuis son serveur vers le serveur d'Univ-Money via l'URI **localhost:5000/api/create/account/standard**. Le serveur d'Univ-Money vérifie l'en-tête de la requête afin de pouvoir s'assurer que c'est le bon utilisateur et que les paramètres sont complets et correctes. Au cas échéant, il demande une autorisation au système d'API-money. Sinon il envoie un message d'erreur à l'utilisateur. Une fois la demande autorisée, le serveur d'API-money crée le compte et le système enregistre les informations dans la base de données d'Univ-Money. Sinon un message d'erreur est envoyé à l'utilisateur. Enfin le serveur d'Univ-Money envoie une réponse à la demande de l'utilisateur pour faire savoir que le compte est créé avec succès.

Les diagrammes d'activités et de séquences de ce cas d'utilisation sont représentés respectivement par les figures Figure 6 et Figure 7 suivantes.

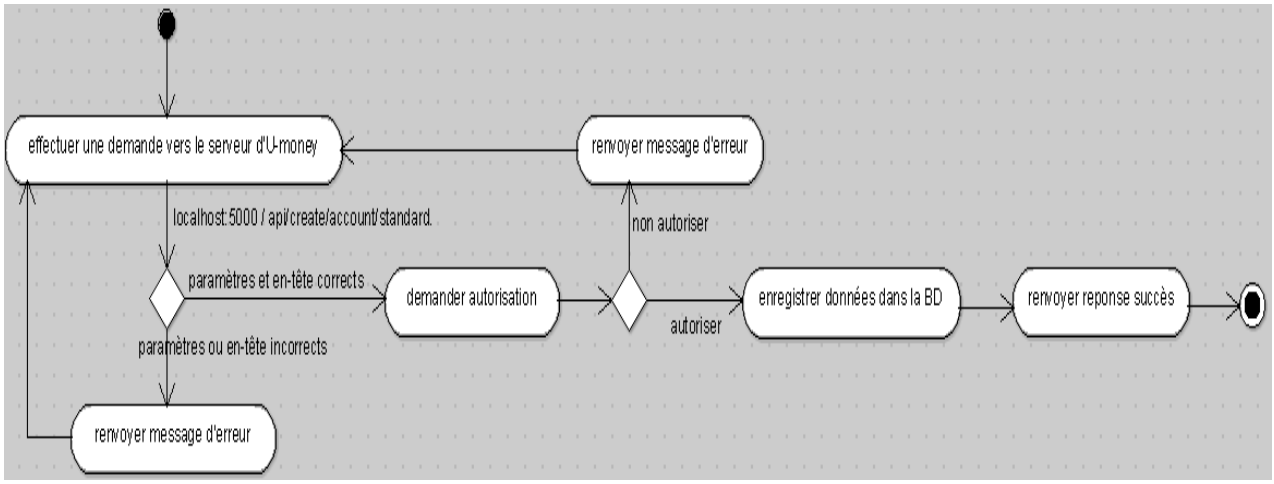


*Figure 6 : diagramme d'activités du cas d'utilisation « créer compte étudiant »*

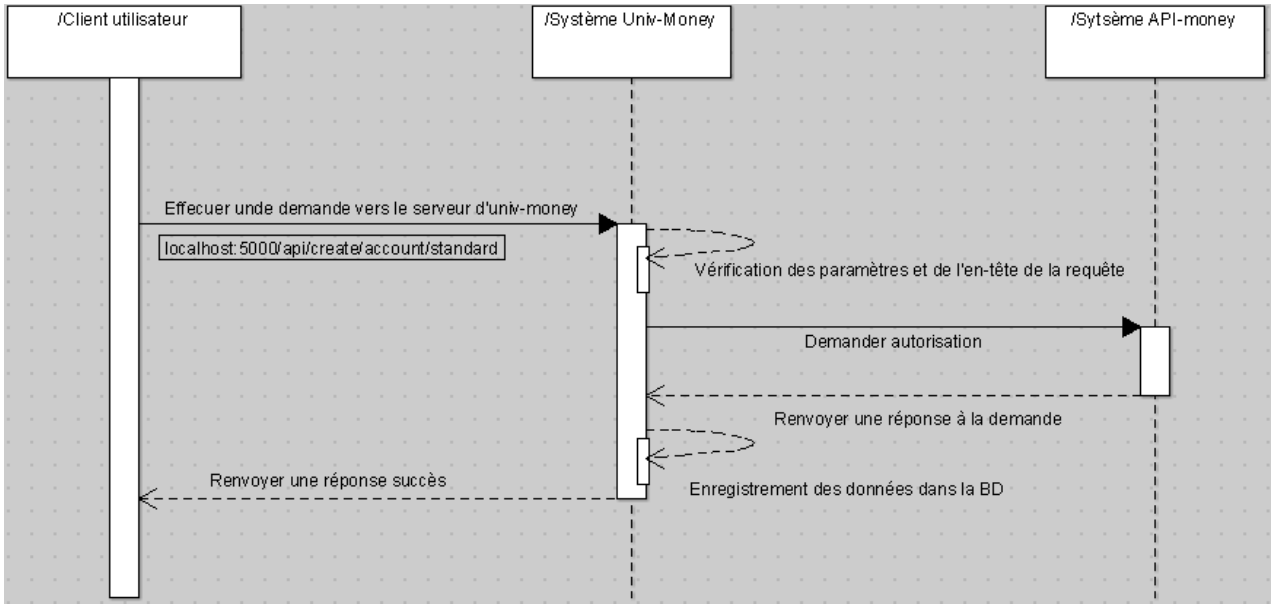


*Figure 7 : diagramme de séquences du cas d'utilisation « créer compte étudiant »*

## 2. Le cas d'utilisation « transférer de l'argent »

Cette fonctionnalité permet de faire des transferts de monnaie électronique entre les comptes Univ-Money. Elle se réalise en suivant plusieurs étapes. Le client qui veut utiliser Univ-Money dans son système fait d'abord une demande d'intégration du système Univ-Money afin de disposer de la documentation et des outils pour utiliser Univ-Money. Ensuite l'utilisateur fait une requête depuis son serveur vers le serveur d'Univ-Money via l'URI **localhost:5000/api/create/account/standard**. Le serveur d'Univ-Money vérifie que la demande est correcte. Au cas échéant, il demande une autorisation au système d'API-money.

Sinon il envoie un message d'erreur à l'utilisateur. Une fois la demande autorisée, le serveur d'API-money exécute le transfert et le système enregistre les informations dans la base de données d'Univ-Money. Sinon un message d'erreur est envoyé à l'utilisateur. Enfin le serveur d'Univ-Money envoie une réponse à la demande de l'utilisateur pour faire savoir que le compte est créé avec succès.

Les diagrammes d'activités et de séquences de ce cas d'utilisation sont représentés respectivement par les figures Figure 8 et Figure 9 suivantes.

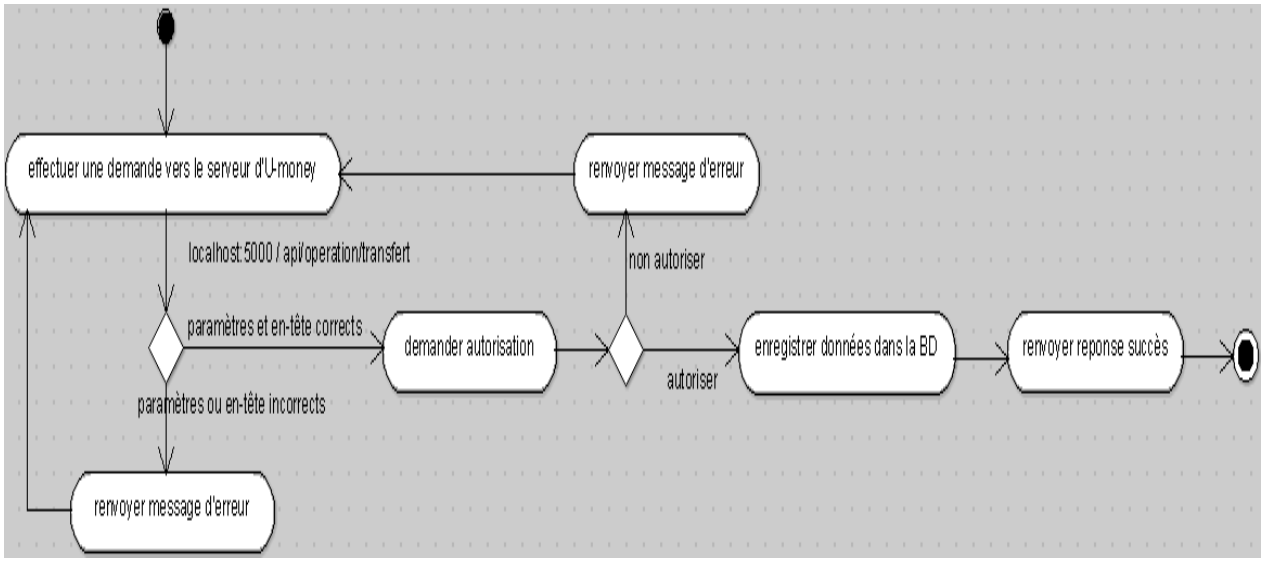


*Figure 8 : diagramme d'activités du cas d'utilisation « transférer de l'argent »*

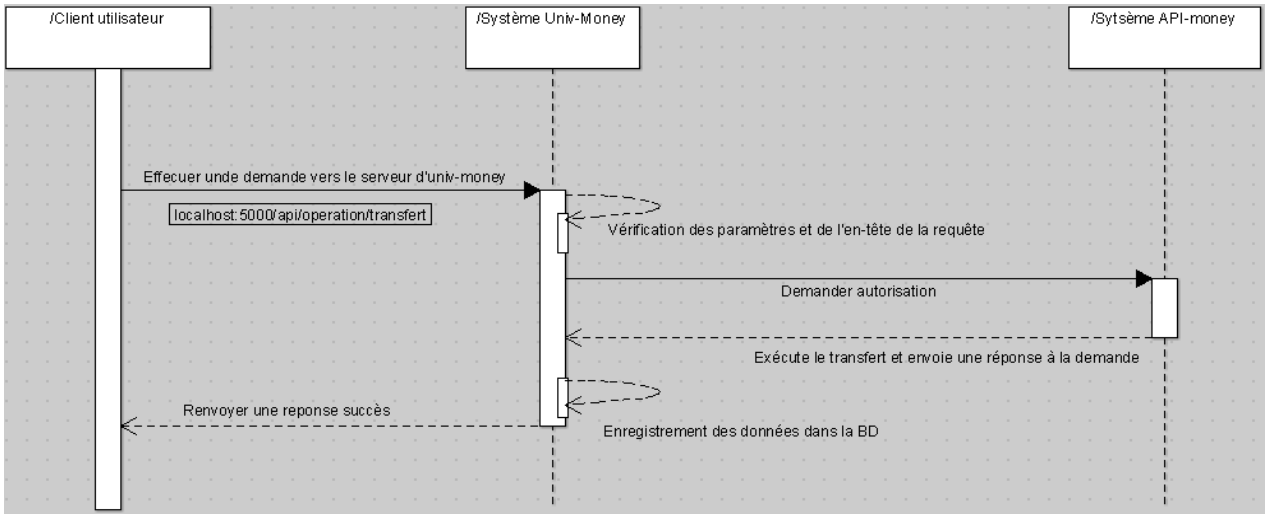


*Figure 9: diagramme de séquences du cas d'utilisation « transférer de l'argent»*

Dans cette partie nous avons analysé la capture des besoins fonctionnels du système en mettant en évidence, à l'aide des diagrammes de séquences et d'activités, quelques événements qui permettront aux utilisateurs de bien utiliser le système. Nous allons passer à la branche de droite du modèle 2TUP en attaquant le chapitre suivant qui consiste à la capture des besoins techniques et la conception du système.

# Chapitre IV : Capture des besoins techniques et conception du système

## I. Capture des besoins techniques

La capture des besoins techniques couvre, par complémentarité avec celle des besoins fonctionnels, toutes les contraintes qui ne traitent ni de la description du métier des utilisateurs, ni de la description applicative. Cette étape a lieu lorsque les architectes ont obtenu suffisamment d'informations sur les prérequis techniques. Ils doivent a priori connaître au moins le matériel, à savoir les machines et réseaux, les progiciels à intégrer, et les outils retenus pour le développement [6]. Dans cette partie, nous abordons la configuration matérielle du système ensuite l'élaboration du modèle de spécification logicielle et en fin l'organisation en couche du modèle de spécification logicielle.

### 1. Configuration matérielle du système Univ-Money

La configuration géographique du système Univ-Money impose le développement d'une solution client/serveur à trois niveaux : un niveau d'affichage et de traitements locaux, un niveau de traitements applicatifs et un niveau de services de base de données. La configuration matérielle est schématisée par le diagramme de déploiement UML à la Figure 10 suivante.

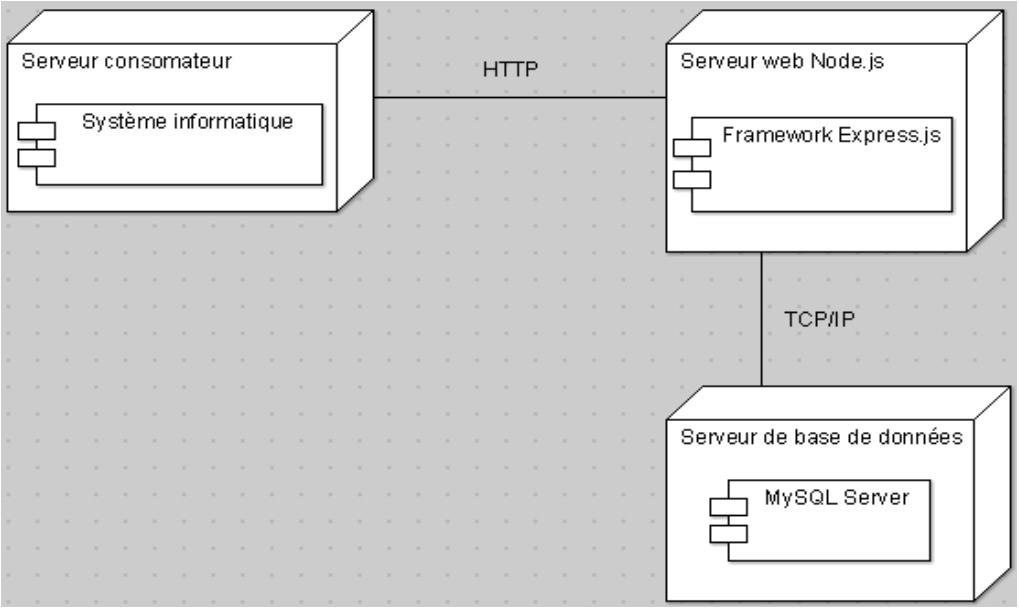


*Figure 10 : diagramme de déploiement du système Univ-Money*

## 2. Elaboration du modèle de spécification logicielle

Une fois que les spécifications techniques et d'architecture sont exprimées, on peut s'intéresser aux fonctionnalités propres du système technique en procédant à une spécification logicielle. Dans ce cadre, on propose d'utiliser les cas d'utilisation de manière différente que pour la spécification fonctionnelle. C'est pourquoi nous avons introduit le concept d'exploitant et de cas d'utilisation technique.

### a. Les exploitants

Les exploitants, appelés aussi « acteurs techniques » du système, sont les acteurs qui bénéficient des fonctionnalités techniques du système Univ-Money :

- Client utilisateur : ce sont les systèmes qui utilisent d'une manière ou d'une autre les fonctionnalités du système.
- Client administrateur : c'est le système chargé de gérer toute la gestion de la plateforme ainsi que de celle des utilisateurs.

### b. Identification des cas d'utilisation techniques

Un cas d'utilisation technique est destiné à l'exploitant. C'est une séquence d'actions produisant une valeur ajoutée opérationnelle ou purement technique. Les cas d'utilisation techniques sont absolument distincts des cas d'utilisation de la branche gauche du modèle en Y : ils ne produisent aucune valeur ajoutée fonctionnelle. La branche droite du modèle en Y recouvre en effet tous les services techniques dont un utilisateur bénéficie, parfois même sans s'en rendre compte. Les cas d'utilisation techniques du système Univ-Money sont d'abord identifiés en considérant l'attente opérationnelle de chaque exploitant [6]:

- l'utilisateur va travailler avec des entités sous la forme d'objets, ce qui implique la mise en œuvre des mécanismes de persistance et de gestion de cycle de vie des objets ;
- plusieurs systèmes utilisateurs utilisent les mêmes objets ce qui implique un mécanisme de synchronisation ;
- plusieurs utilisateurs peuvent travailler en parallèle. L'intégrité est le mécanisme qui empêche la mise à jour simultanée d'une même entité par deux utilisateurs différents ;
- chaque utilisateur bénéficie également d'une gestion des charges au niveau du serveur. Ainsi, les temps de réponse du système ne s'en trouvent pas dégradés en fonction du nombre d'utilisateurs connectés ;

- l'utilisateur doit envoyer des requêtes avec un en-tête autorisation qui contient la clé api qui lui permet d'être reconnu du système pour pouvoir y travailler. L'authentification est le mécanisme qui protège le système des intrusions externes ;
- chaque utilisateur doit disposer d'une aide contextuelle qui l'aide à exploiter le système de la manière la plus efficace ;
- le système doit être exploitable ; à ce titre, il faut qu'il soit en mesure de générer des traces et des alertes qui vont faciliter sa maintenance. C'est cette analyse technique du problème qui permet d'introduire le client administrateur comme exploitant du système ;
- les utilisateurs sont soumis à des règles de sécurité. Dans un système client-serveur, ces aspects recouvrent l'authentification, l'habilitation, le cryptage, la non-répudiation et l'audit.

Ces contraintes d'utilisation techniques donnent lieu à un premier modèle de spécification logicielle représenté par le diagramme de cas d'utilisation de la Figure 11 ci-dessous.

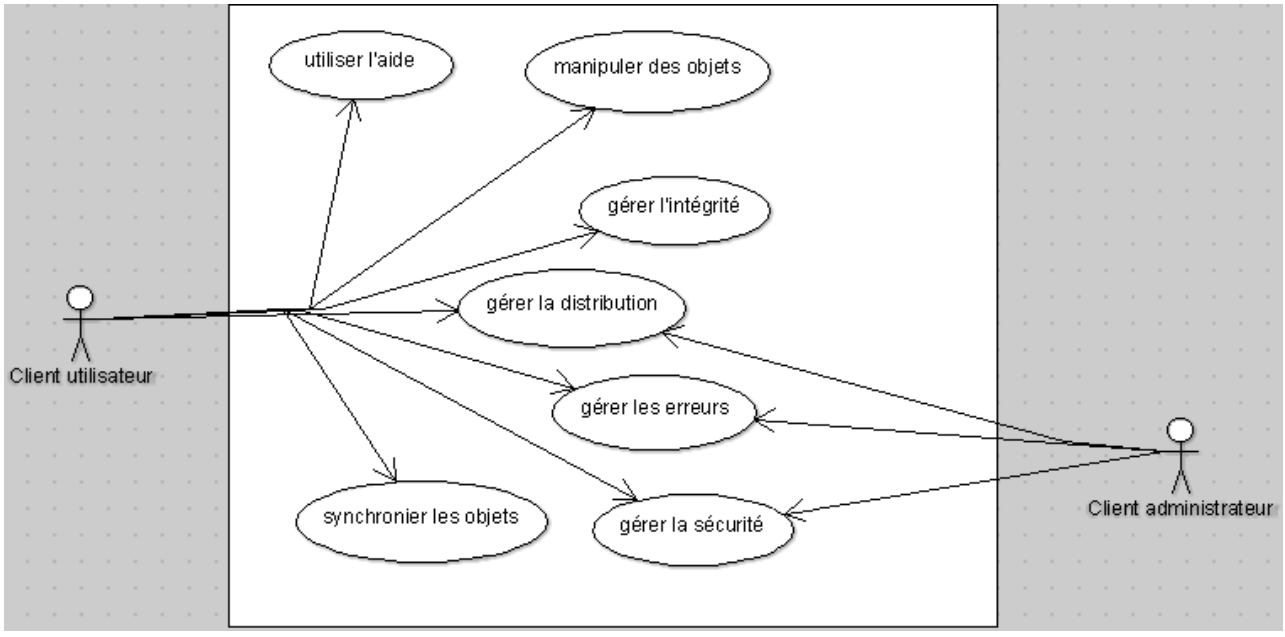


*Figure 11 : modèle de spécification logicielle du système Univ-Money*

## 3. Organisation en couche du modèle de spécification logicielle

Une couche logicielle représente un ensemble de spécifications ou de réalisations qui respectivement expriment ou mettent en œuvre un ensemble de responsabilités techniques et homogènes pour un système logiciel. Les couches s'empilent en niveaux pour couvrir des

transformations logicielles successives, de sorte que la couche d'un niveau ne puisse utiliser que les services des couches des niveaux inférieurs. Le style d'architecture en couches préconisé pour le développement d'un système est représenté par la Figure 12 suivante en illustrant le rôle et la description des cinq couches logicielles [6].

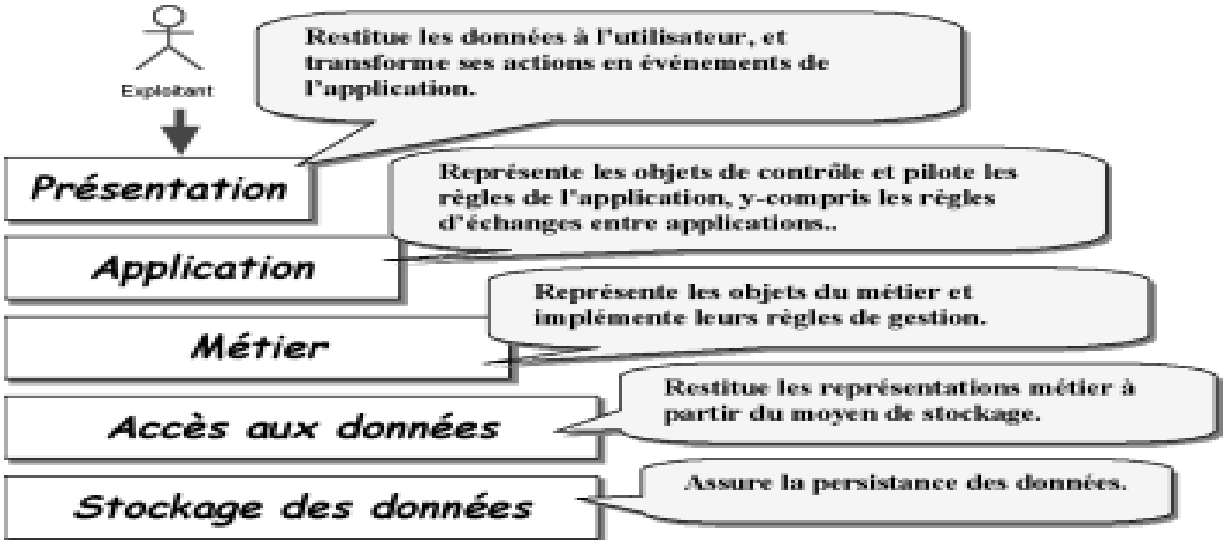


*Figure 12 : organisation en couche du modèle de spécification logicielle du système Univ-Money*

Cette partie nous a permis de sortir les spécifications techniques du système Univ-Money. Nous avons maintenant les éléments nécessaires à la conception générique. Dans la partie qui suit nous allons passer à la conception générique de système.

## II. Conception générique du système

La conception générique consiste à développer la solution qui répond aux spécifications techniques que nous avons présentées au chapitre précédent. Cette conception est qualifiée de générique car elle est entièrement indépendante des aspects fonctionnels spécifiés en branche gauche. Dans cette partie nous abordons d'abord l'élaboration du modèle logique de conception technique pour ensuite présenter une organisation de ce dernier.

### 1. Elaboration du modèle logique de conception technique

Les classes, interfaces et frameworks techniques sont les briques de construction d'un modèle logique.

### c. Les classes

L'intégrité de conception s'exprime sous la forme d'un ensemble de classes techniques que les concepteurs du projet vont par la suite réutiliser pour développer les différentes composantes fonctionnelles du système. À titre d'illustration, le mécanisme de contrôle des transactions peut être conçu par un ensemble de classes techniques réutilisées, quelle que soit l'application envisagée dans Univ-Money. On constate cependant qu'une classe technique fonctionne rarement seule, c'est pourquoi le concept-clé de la conception générique est le framework technique.

### d. Les frameworks techniques

Un framework est un réseau de classes qui collaborent à la réalisation d'une responsabilité qui dépasse celle de chacune des classes qui y participent. Un framework technique ne concerne que les responsabilités de la branche droite du processus.

### e. Organisation du modèle logique de conception technique

L'organisation du modèle logique reprend les couches logicielles. À chaque couche correspond un framework technique, en partie abstrait, qui définit des interfaces de réalisation des responsabilités logicielles comme représenté dans la Figure 13 ci-dessous [6]:

- Framework présentation : définit les classes, interfaces, mécanismes de base pour réaliser la gestion des objets.
- Framework applicatif : il utilise les mêmes éléments pour rafraîchir les vues, charger les modèles de fonctionnement et contrôler les commandes.
- Framework métier : définit les éléments permettant d'identifier les objets métier et de les gérer en services distribués.
- Framework accès aux données : définit les mécanismes de chargement, sauvegarde, mise à jour et de recherche d'objets dans une BDD.

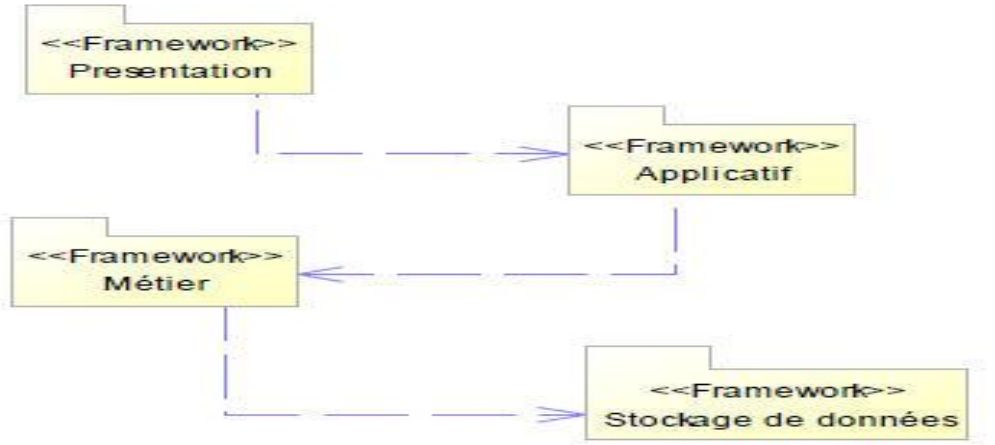


*Figure 13 : Organisation des frameworks techniques du système Univ-Money*

Le découpage logique du système Univ-Money est représenté par le diagramme de paquetage, de la Figure 14 ci-dessous, composé des trois paquetages suivants :

- Gestion des utilisateurs ;
- Gestion des opérations ;
- Gestion des comptes.

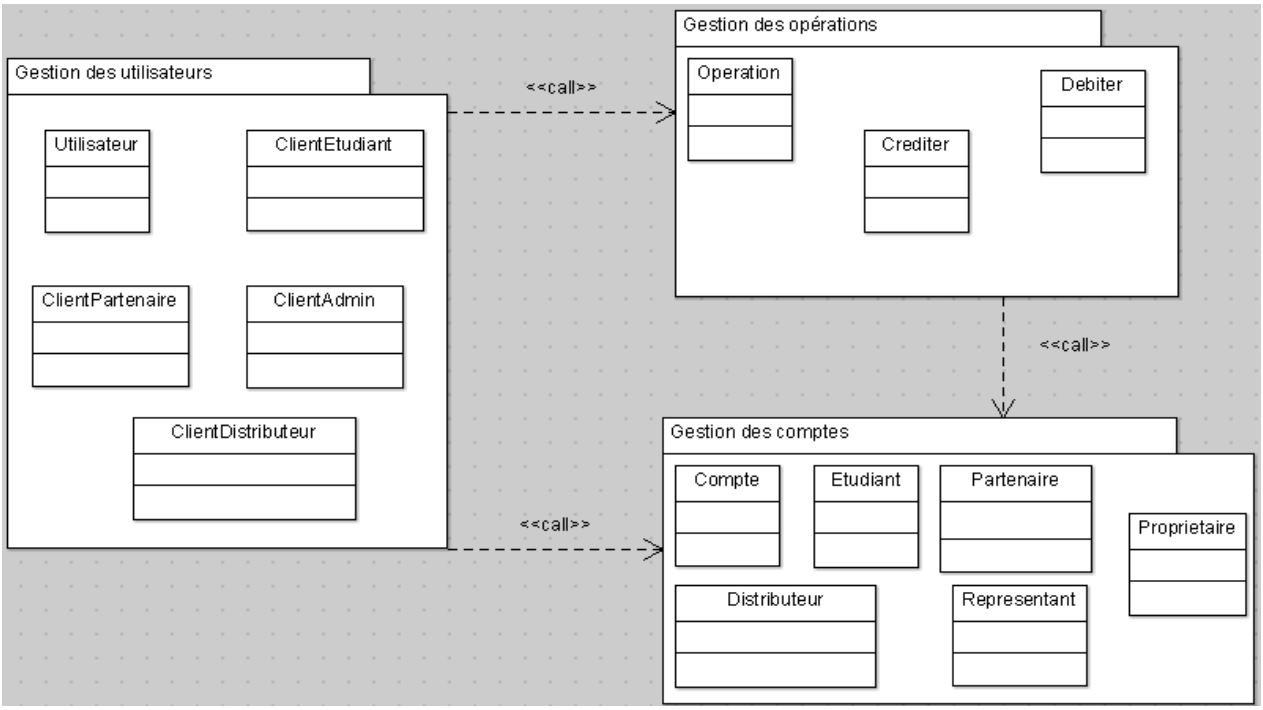


*Figure 14 : diagramme de paquetage du système Univ-Money*

Dans la partie qui suit nous allons faire la conception préliminaire du système.

# III. Conception préliminaire du système

La conception préliminaire est l'étape la plus importante du processus 2TUP vu qu'elle représente le cœur. En effet dans cette étape nous allons enfin quitter les deux branches droite et gauche afin de faire la fusion entre les deux études technique et fonctionnelle. Dans cette partie nous allons présenter l'architecture REST qui va nous permettre de concevoir nos services web suivi de l'architecture du système Univ-Money et en fin le diagramme de composants du système.

## 1. Architecture REST

### a. Généralité de l'architecture REST

REST (REpresentational State Transfer) est une architecture basée sur des standards Web et utilise le protocole HTTP. Il tourne autour de la ressource où chaque composant est une ressource et une ressource est accessible par une interface commune à l'aide de méthodes standard HTTP. REST a été introduit pour la première fois par Roy Fielding en 2000 [7].

Dans l'architecture REST, un serveur REST fournit simplement un accès aux ressources et le client REST accède et modifie les ressources. Ici, chaque ressource est identifiée par des URI / ID globaux. REST utilise diverses représentations pour représenter une ressource comme le texte, JSON, XML. JSON est le plus populaire.

Les quatre méthodes HTTP suivantes sont couramment utilisées dans l'architecture basée sur REST.

- GET : Fournit un accès en lecture seule à une ressource.
- POST : Utilisé pour créer une nouvelle ressource.
- DELETE : Utilisé pour supprimer une ressource.
- PUT : Utilisé pour mettre à jour une ressource existante ou créer une nouvelle ressource.

La Figure 15 suivante est une représentation graphique de l'architecture REST.

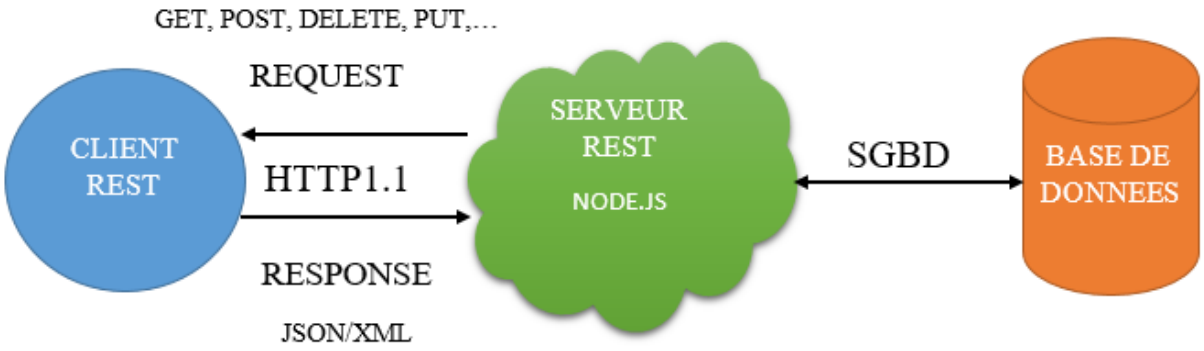


*Figure 15 : architecture REST*

L'architecture REST définit un ensemble de contraintes à utiliser pour créer des services web. Nous allons décrire ces contraintes dans la partie qui suit.

### b. Les contraintes architecturales de REST

Un système REST est défini par six contraintes architecturales. Ces contraintes restreignent la façon dont le serveur peut traiter et répondre aux requêtes du client afin que, en agissant dans ces contraintes, le système gagne des propriétés non fonctionnelles désirables, telles que la performance, l'extensibilité, la simplicité, l'évolutivité, la visibilité, la portabilité et la fiabilité. Un système qui viole une de ces contraintes ne peut pas être considéré comme adhérant à l'architecture REST [7].

#### *i. Client-serveur*

Les responsabilités sont séparées entre le client et le serveur. Découpler l'interface utilisateur du stockage des données améliore la portabilité de l'interface utilisateur sur plusieurs plateformes. L'extensibilité du système se retrouve aussi améliorée par la simplification des composants serveurs. Mais peut-être encore plus essentiel pour le Web, la séparation permet aux composants d'évoluer indépendamment, supportant ainsi les multiples domaines organisationnels nécessaires à l'échelle d'Internet.

#### *ii. Sans état*

La communication client–serveur s'effectue sans conservation de l'état de la session de communication sur le serveur entre deux requêtes successives. L'état de la session est conservé par le client et transmis à chaque nouvelle requête. Les requêtes du client contiennent donc toute l'information nécessaire pour que le serveur puisse y répondre. La visibilité des interactions entre les composants s'en retrouve améliorée puisque les requêtes sont complètes. La tolérance aux échecs est également plus grande. De plus, le fait de ne pas avoir à maintenir une connexion permanente entre le client et le serveur permet au serveur de répondre à d'autres requêtes venant d'autres clients sans saturer l'ensemble de ses ports de communication, ce qui améliore l'extensibilité du système.

Cependant une exception usuelle à ce mode sans état est la gestion de l'authentification du client, afin que celui-ci n'ait pas à renvoyer ces informations à chacune de ses requêtes.

### *iii. Mise en cache*

Les clients et les serveurs intermédiaires peuvent mettre en cache les réponses. Les réponses doivent donc, implicitement ou explicitement, se définir comme pouvant être mise en cache ou non, afin d'empêcher les clients de récupérer des données obsolètes ou inappropriées en réponse à des requêtes ultérieures. Une mise en cache bien gérée élimine partiellement voire totalement certaines interactions client–serveur, améliorant davantage l'extensibilité et la performance du système.

### *iv. En couche*

Un client ne peut habituellement pas dire s'il est connecté directement au serveur final ou à un serveur intermédiaire. Les serveurs intermédiaires peuvent améliorer l'extensibilité du système en mettant en place une répartition de charge et un cache partagé. Ils peuvent aussi renforcer les politiques de sécurité.

### *v. Code à la demande*

Les serveurs peuvent temporairement étendre ou modifier les fonctionnalités d'un client en lui transférant du code exécutable par exemple par des applets Java ou des scripts JavaScript. Cela permet de simplifier les clients en réduisant le nombre de fonctionnalités qu'ils doivent mettre en œuvre par défaut et améliore l'extensibilité du système. En revanche, cela réduit aussi la visibilité de l'organisation des ressources. De ce fait, elle constitue une contrainte facultative dans une architecture REST.

### *vi. Interface uniforme*

La contrainte d'interface uniforme est fondamentale dans la conception de n'importe quel système REST. Elle simplifie et découple l'architecture, ce qui permet à chaque composant d'évoluer indépendamment.

Les contraintes architecturales de REST confèrent aux systèmes qui les respectent les propriétés architecturales suivantes :

- Performance dans les interactions des composants, qui peuvent être le facteur dominant dans la performance perçue par l'utilisateur et l'efficacité du réseau.
- Extensibilité permettant de supporter un grand nombre de composants et leurs interactions.
- Simplicité d'une interface uniforme.
- Evolutivité des composants pour répondre aux besoins.

- Visibilité des communications entre les composants par des agents de service.
- Portabilité des composants en déplaçant le code avec les données.
- Fiabilité dans la résistance aux pannes du système en cas de pannes des composants, des connecteurs ou des données.

## 2. Architecture du système Univ-Money

L'architecture désigne la structure générale nécessaire et inséparable à un système informatique, l'organisation des différents éléments du système (logiciels, matériels, humains, informations) et des relations entre les éléments.

Notre système est composé par les éléments suivants :

- Une couche métier qui permet le traitement des opérations.
- Une couche présentation qui permet la communication avec les systèmes externes au système.
- Un serveur d'application Node.js.
- Un système de gestion de base de données pour l'utilisation de notre base de données où nous allons stocker nos données.
- Le système API-money mis en place par la société W-HA qui fournit les fonctionnalités du Calpé.
- Des clients externes qui utilisent les services fournis par le serveur Univ-money.

La Figure 16 ci-dessous montre l'architecture générale de notre système :

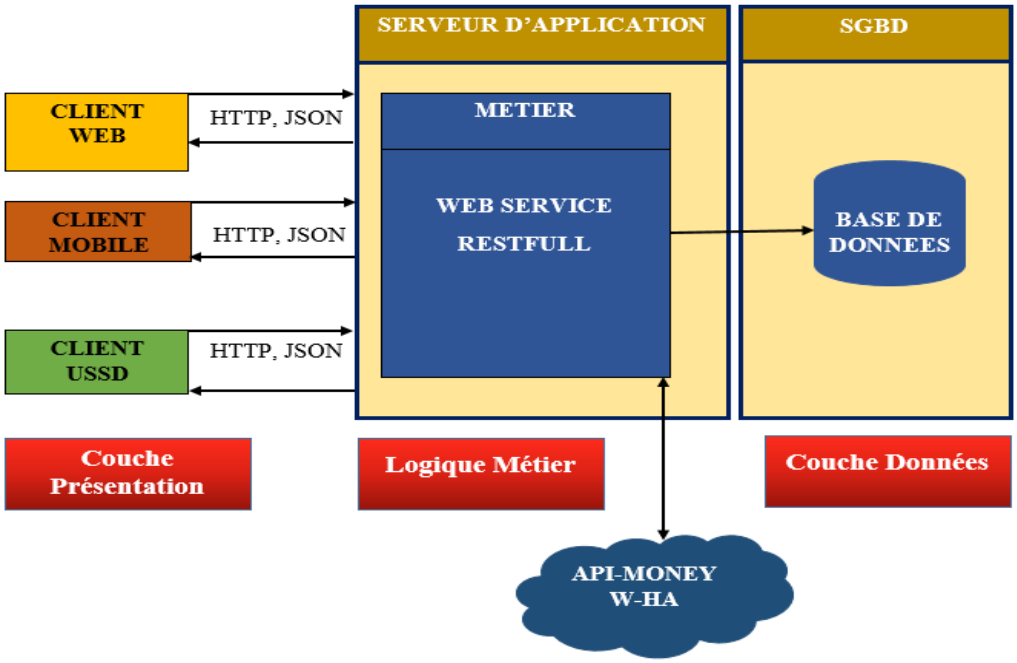


*Figure 16 : architecture générale du système Univ-Money.*

## 3. Le diagramme de composants

L'architecture du système nous permet de définir le diagramme de composant de notre système. Ce diagramme a comme rôle la définition des composants ainsi que leurs relations. Le diagramme de composants de notre système est illustré à la Figure 17 suivante.

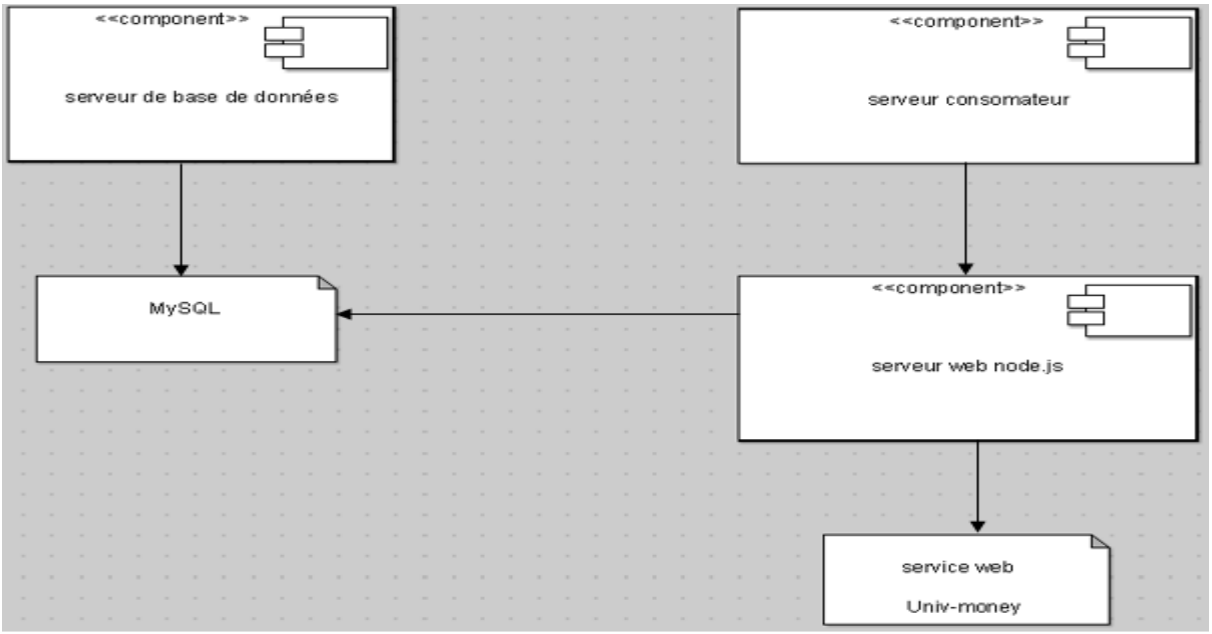


*Figure 17 : diagramme de composants du système Univ-Money.*

Une fois l'architecture est mise en place et que les composants sont sorties, on peut passer donc à la conception détaillée du système qui consiste à construire et à documenter de manière précise les classes et le modèle relationnel qui constitue le codage de la solution.

# IV. Conception détaillée du système

La conception détaillée est une activité qui s'inscrit dans l'organisation définie par la conception préliminaire. Le modèle logique y est particulièrement important dans la mesure où c'est en conception détaillée que l'on génère le plus gros volume d'informations. Dans cette partie nous allons établir le dictionnaire des données et ensuite présenter le diagramme de classes.

## 1. Dictionnaire des données

Le Tableau 13 ci-dessous représente la collection de données nécessaires à la conception de notre base de données.

*Tableau 13 : Dictionnaire des données*

| TABLE | ATTRIBUT | TYPE |
|---|---|---|
| **OPERATION** | OPERATIONID | INT |
| | DATEOPERATION | DATE |
| | TYPEOPERATION | ENUM ('DEPOT', 'RETRAIT') |
| | MONTANT | DOUBLE |
| **COMPTE** | COMPTEID | INT |
| | DATECREATION | DATE |
| | TYPECOMPTE | ENUM ('STANDARD', 'BUSINESS', 'INSTANCE UNIVERSITAIRE') |
| | SOLDE | DOUBLE |
| | ACCOUNT_LINK | VARCHAR(100) |
| | WALLET_LINK | VARCHAR(100) |
| **UTILISATEUR** | USERID | INT |
| | USERNAME | VARCHAR(100) |
| **PROPRIETAIRE** | PROPRIETAIREID | INT |
| | NOMCOMPLET | VARCHAR(100) |

| | | |
|---|---|---|
| | DATE_NAISSANCE | DATE |
| | TEL | VARCHAR(20) |
| | EMAIL | VARCHAR(150) |
| | NUMERO_INE | VARCHAR(20) |
| | CODESECRET | VARCHAR(10) |
| | ADRESSE | VARCHAR(150) |
| | BUSINESS_TYPE | VARCHAR(100) |
| | REGISTRATION_NUMBER | VARCHAR(100) |
| **REPRESENTANT** | REPRESENTANTID | INT |
| | NOM | VARCHAR(100) |
| | PRENOM | VARCHAR(100) |
| | DATE_NAISSANCE | DATE |

## 2. Diagramme de classes

Le diagramme de classes est un schéma utilisé en génie logiciel pour présenter les classes et les interfaces des systèmes ainsi que leurs relations. Ce diagramme fait partie de la partie statique d'UML, ne s'intéressant pas aux aspects temporels et dynamiques.

Une classe décrit les responsabilités, le comportement et le type d'un ensemble d'objets. Les éléments de cet ensemble sont les instances de la classe.

### a. Diagramme de classes participantes à la gestion des utilisateurs

Le diagramme de la Figure 18 ci-dessous permet de gérer les utilisateurs du système. Les classes Utilisateur, ClientAdministrateur, ClientDistributeur, ClientPartenaire et ClientEtudiant sont les classes participantes aux fonctionnalités de la gestion des utilisateurs.

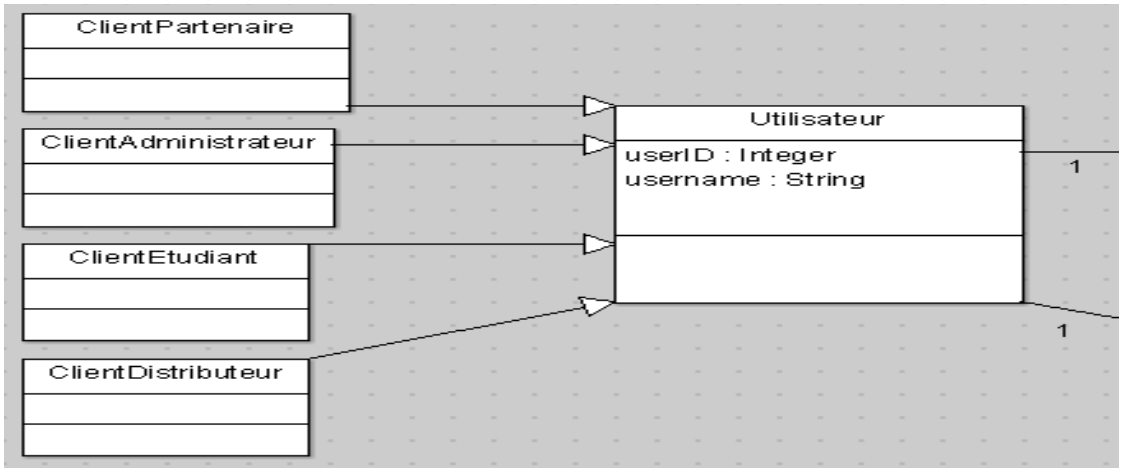


*Figure 18 : diagramme de classe participantes à la gestion des utilisateurs.*

### b. Diagramme de classes participantes à la gestion des comptes

Le diagramme de la Figure 19 ci-après permet de gérer les comptes relativement à leurs propriétaires et les éventuels représentants. Les classes Compte, Etudiant, Distributeur, Partenaire, Proprietaire, Representant sont les classes qui interviennent aux fonctionnalités de la gestion des comptes.

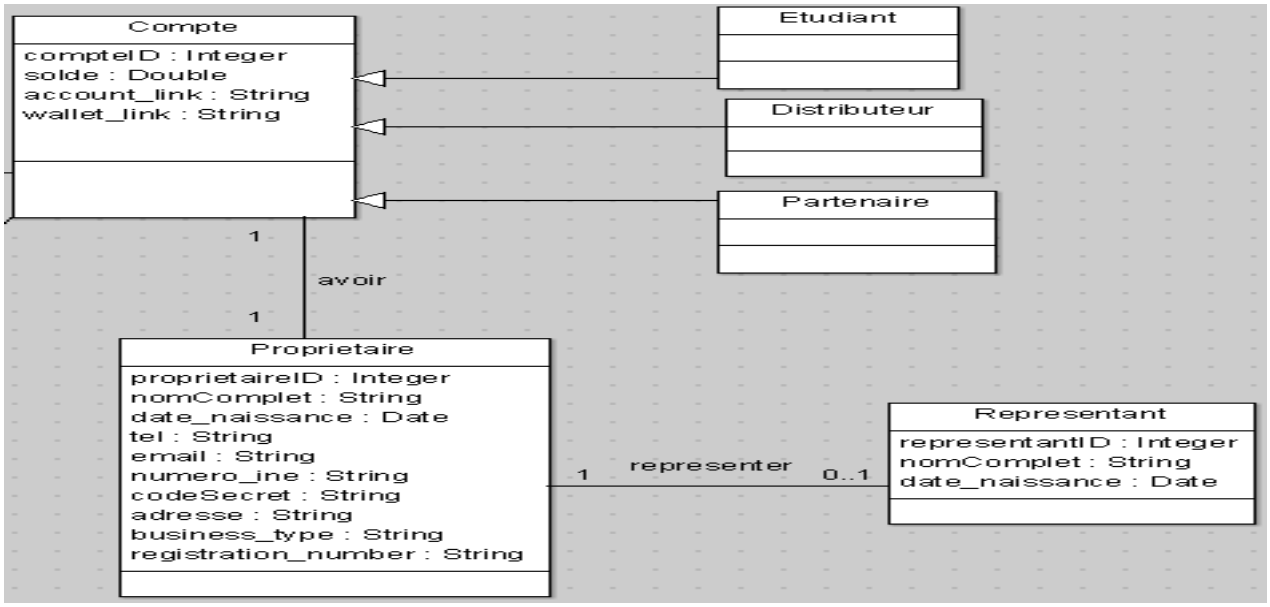


*Figure 19 : diagramme de classe participantes à la gestion des comptes.*

### c. Diagramme de classes participantes à la gestion des opérations

Le diagramme de la Figure 20 ci-dessous permet de gérer les différentes opérations effectuées dans le système. Les classes Operation, Crediter, Debiter sont les classes qui interviennent aux fonctionnalités de la gestion des opérations.

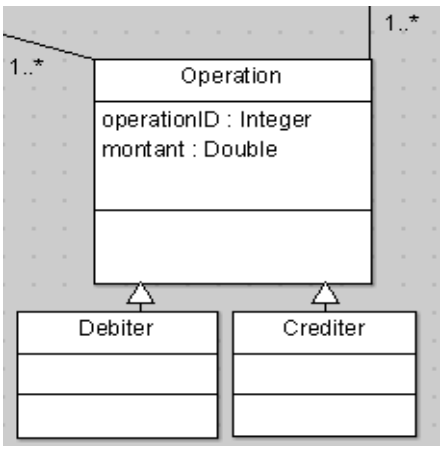


*Figure 20 : diagramme de classe participantes à la gestion des opérations.*

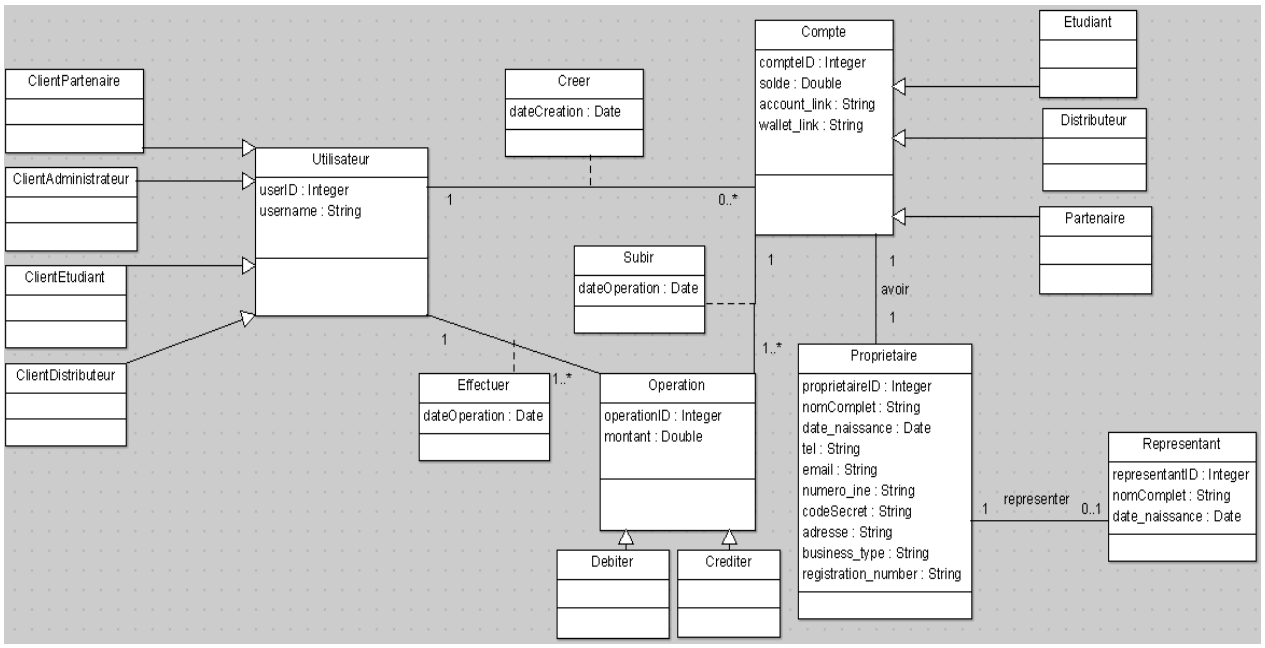


*Figure 21: diagramme de classes complet*

Dans ce chapitre nous avons évoqué la capture des besoins techniques et la conception du système où on a parlé de la capture des besoins techniques, de la conception préliminaire, de la conception générique et de la conception détaillée. Le travail fait dans cette partie nous facilite le passage à l'étape suivante où il s'agit du codage et de la présentation des services.

# Chapitre V : Implémentation des services d'Univ-Money

L'étape de codage permet d'effectuer la réalisation des services web et les tests des unités de code au fur et à mesure de leur réalisation. La fin du codage fera l'objet d'une présentation des services. Dans cette partie nous allons montrer l'utilisation des services de l'API-money, ensuite les technologies et les outils utilisés pour l'édition des codes, la gestion de la base de données et les tests et enfin l'implémentation de la base de données et des services.

## I. Utilisation de l'API-money

Le partenaire reçoit les éléments techniques suivants pour tester la solution API-money :

- URL de la plate-forme de test : https://test-emoney-services.w-ha.com/api/
- URL Documentation associée : https://www.API-money.com/docs/
- Les credentials api_access_key et api_secret_key qui sont reçus par mail

En même temps que la création des credentials, un compte partenaire est créé au profit partenaire qui est Univ-Money dans notre cas. Il s'agit d'un compte spécifique qui permettra d'y associer un wallet de type EMONEY pour la distribution de monnaie électronique et un wallet de type FEES, pour récupérer les éventuelles commissions, lors des opérations de cash-in, transfert ou cash-out. Univ-Money dispose d'une documentation qui lui permet d'alimenter son compte de type EMONEY en monnaie électronique via un encaissement par carte de crédit ou par virement bancaire afin de pouvoir distribuer de la monnaie électronique [8].

Cette partie a pour objectif de montrer l'utilisation de l'API-money en indiquant les principales requêtes. Elle ne décrit pas l'ensemble des requêtes et ne reflète pas l'ensemble des possibilités offertes par API-money. Pour des mesures de sécurité, chaque requête nécessite une authentification.

### 1. Authentification

Toutes les demandes envoyées vers le serveur d'API-money doivent être entièrement authentifiées via l'en-tête HTTP «Authorization». Elles doivent être envoyées directement depuis le serveur du partenaire vers le serveur d'API-money et non depuis le navigateur ou l'application d'un utilisateur. Cette authentification est assurée par le calcul d'un HMAC, sur la

base des credentials et d'un timestamp. Le contenu de l'en-tête doit respecter le format décrit dans le Tableau 14 ci-dessous.

```
AUTHORIZATION: AUTH API_ACCESS_KEY:TIMESTAMP:VERSION:SIGN
```

*Tableau 14 : paramètres de hachage*

| Propriété | Type | Valeur | Description |
|---|---|---|---|
| Api_access_key | String | taille = 32 max | Clé API (fournie par W-HA) |
| Timestamp | Long | Ne doit pas vieillir plus de 5 minutes | Horodatage de la demande en millisecondes |
| Version | Integer | | Numéro de version du processus d'authentification |
| Sign | String | | Signature HMAC-SHA256 |

**Calcul de signature**

Le paramètre «sign» est un HMAC-SHA256 des paramètres suivants séparés par «:» :

- api_access_key,
- timestamp,
- version,
- corps de la requête (peut être vide dans certains cas).

Une clé secrète partagée (fournie par W-HA) appelée « api_secret_key » est utilisée pour calculer le hachage.

- StringToSign = api_access_key: horodatage: version: request_body
- Sign = HMAC-SHA256 (StringToSign, api_secret_key)

**Exemple :**

- api_access_key = OLqMu27t1mylpc2D
- api_secret_key = YMy7t54-WaF9F!LOSp994p1?0x8pUp
- timestamp = 1494862655078
- version = 1
- request_body = {"tag":"my_new_tag"}
- StringToSign = OLqMu27t1mylpc2D:1494862655078:1:{"tag":"my_new_tag"}
- Sign = 2b6cf86e9f3d5c50a5b7f79aa10c9ce6da1fcd31211bf87374347274c168cf01

L'en-tête « Authorization » de la requête est de la forme :

```
Authorization:AUTH
OLqMu27t1mylpc2D:1494862655078:1:2b6cf86e9f3d5c50a5b7f79aa10c9ce6da1fcd31211bf
87374347274c168cf01
```

## 2. Création de compte

La création de compte dépend du type de compte qu'on doit créer soit STANDARD ou BUSINESS. Dans le contexte d'Univ-Money, les comptes STANDARD sont destinés aux étudiants et les comptes BUSINESS sont au profit des structures de l'Université.

Pour créer un compte STANDARD ou particulier on effectue une demande depuis notre serveur vers le serveur d'API-money via l'opération ***POST /accounts/standard***. Le serveur d'Univ-Money doit fournir les informations personnelles du propriétaire du compte. Ces informations constituent le corps de la requête avec l'adresse mail ou le numéro de téléphone du propriétaire obligatoire. Elles sont envoyées de la forme suivante.

```
POST /accounts/standard
{
    "subscriber" :{
        "lastname" : "Martin",
        "firstname" : "Philippe",
        "birthdate" : "1986-03-01",
        "birth_country" : "FRA",
        "birth_city" : "Paris",
        "nationality" : "FRA",
        "citizen_us" : false,
        "fiscal_us" : false,
        "fiscal_out_france" : true
    },
    "address" :{
        "label1" : "12 rue de Stalingrad",
        "zip_code" : "92800",
        "city" : "Puteaux",
        "country" : "FRA"
    },
    "email" : "m.philippe@wha.fr",
    "tag" : "account_type1"
}
```

*Figure 22 : corps de la requête de création de compte STANDARD*

Pour créer un compte BUSINESS ou professionnel on effectue une demande depuis notre serveur vers le serveur d'API-money via l'opération ***POST /accounts/business***. Le serveur d'Univ-Money doit fournir les informations relatives à la société propriétaire du compte ainsi

de même que celles de son représentant c'est-à-dire celui qui gère le compte de la société. Ces informations constituent le corps de la requête. Elles sont envoyées de la forme suivante.

```
POST /accounts/business
{
    "name" : "Association Sportive P10",
    "business_type" : "ASSOCIATION",
    "email" : "asw.contact@wha.fr",
    "registration_number" : "100018757",
    "phone_number" : "33129541388",
    "representative" : {
        "lastname" : "Julien",
        "firstname" : "Dore",
        "birthdate" : "1970-12-01",
        "nationality" : "FRA"
    },
    "address" : {
        "label1" : "88 rue Barthe",
        "zip_code" : "75010",
        "city" : "Paris",
        "country" : "FRA"
    },
    "tag" : "account_type2"
}
```

*Figure 23 : corps de la requête de création de compte BUSINESS*

Le serveur d'API-money envoie comme réponse l'identifiant du compte qui vient d'être créé. Cette réponse est de la forme suivante.

- Pour les comptes STANDARD :

```
HTTP/1.1 201 CREATED
{
    "id": "AS-459652557845695458"
}
```

- Pour les comptes BUSINESS :

```
HTTP/1.1 201 CREATED
{
    "id": "AB-459652557845695458"
}
```

Ces comptes doivent être associés à des wallets de type EMONEY pour qu'ils puissent bénéficier des services offerts par Univ-Money. Pour ce faire, on effectue une demande depuis notre serveur vers le serveur d'API-money via l'opération ***POST /wallets***. Le serveur d'Univ-Money envoie les informations à savoir identifiant du compte créé et le type de wallet à créer. Elles sont envoyées de la manière suivante.

```
POST /wallets
{
    "account_id" : "A-459652557845695458",
    "type" : "EMONEY"
}
```

*Figure 24 : corps de la requête d'association de compte à un wallet*

Le serveur d'API-money répond à son tour en envoyant l'identifiant du wallet créé de la forme

```
HTTP/1.1 201 CREATED
{
    "id" : "WE-787841204695485624"
}
```

.

### 3. Transfert

Univ-Money utilise son wallet de type EMONEY pour transférer de la monnaie électronique vers les comptes de ses utilisateurs. Cette opération est le service « dépôt » offert par le système d'Univ-Money. Pour cela, on effectue une demande depuis notre serveur vers le serveur d'API-money via l'opération ***POST /transfers***. Le serveur d'Univ-Money envoie la référence du transfert, le wallet qui envoie, le wallet qui reçoit, le wallet qui reçoit les éventuels frais de transfert, le montant du transfert et le montant des frais. Ces informations constituent le corps de la requête et sont envoyées de la forme suivante.

```
POST /transfers
{
    "partner_ref" : "TSF-u1594-20180310093048",
    "tag" : "Chuck Birthday gift",
    "sender_wallet_id" : "WE-4596525578456954",
    "receiver_wallet_id" : "WE-9876543219876543",
    "fees_wallet_id" : "WF-1234567891234567",
    "amount" : 210,
    "fees" : 3
}
```

*Figure 25 : corps de la requête d'une opération de transfert*

Le serveur d'API-money envoie comme réponse l'identifiant de la transaction qui vient d'être effectué. Cette réponse est de la forme suivante.

```
HTTP/1.1 201 CREATED
{
    "id" : "TX-8148254337416765"
}
```

### 4. Retrait

Cette fonctionnalité sera utilisée par Univ-Money pour retirer de l'argent depuis son compte de type EMONEY. Pour effectuer un retrait, Univ-Money doit dans un premier temps associer un

compte bancaire à son wallet, via l'opération ***POST /bankaccounts*** en renseignant l'IBAN de son compte bancaire. Une fois un compte bancaire associé au wallet, il est possible d'effectuer un « cash-out » ou « retrait » via l'opération ***POST /cash-out***. Le serveur d'Univ-Money envoie la référence du transfert, le wallet qui doit être débité, le wallet qui reçoit les éventuels frais de transfert, le montant du transfert, le montant des frais et l'identifiant du compte bancaire associé au wallet. Ces informations constituent le corps de la requête et sont envoyées de la forme suivante. Univ-Money dispose d'une documentation pour faire ces opérations.

# II. Les technologies et outils utilisés

## 1. Node.js

Node.js est un environnement d'exécution JavaScript, écrit par Ryan Dahl en 2009, open source, multiplateforme, back-end, qui exécute du code JavaScript en dehors d'un navigateur Web. Node.js permet aux développeurs d'utiliser JavaScript pour écrire des outils de ligne de commande et pour exécuter des scripts côtés serveur pour produire un contenu de page Web dynamique avant que la page ne soit envoyée au navigateur Web de l'utilisateur. Par conséquent, Node.js représente un paradigme «JavaScript partout», unifiant le développement d'applications Web autour d'un seul langage de programmation, plutôt que différents langages pour les scripts côté serveur et côté client.

Node.js a de nombreux avantages : système de paquet intégré (NPM), modèle non bloquant, performance du moteur V8, logiciel libre (licence MIT). Il dispose également d'une communauté très active. Son principal atout est la possibilité de coder en JavaScript, un langage de programmation déjà connu [9].

*Figure 26 : Logo Node.js*

## 2. SQL

Le **SQL** (Structured Query Language ou Langage de requête structurée en français) est un langage permettant de communiquer avec une base de données. Ce langage informatique est notamment très utilisé par les développeurs web pour communiquer avec les données d'un site web. Il permet de lire, d'insérer, de modifier et de supprimer des données dans une base de données relationnelles [10].

## 3. MySQL server

MySQL est un Système de Gestion de Base de Données (SGBD) parmi les plus populaires au monde. Il est distribué sous double licence, une licence publique générale GNU et une propriétaire selon l'utilisation qui en est faite. La première version de MySQL est apparue en 1995 et l'outil est régulièrement entretenu. Ce système est particulièrement connu des développeurs pour faire partie des célèbres quatuors : WAMP (Windows, Apache, MySQL et PHP), LAMP (Linux) et MAMP (Mac). Ces packages sont si populaires et simples à mettre en œuvre que MySQL est largement connu et exploité comme système de gestion de base de données pour des applications utilisant PHP. C'est d'ailleurs pour cette raison que la plupart des hébergeurs web proposent PHP et MySQL.

MySQL est un serveur de base de données relationnelles SQL qui fonctionne sur de nombreux systèmes d'exploitation (dont Linux, Mac OS X, Windows, Solaris, FreeBSD…) et qui est accessible en écriture par de nombreux langages de programmation, incluant notamment PHP, Java, Ruby, C, C++, .NET, Python, Javascript … [11]


*Figure 27 : Logo MySQL*

## 4. MySQL WorkBench

MySQL Workbench est un outil visuel unifié pour les architectes de bases de données, les développeurs et les administrateurs de bases de données. MySQL Workbench fournit des outils de modélisation de données, de développement SQL et d'administration complets pour la

configuration du serveur, l'administration des utilisateurs, la sauvegarde et bien plus encore. MySQL Workbench est disponible sur Windows, Linux et Mac OS X [12].

*Figure 28 : Logo MySQL WorkBench*

## 5. ArgoUML

ArgoUML est un logiciel libre de création de diagrammes UML. Programmé en Java, il est édité sous licence EPL 1.0. Il est multilingue, supporte la génération de code et l'ingénierie inverse. Il supporte sept types de diagrammes : cas d'utilisation, classes, séquence, état, collaboration, activité et déploiement [13].

*Figure 29 : Logo ArgoUML*

## 6. Visual studio code

Visual Studio Code est un éditeur de code source léger mais puissant qui s'exécute sur le bureau et est disponible pour Windows, macOS et Linux. Il est livré avec un support intégré pour JavaScript, TypeScript et Node.js et dispose d'un riche écosystème d'extensions pour d'autres langages (tels que C ++, C #, Java, Python, PHP) et des environnements d'exécution (tels que .NET et Unity) [14].

*Figure 30 : Logo visual studio code*

### 7. Postman

Postman est la plateforme de collaboration pour le développement d'API. Postman simplifie chaque étape de la création d'une API et rationalise la collaboration afin que nous puissions créer de meilleures API, plus rapidement. Il permet de faire des tests durant le développement des APIs [15].

*Figure 31 : Logo Postman*

L'utilisation de ces technologies et de ces outils nous a permis d'implémenter nos services. Cette implémentation est détaillée dans la partie suivante.

# III. Implémentation d'Univ-Money

## 1. Implémentation de la base de données

### a. Modèle relationnel

Il consiste, dans cette partie, à étudier sous quelle forme sont sauvegardées les instances des classes sur un support physique ou en d'autres termes un SGBD. L'utilisation d'un SGBD relationnel impose un changement dans la représentation des structures des classes. Nous allons concevoir le modèle relationnel en respectant les règles de passage du modèle objet vers le modèle relationnel.

- **Compte** (compteID, dateCreation, typeCompte, solde, account_link, wallet_link, #user_userID, #proprietaire_proprietaireID)
- **Propriétaire** (proprietaireID, nomComplet, date_naissance, tel, email, numero_ine, codeSecret, adresse, business_type, resgitration_number, #representant_representatntID)
- **Représentant** (representatntID, nom, prenom, date_naissance)
- **Opération** (operationID, dateOperation, typeOperation, montant, #user_userID, #compte_compteID)
- **User** (userID, username)

### b. Etablissement de la connexion à la base de données

La connexion à la base de données nécessite le démarrage de MySQL. Ensuite la configuration est faite dans le fichier db.js de notre système. La fonction createConnection() appliquer à l'objet mysql du module mysql permet d'établir la connexion avec les paramètres suivants :

- Domaine : localhost (hôte local)
- Utilisateur : root
- Mot de passe : password
- Base de données : wallet_database
- Port : 3306

App > exports > JS db.js > ...

```
1   var mysql = require('mysql')
2
3   connection = ''
4   var connection_database = () => {
5       connection = mysql.createConnection({
6           host: 'localhost',
7           user: 'root',
8           password: 'password',
9           database: 'wallet_database',
10          port: 3306
11      })
12      connection.connect((err) => {
13          if (err) {
14              console.error('error connecting : ' + err.stack);
15          }
16          console.log('Connecting as id ' + connection.threadId);
17      })
18  }
19  exports.connection_database = connection_database
```

*Figure 32 : Etablissement de la connexion à la base de données*

## 2. Implémentation des services d'Univ-Money

L'implémentation consiste à coder la logique métier des services de notre système avec Node.js. Pour ce faire, nous adoptons l'architecture REST afin que nous pussions avoir des services web RESTful. La structure du système générée à partir de Swagger Editor est présentée ci-dessous.

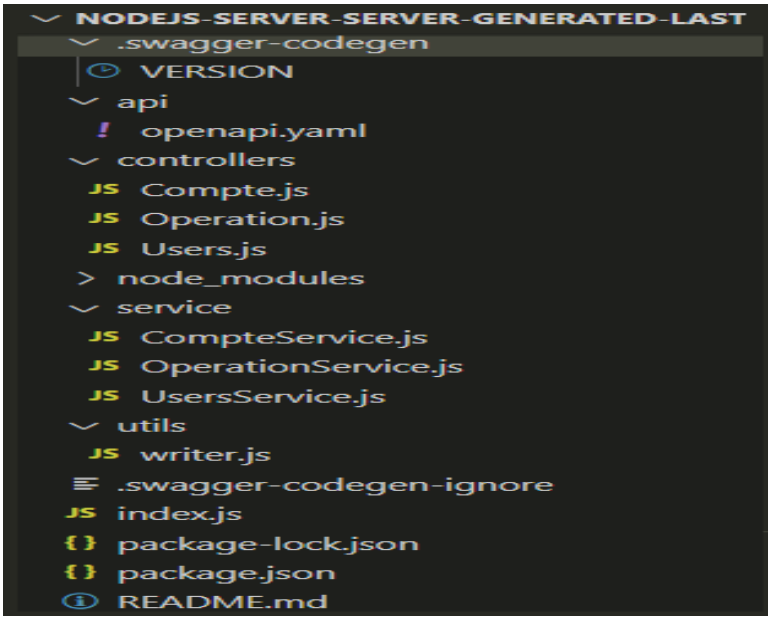


*Figure 33: structure du système générée avec Swagger Editor*

Ainsi, dans cette partie nous allons présenter quelques captures d'écran relatives au service de la création de compte standard.

## a. Définition de la création de compte standard

La figure suivante représente la déclaration de la fonction qui traite la sauvegarde des données au niveau de notre base de données pour garder l'historique.

```
7   exports.saveCompteStandard = (nomComplet, date_naissance, categorie, tel, email, numero_ine, codeSecret, adresse,
8       typeCompte, account_link, wallet_link, createur) => {
9       return new Promise(function(resolve, reject) {
10          saveUserStandard(nomComplet, date_naissance, categorie, tel, email, numero_ine,
11              codeSecret, adresse).then((proprietaire) => {
12  >           getUserID(createur).then((response) => {···
24              }).catch((err) => {
25                  reject(err)
26              })
27
28          }).catch((err) => {
29              reject(err)
30          })
31      })
32  }
```

*Figure 34 : Définition de la création de compte*

## b. Communication avec l'API-money

La communication avec le système d'API-money est représentée dans la figure suivante. Cette communication permet au système d'Univ-Money d'avoir une autorisation pour créer le compte.

```
80      let body = {
81  >       "subscriber": {···
91          },
92  >       "address": {···
97          },
98          "email": email,
99          "tag": "account_type2"
100     }
101
102     let hmac = crypto.createHmac('sha256', api_secret_key);
103     hmac.update(api_access_key + ':' + TimeStamp + ':' + version + ':' + JSON.stringify(body));
104     let hashAccount = hmac.digest('hex');
105
106     let options = {
107         method: 'post',
108         url: url + 'accounts/standard',
109         json: true,
110         headers: {
111             'Authorization': 'AUTH ' + api_access_key + ':' + TimeStamp + ':1:' + hashAccount,
112             'Content-Type': 'application/json'
113         },
114         body: body
115     };
116 >   request(options, (err, responseAccount, body) => {···
191     });
```

*Figure 35 : Communication avec le système API-money*

### c. Procédure d'accès à la ressource

Le fichier **index.js** assure la configuration des routes et la création d'un serveur de la manière suivante**.**

```
JS index.js > ...
1   'use strict';
2
3   var path = require('path');
4   var http = require('http');
5
6   var oas3Tools = require('oas3-tools');
7   var serverPort = 5000;
8
9   // swaggerRouter configuration
10  var options = {
11      routing: {
12          controllers: path.join(__dirname, './controllers')
13      },
14  };
15
16  var expressAppConfig = oas3Tools.expressAppConfig(path.join(__dirname, 'api/openapi.yaml'), options);
17  var app = expressAppConfig.getApp();
18
19  // Initialize the Swagger middleware
20  http.createServer(app).listen(serverPort, function () {
21      console.log('Your server is listening on port %d (http://localhost:%d)', serverPort, serverPort);
22      console.log('Swagger-ui is available on http://localhost:%d/docs', serverPort);
23  });
```

*Figure 36 : Configuration de notre fichier principal index.js*

Dans le fichier **controllers/compte.js** on définit la fonction pour accéder à la ressource de la manière suivante. C'est cette fonction qui permet de renvoyer la réponse à la requête de l'utilisateur.

```
26  module.exports.addAccountStandard = function addAccountStandard (req, res, next, body) {
27    Compte.addAccountStandard(body)
28      .then(function (response) {
29        utils.writeJson(res, response);
30      })
31      .catch(function (response) {
32        utils.writeJson(res, response);
33      });
34  };
```

*Figure 37: focntion d'accès à la ressource*

Dans le fichier **service/compteService.js** on a la fonction **addAccountStandard()**, représentée à la figure ci-après, qui contient le code métier du service création de compte étudiant.

```
50  /**
51   * Créer un nouveau compte Standard
52   *
53   * body CompteStandard les paramètres demandés pour la création de compte standard
54   * returns ApiResponse
55   **/
56  exports.addAccountStandard = function(body) {
57    return new Promise(function(resolve, reject) {···
69    });
70  }
```

*Figure 38 : code métier du service création de compte étudiant*

Ce chapitre nous a permis de parler d'API-money et de ses services et comment nous les avons utilisés. De plus, on a décrit l'implémentation des services d'Univ-Money en prenant le cas du service « créer compte étudiant ». Maintenant qu'on ait implémenté nos services, nous parlerons dans le prochain chapitre de l'utilisation de nos services et la présentation d'un prototype.

# Chapitre VI : Utilisation des services et réalisation de prototypes

Univ-Money a mis en place un système composé de services web pour effectuer la distribution et l'encaissement de la monnaie électronique dans des plateformes informatiques différentes. Dans cette partie nous allons montrer l'utilisation des services du système Univ-Money dans ces plateformes. Pour ce faire, nous présenterons l'utilisation des services d'Univ-Money en montrant les interfaces des services du contrat obtenu avec Swagger et ensuite nous présenterons des prototypes qui ont été réalisés.

## I. Utilisation des services d'Univ-Money

Les services d'Univ-Money sont des services web RESTful qui sont chacun identifiés par un URI et accessibles par la syntaxe et la sémantique du protocole HTTP. Pour accéder à un service il faudra lancer une requête du serveur demandeur vers le serveur d'Univ-Money. Chaque requête contient un en-tête autorisation qui permet l'authentification. Ainsi, pour utiliser les services d'Univ-Money, l'utilisateur doit disposer d'une clé api pour s'authentifier. Dans cette partie nous parlerons d'abord de l'authentification et ensuite de l'utilisation des services les plus importants du système Univ-Money notamment la création de compte pour les étudiants, la création de compte pour les structures de l'Université, le paiement des frais d'inscription, le paiement du logement, le paiement des tickets de restauration, le paiement des frais médicaux et le paiement des bourses pédagogiques.

### 1. L'authentification

Toutes les requêtes nécessitent d'être entièrement authentifiées via l'en-tête HTTP « Authorization ». Les demandes doivent être envoyées depuis les serveurs des clients qui utilisent le système Univ-Money vers le serveur d'Univ-Money. Le contenu de l'en-tête doit respecter le format décrit ci-après : AUTHORIZATION : API_KEY.

## 2. Le service création de compte étudiant

### a. L'URI du service

Le service de la création de compte pour les étudiants est accessible via l'opération représentée par la Figure 39 suivante.

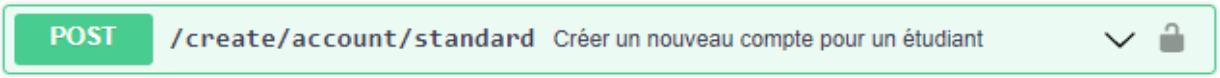


*Figure 39 : URI « créé compte étudiant »*

### b. Elaboration du corps de la requête

Cette section permet d'expliciter les paramètres que le système Univ-Money doit recevoir pour la création de compte STANDARD ainsi que le format qu'ils sont envoyés.


*Figure 40 : paramètres requis pour la création de compte STANDARD*

## 3. Le service création de compte partenaire

### a. L'URI du service

Le service de la création de compte est accessible via l'opération représentée par la Figure 41 suivante.

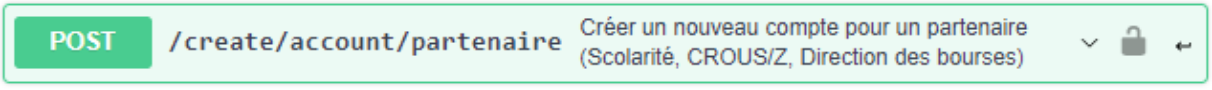


*Figure 41 : URI « créé compte partenaire »*

### b. Elaboration du corps de la requête

Cette section permet d'expliciter les paramètres que le système Univ-Money doit recevoir pour la création d'un compte pour un partenaire (Scolarité, CROUS/Z, Direction des bourses,…) ainsi que le format qui doivent être envoyés.

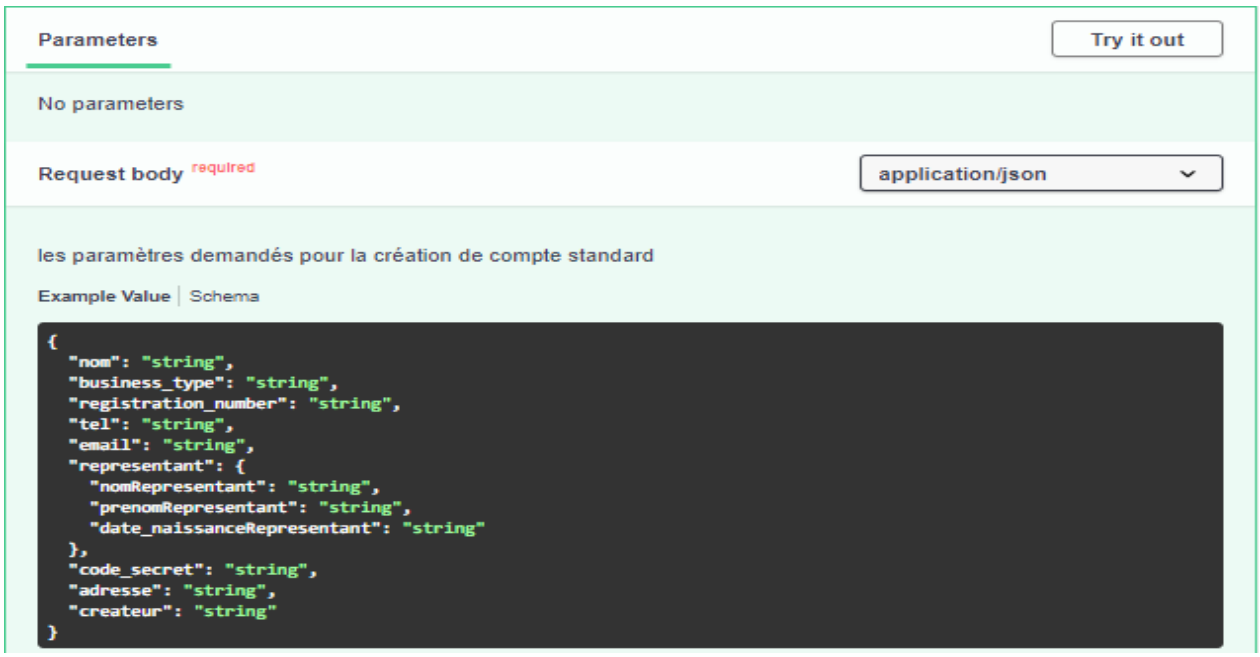


*Figure 42 : paramètres requis pour la création de compte partenaire*

## 4. Le service création de compte distributeur

### a. L'URI du service

Le service de la création de compte est accessible via l'opération représentée par la Figure 43 suivante.


*Figure 43 : URI « créé compte distributeur »*

### b. Elaboration du corps de la requête

Cette section permet d'expliciter les paramètres que le système Univ-Money doit recevoir pour la création d'un compte pour distributeur ainsi que le format qui doivent être envoyés.

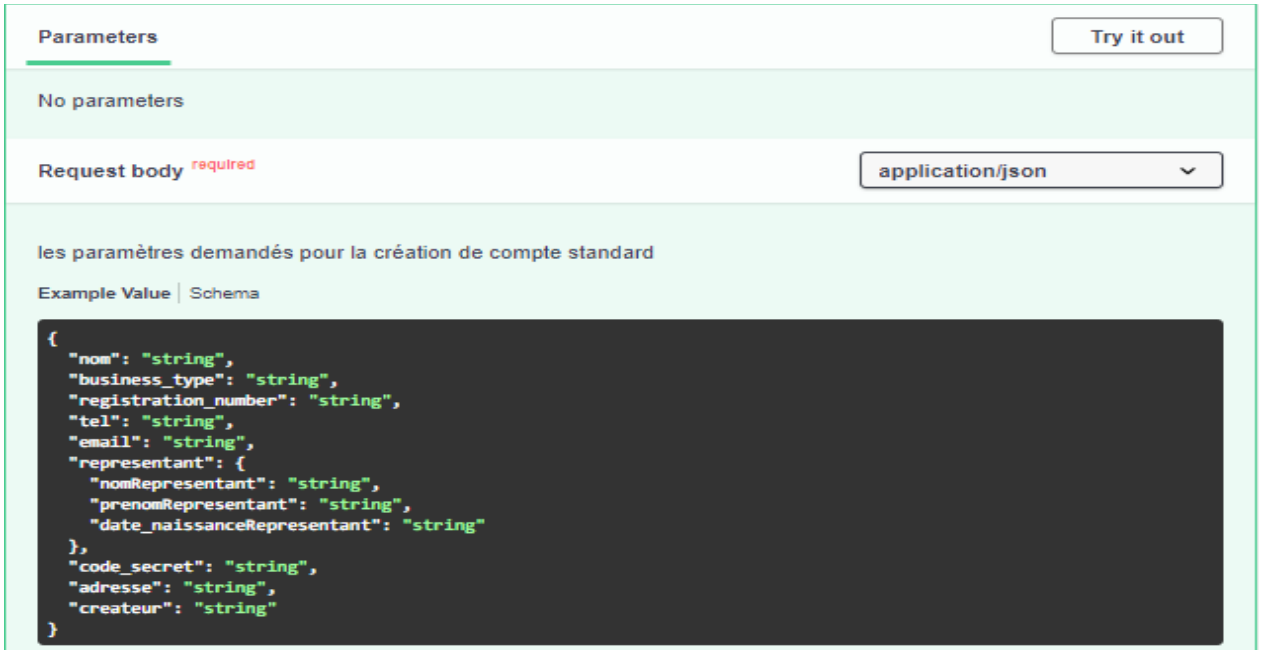


*Figure 44 : paramètres requis pour la création de compte distributeur*

## 5. Le service du transfert d'argent

### a. L'URI du service

Le service du transfert d'argent est accessible via l'opération représentée par la Figure 45 suivante.


*Figure 45 : URI « transfert d'argent »*

### b. Elaboration du corps de la requête

Cette section permet d'expliciter les paramètres que le système Univ-Money doit recevoir pour le paiement des frais d'inscription notamment le compte de l'étudiant, le compte de la scolarité et le montant à payer ainsi que le format que les paramètres ont envoyés.

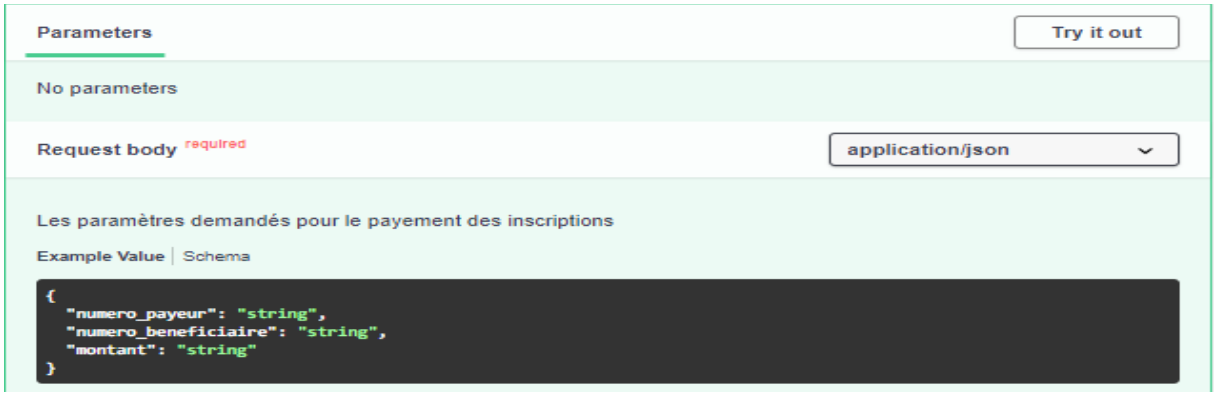


*Figure 46 : paramètres requis pour le paiement des frais d'inscription*

# II. Réalisation et présentation de prototypes

Afin de mettre en œuvre les fonctionnalités décrites dans le chapitre II précédent, des prototypes de service Web ont été développés. Ces derniers implémentent le service de création de compte pour les étudiants, le service de transfert, et le service de la consultation de solde dans des systèmes différents et conformément au contrat de service précédemment décrit (cf. chapitre II), en s'appuyant sur des librairies et composantes logicielles existantes. Dans cette partie, nous identifierons d'abord les fonctionnalités des systèmes mises en place pour ensuite parler du choix technologique pour la réalisation des prototypes en montrant les interfaces qui ont été conçues.

## 1. Le choix technologique

Les prototypes sont réalisés en utilisant des technologies côté client, des technologies côté serveur et des outils pour la gestion des différentes bases de données.

### a. Les technologies côté client

Pour construire ces prototypes, nous avons recours à des langages qui nous ont servi d'établir les plans dont nous avons besoin pour réaliser les prototypes. Ces langages sont les suivants.

- **HTML (*HyperText Markup Language*)** : c'est un langage à balises de données hypertextes. Il est basé sur l'hypertexte qui permet de lier plusieurs documents entre eux et les balises qui définissent la sémantique de l'élément décrit.
- **CSS (*Cascading Style Sheets*)** : il permet la mise en forme des documents HTML notamment le rendu visuel et des animations

- **JavaScript** : c'est le langage de programmation côté client utilisé pour les interactivités des pages, pour interagir avec les éléments de la page, pour interagir avec le système et pour la logique applicative de l'application.
- **EJS** : c'est un langage de modèles simple qui nous a permis de générer un balisage HTML avec du JavaScript simple.

### b. Les technologies côté serveur

Le langage côté serveur que nous avons utilisé pour la récupération et le stockage des informations dans les bases de données est **SQL (Structured Query Language)** très utilisé par les développeurs web pour communiquer avec les données d'un site web.

### c. Le runtime Node.js

Node.js est un environnement d'exécution open source et multiplateforme permettant d'exécuter du code JavaScript en dehors d'un navigateur. Nous l'avons utilisé pour créer des services back-end tels que des APIs qui sont les fonctionnalités des prototypes.

### d. Le framework Express.js

Express JS est un framework pour construire des applications web basées sur Node.js. C'est le framework standard pour le développement de serveur en Node.js. Il facilite l'organisation des fonctionnalités de notre application avec le middleware et le routage. De plus, il ajoute des utilitaires utiles aux objets HTTP de Node.js et facilite le rendu des objets HTTP dynamiques.

### e. Le système de gestion des bases de données

Un Système de Gestion de Base de Données (SGBD) est un logiciel qui permet de stocker des informations dans une base de données, de les rechercher, de les supprimer et de les modifier. Dans la réalisation des prototypes nous avons choisi de travailler avec MySQL qui est un SGBD parmi les plus populaires au monde. De plus, il fonctionne sur de nombreux systèmes d'exploitation et est accessible en écriture par JavaScript.

## 2. Identification des prototypes

Les prototypes que nous avons réalisés font intervenir trois systèmes parmi les systèmes que nous avons définis au chapitre II à savoir le client gestion scolarité, le client gestion des bourses et le client web pour les étudiants. Cette réalisation regroupe plusieurs fonctionnalités reparties

entre ces différents systèmes que nous allons décrire dans les différentes sous-sections suivantes.

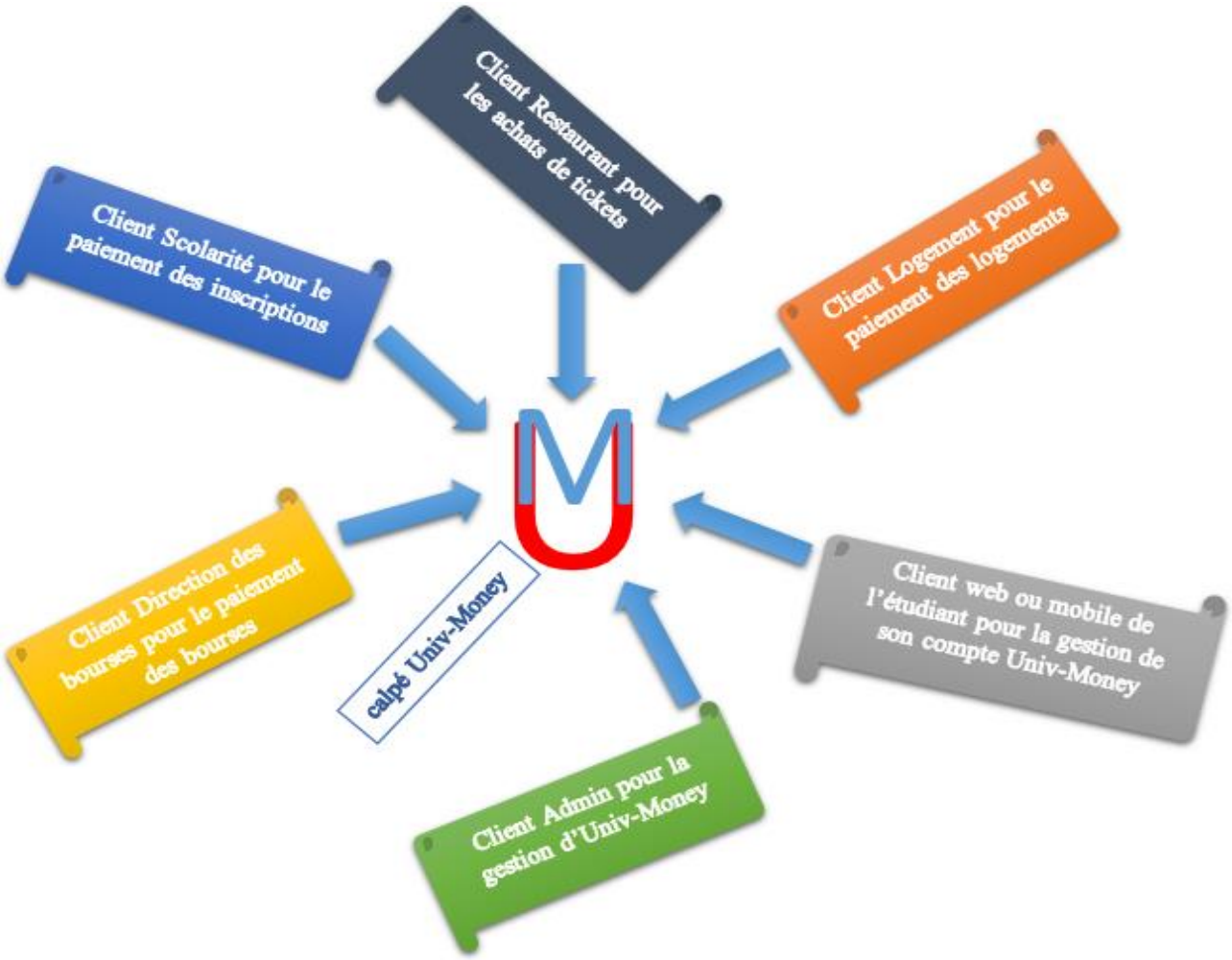


*Figure 47: Architecture générale d'utilisation d'Univ-Money par les clients partenaires*

### a. Client scolarité

Le système scolarité est une application web, mise à la disposition de la scolarité, qui permet d'associer un compte Univ-Money à chaque étudiant inscrit. De ce fait, le client gestion scolarité fait appel au service « ***créer compte étudiant*** » du système Univ-Money pour effectuer cette association.

Une fois qu'un étudiant s'est inscrit à l'Université, un compte Univ-Money doit être créé à son profit pour qu'il puisse bénéficier des différents services destinés aux étudiants. Pour la création des comptes nous avons listé les étudiants qui n'ont pas encore de compte Univ-Money ensuite

nous associons un compte à chaque étudiant via un bouton dont on dispose dans l'interface. Cette étape est représentée par la figure suivante.

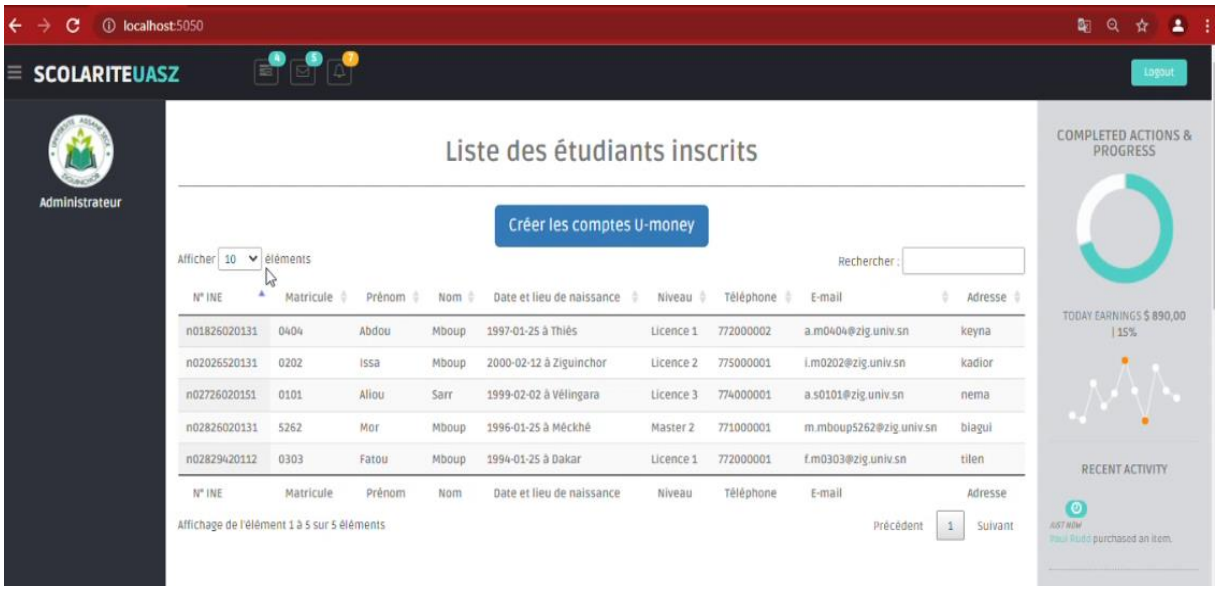


*Figure 48 : liste des étudiants inscrits*

La figure ci-après est le résultat de la création des comptes dans le tableau de bord d'API-money.

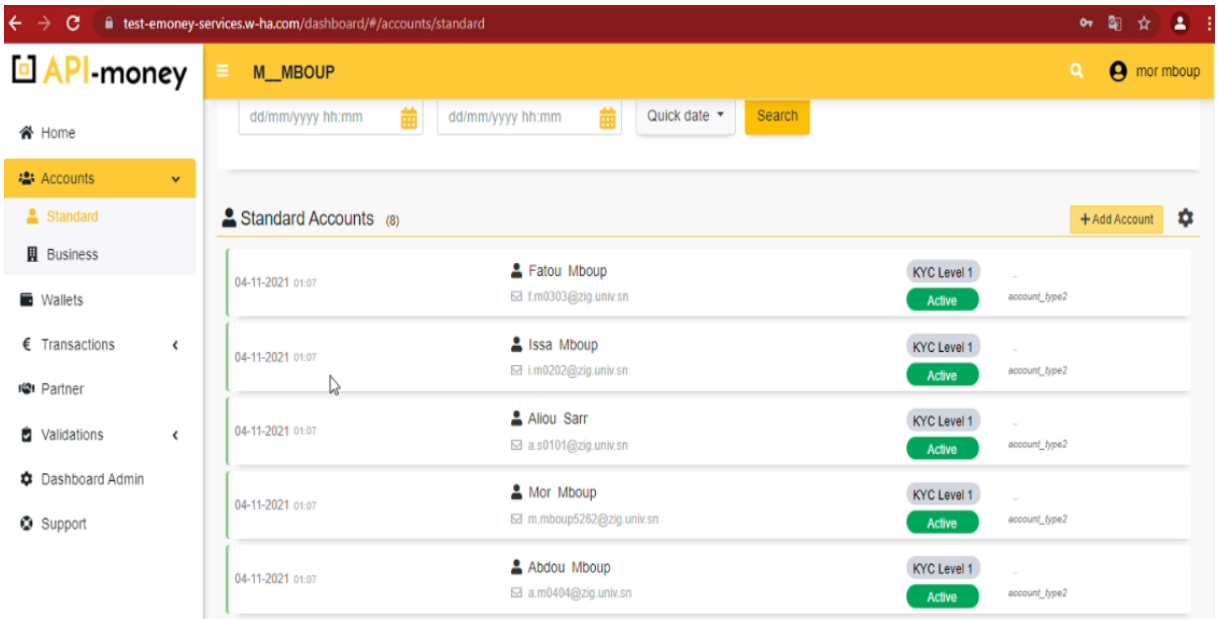


*Figure 49 : tableau de bord d'API-money après création des comptes*

Cette fonctionnalité nous permet de passer à l'étape de la sous-section suivante.

### b. Client direction des bourses

Le système direction des bourses permet à la Direction des bourses de faire les paiements de bourses directement dans les comptes Univ-Money des étudiants. Pour ce, nous avons mis en place un système qui a la fonctionnalité « **payer des bourses** » qui fait appel au service « **transférer de l'argent** » du système Univ-Money pour faire les virements. En effet, la Direction des bourses détient en avance un compte Univ-Money qui lui permet de faire le transfert vers les comptes Univ-Money des étudiants.

Après que les comptes ont été créés, les premiers virements peuvent maintenant s'effectuer du compte de la Direction de bourses que nous avons créé manuellement vers les comptes étudiants. Pour ce, l'état de paiement doit être établi avant d'effectuer le paiement. Dans le système de la gestion des bourses, les étudiants disposent d'un portail qui leur permet de déposer leurs dossiers. Dans ce portail l'étudiant s'authentifie avant d'y accéder comme le montre la figure ci-dessous.

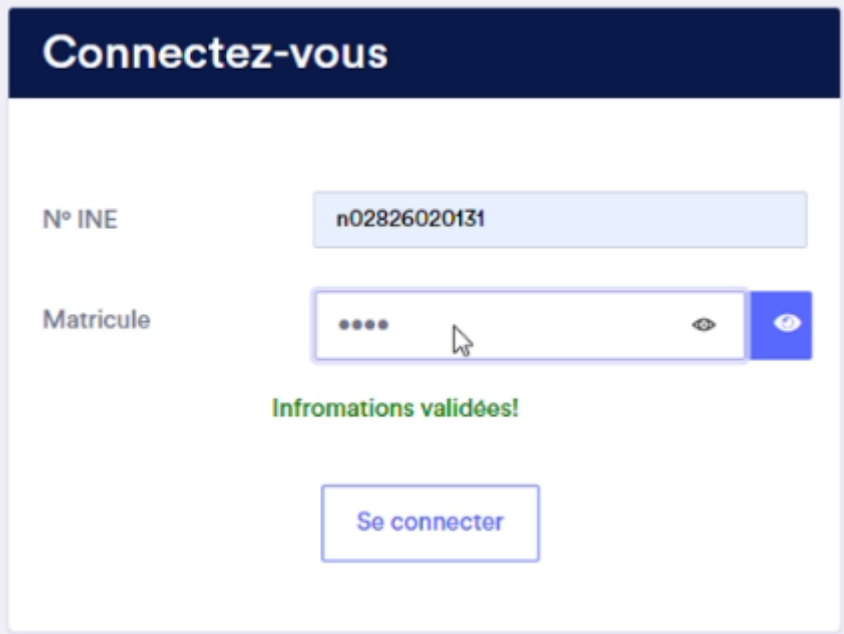


*Figure 50 : interface d'authentification des étudiants*

Ensuite le menu « ***Mon dossier*** » permet à chaque étudiant de déposer son dossier et de le visualiser comme le représentent les figures suivantes.

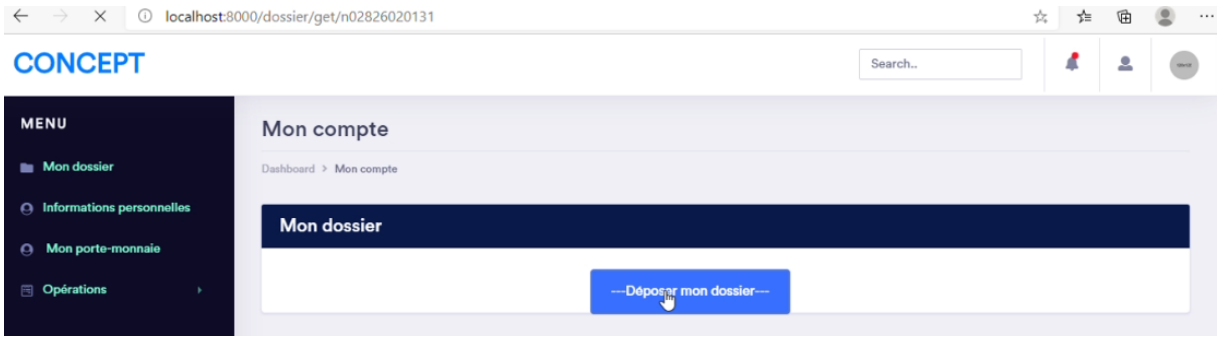


*Figure 51 : interface de dépôt des dossiers*

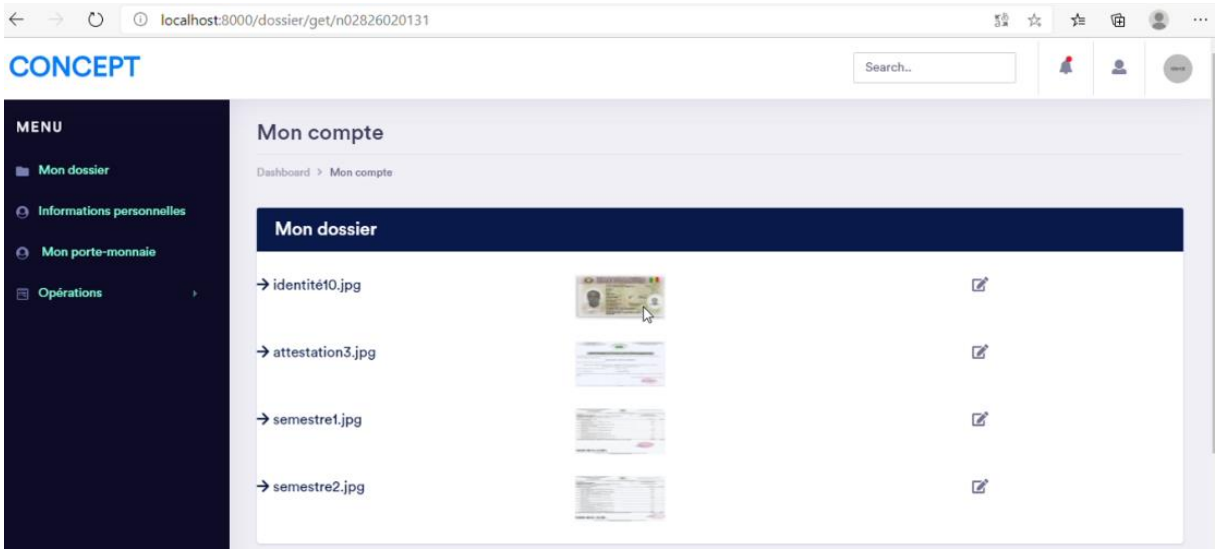


*Figure 52 : interface de visualisation du dossier d'un étudiant*

Maintenant le gestionnaire des bourses peut s'authentifier pour accéder à son espace afin de procéder à la vérification des dossiers qui peuvent être validés ou refusés comme le montrent les figures ci-dessous.

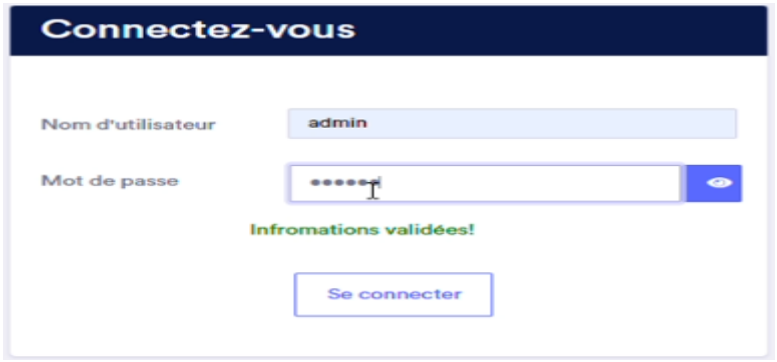


*Figure 53 : interface d'authentification du gestionnaire des bourses*

Une fois dans son portail, le gestionnaire des bourses peut voir la liste des étudiants qui ont déposé leurs dossiers.


*Figure 54 : liste des dossiers des étudiants*

Ainsi, il peut passer à la visulisation et à la validation des dossiers des étudiants.

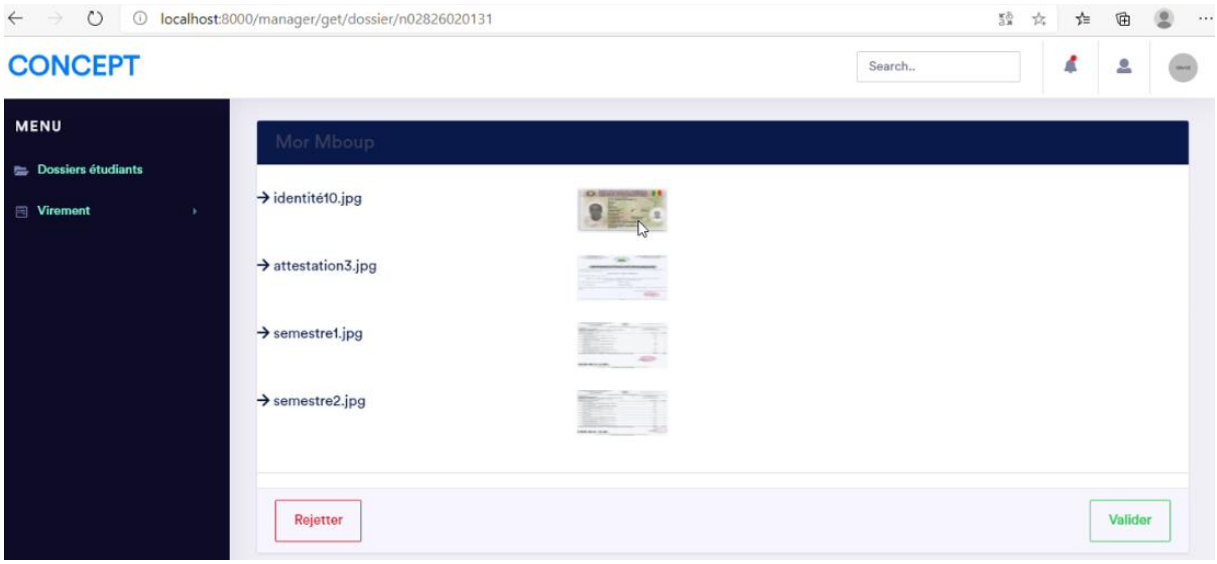


*Figure 55 : interface de visualisation et de validation des dossiers des étudiants*

Après chaque validation, l'étudiant est ajouté à la liste des étudiants qui percevront leurs bourses à la fin du mois. Le sous-menu « ***Virement > mensuel*** » permet d'effectuer le virement mensuel en un clic sur tous les comptes des étudiants qui sont dans la liste représentée de la manière suivante.

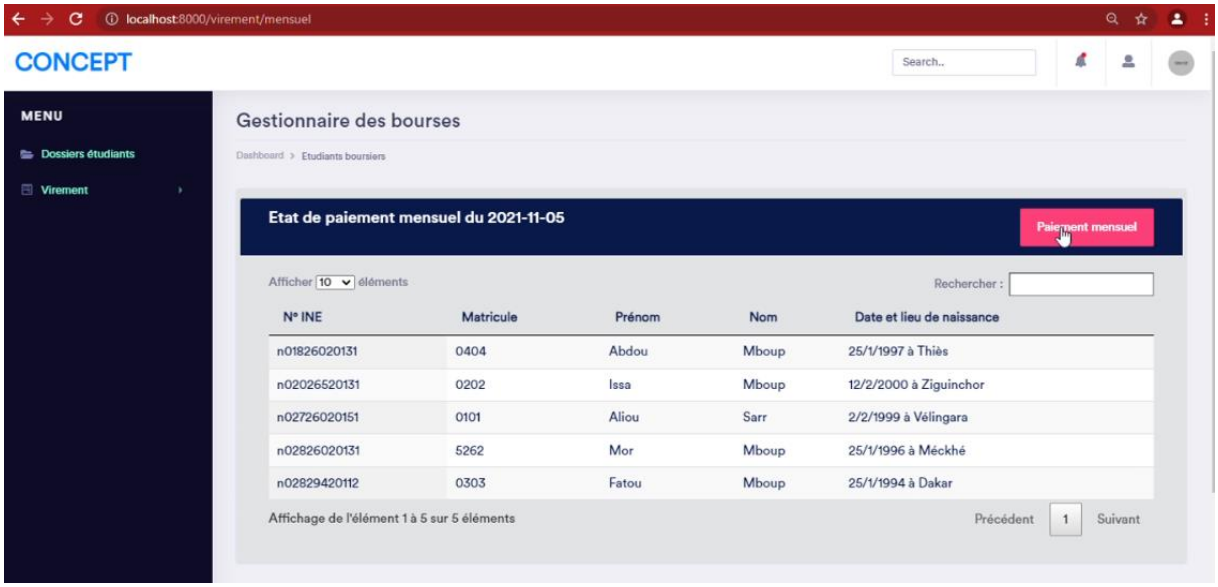


*Figure 56 : interface de paiement mensuel des bourses*

Le tableau de bord de l'API-money avant et après le virement est représenté par les figures suivantes.

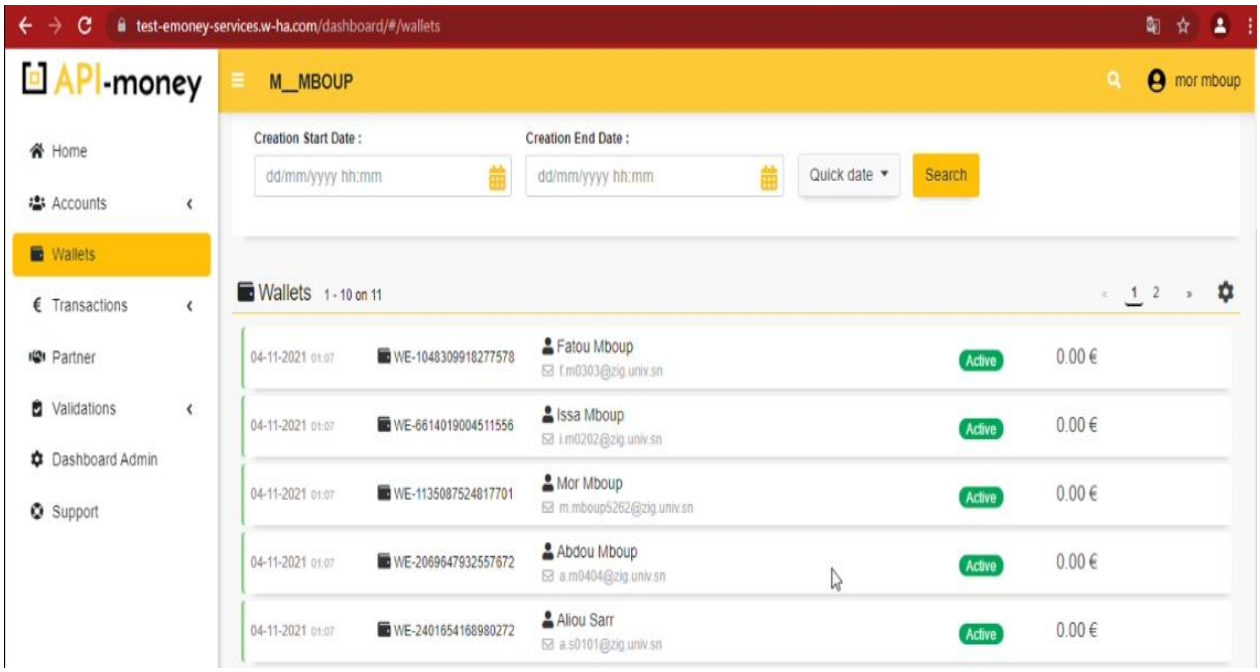


*Figure 57 : interface avant virement*

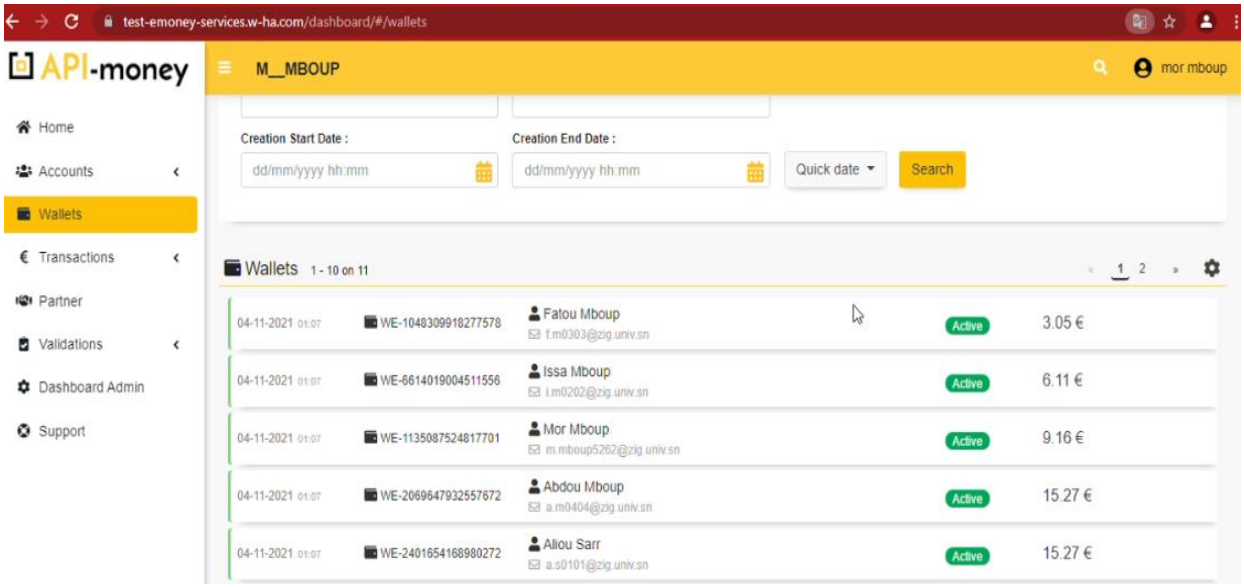


*Figure 58 : interface après virement*

Après avoir effectué les premiers virements on peut ainsi passer aux fonctionnalités développées dans le client web pour les étudiants.

### c. Le système étudiant

Le système étudiant est une application web qui permet à l'étudiant de « ***payer son inscription*** », de « ***payer son logement*** » et de « ***payer des tickets restaurant*** ». Toutes ces fonctionnalités font appel au service « ***transférer de l'argent*** » du système Univ-Money pour faire chaque paiement de service. Il permet aussi à l'étudiant de voir son solde conformément au service « ***consulter solde*** » du système Univ-Money.

Donc les fonctionnalités du prototype réalisé sont réparties dans le Tableau 15 suivant selon les acteurs des différents systèmes qui composent le prototype et les services d'Univ-Money.

*Tableau 15 : les fonctionnalités du prototype selon les acteurs*

| **Fonctionnalités** | **Acteurs** | **Services Univ-Money** |
|---|---|---|
| **Voir son solde** | Etudiant | Consulter solde |
| **Payer son inscription** | Etudiant | Transférer de l'argent |
| **Payer son logement** | Etudiant | Transférer de l'argent |

| | | |
|---|---|---|
| **Payer des tickets restaurant** | Etudiant | Transférer de l'argent |

L'étudiant qui a son compte alimenté par la réception de sa bourse ou par une opération de dépôt auprès d'un point de distribution peut désormais payer ses frais d'inscription avec son compte Univ-Money gratuitement et sans suivre un rang. Il s'authentifie d'abord pour accéder à la plateforme de la manière suivante.


*Figure 59 : interface d'authentification*

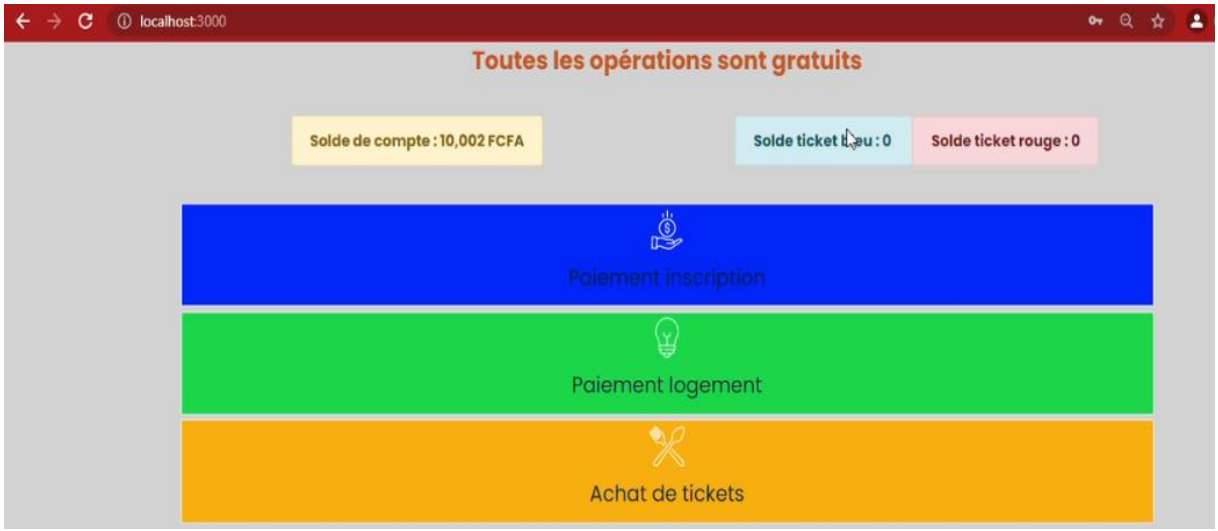


*Figure 60 : interface d'accueil du client web étudiant*

La section paiement inscription permet à l'étudiant de confirmer le paiement en entrant son code secret et en un seul clic.


*Figure 61 : interface du paiement des frais d'inscription*

L'étudiant peut aussi payer la mensualité de son logement avec son compte Univ-Money gratuitement. Ce paiement se fait en un seul clic après avoir saisi son code secret. La figure suivante représente l'interface de paiement du logement.


*Figure 62 : interface du paiement des frais de logement*

Ainsi de même, l'étudiant peut acheter des tickets avec son compte Univ-Money gratuitement via l'interface de la figure ci-dessous. Il choisit le nombre de tickets bleus et le nombre de tickets rouges qu'il veut et qui est possible par rapport à la somme disponible dans son compte. Puis, il confirme le paiement en saisissant son code secret.


*Figure 63 : interface d'achat de ticket*

Ce chapitre nous a permis de montrer le résultat obtenu. En parlant des moyens utilisés pour atteindre notre objectif, en plus, nous avons fait quelques captures d'écran des tests que nous avons effectués.

# Conclusion et perspectives

Le mémoire de fin d'étude s'est déroulé en deux phases principales. La première, celle de l'étude conceptuelle et technique. Ce qui nous a permis de proposer pour l'UASZ, une conception d'une solution informatique réalisable pour la mise en œuvre d'un système de gestion de transactions estudiantines au sein de l'Université.

Ceci afin de résoudre les problèmes de lenteur appréhendés dans le fonctionnement actuel. En effet, ce système permet, au niveau de la Direction des bourses, de faire le virement des bourses avant le 05 de chaque mois vu le caractère numérique de la bourse dans ce système mis en place. Une fois son compte Univ-Money alimenté par sa bourse ou auprès d'un point de distribution, l'étudiant peut désormais faire des transactions de paiement des frais d'inscription, de paiement des tickets pour la restauration, de paiement de la mensualité du logement social,….

La seconde phase a consisté en la réalisation du prototype par l'implémentation de certains services du système Univ-Money en vue de produire un logiciel adapté et spécifique à des tâches bien précises.

En développant ce sujet, nous étions invité à réaliser plusieurs objectifs entre autres, la conception du système Univ-Money en vue de mettre à la disposition de l'Université une solution pour l'amélioration de sa politique de gestion des services rendus aux étudiants.

Ainsi, à travers ce système nous pourrons gérer les services rendus aux étudiants grâce à la mise en place des systèmes clients pour la gestion de la scolarité, des logements, du restaurant, des bourses….

D'une part les étudiants peuvent payer leurs frais d'inscription, leurs frais de logement ainsi que leurs frais médicaux via les applications web et mobile mises à leur disposition sans avoir à suivre des files d'attentes et sans frais en guise de commissions.

D'autre part, la Direction des bourses peut, au début de chaque mois, être dans les délais de paiement même s'il n y a pas de l'argent liquide disponible vu le caractère numérique de la bourse dans le système Univ-Money. De ce fait, il y'aura une diminution remarquable des perturbations que nous vivons dans le milieu universitaire. De plus, toute la somme l'argent dépensée par la Direction des bourses pour le paiement des intérêts du partenariat avec

l'Ecobank peut devenir une source de financement de notre projet ou d'augmentation du nombre d'étudiants qui reçoivent la bourse sociale.

L'intérêt de ce système est modélisé par la Figure 64 suivante.

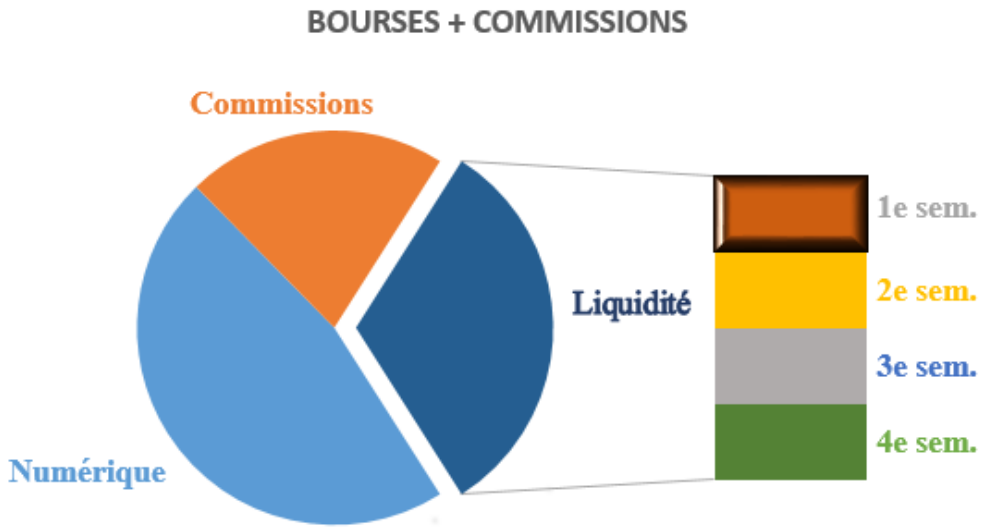


*Figure 64 : représentation graphique des apports d'Univ-Money*

Ce diagramme permet de visualiser l'apport du système Univ-Money au profit de la Direction des bourses. Il est composé de trois secteurs décrits de la manière suivante.

- Le secteur numérique représente la partie des bourses qui sera numérique tout le temps grâce au système Univ-Money vu son caractère numérique.
- Le secteur commissions est la partie qui permet de mettre en évidence l'argent qui est dépensé pour payer des commissions ou intérêts par la Direction de bourses au profit de l'Ecobank et ses partenaires. Le système Univ-money engendre le retour de ces commissions en faveur de la Direction des bourses.
- Le secteur liquidité permet d'expliciter la façon dont on peut procéder pour la distribution de l'argent en espèces. Un des avantages d'Univ-money dans ce secteur est qu'il permet un gain de temps très important en faveur du Ministère de l'Enseignement, de la Recherche et de l'Innovation concernant les contraintes d'indisponibilité de l'argent liquide. L'étudiant n'a plus besoin de récupérer de l'argent pour payer son inscription, son logement ou acheter des tickets pour la restauration car toutes les applications de gestion de ces services implémentent des fonctionnalités liées à son calpé pour lui permettre de faire directement les paiements via son calpé Univ-Money. Ainsi l'argent liquide va plus au niveau des distributeurs et des partenaires qui en auront

besoin pour faire leurs dépenses ou bien faire des retraits d'argent pour les étudiants qui veulent satisfaire d'autres besoins comme acheter des habiles au marché de Boucotte. De ce fait, l'approvisionnement des distributeurs et des partenaires peut se faire de manière séquentielle c'est-à-dire, pendant chaque semaine du mois on leur verse une somme d'argent nécessaire à la satisfaction de ces besoins.

La réalisation effective de ce système de paiement permettra sans doute à l'Université entre autres de disposer d'un système de paiement personnalisé et centralisé à l'Université pour faciliter la gestion des services rendus aux étudiants et d'ouvrir ses portes à de nouvelles perspectives dans les domaines de l'innovation et de la technologie.

Dans la suite de ce mémoire, nous notons comme perspectives le développement des différents systèmes qu'utilise Univ-Money dans les services aux étudiants au sein de l'Université notamment :

- Les clients web et mobile pour les étudiants
- Les clients partenaires comme ceux de la gestion de la scolarité, de la gestion du restaurant, de la gestion des logements, de la gestion des bourses,…
- Le client distributeur pour les points de distribution
- Le client administrateur pour la gestion d'Univ-Money
- …

Dans l'avenir, nous développerons notre propre API-money afin d'être autonome et pour plus de sécurité.

Il est aussi prévu d'élargir le travail à l'endroit des PER, PATS et dans les autres Universités du Sénégal sous l'appui du Ministère de l'Enseignement supérieur, de la Recherche et de l'Innovation.

Notons que le thème « Univ-Money, un porte-monnaie électronique pour la gestion de transactions étudiantes au sein de l'Université » a été d'un apport considérable pour nous, car il nous a permis de mettre réellement en pratique les connaissances acquises théoriquement durant notre formation et de les approfondir à travers de nombreuses recherches qui ont été effectuées dans le cadre de ce mémoire malgré les difficultés liées à des manquements durant le déroulement du projet.

# Webographie

**[1] :** site de la Direction des bourses https://www.directiondesbourses.sn/ consulté 16/08/2021

**[2] :** site de senenews https://www.senenews.com/etudiant/a-la-une/3897_3897_0105.html consulté le 24/08/2021

**[3] :** Portail FinDev https://www.findevgateway.org/fr/actualites/comment-les-banques-sont-elles-impliquees-dans-lemission-du-mobile-money-au-senegal consulté le 26/11/2021

**[4] :** site wikipedia https://fr.wikipedia.org/wiki/Porte-monnaie_électronique consulté le 26/11/2021

**[5]:** site de W-HA https://www.w-ha.com/fr/2017/10/05/lancement-de-api-money/ consulté 25/01/2020

**[6]** : "UML 2 en action'' de Pascal Roques 4e édition EYROLLES disponible sur https://doc.lagout.org/programmation/Databases/SQL/UML.pdf consulté le 26/09/2021

**[7]** : site de wikipedia https://fr.wikipedia.org/wiki/Representational_state_transfer consulté le 25/05/2020

**[8]** : site d'API-money https://www.API-money.com/docs/ consulté 30/01/2020

**[9] :** site de wikipedia https://en.wikipedia.org/wiki/Node.js consulté le 03/03/2020

**[10] :** site de sql https://sql.sh/ consulté le 19/03/2021

**[11] :** site de sql https://sql.sh/sgbd/mysql consulté le 20/03/2021

**[12] :** site mysql https://www.mysql.com/fr/products/workbench/ consulté le 18/04/2021

**[13] :** site de wikipedia https://fr.wikipedia.org/wiki/ArgoUML consulté le 14/02/2021

**[14] :** site de visualstudio https://code.visualstudio.com/docs consulté le 13/6/2020

**[15] :** site de postman https://www.postman.com/company/about-postman/ consulté le 12/01/2020