**UNIVERSITE PROTESTANTE AU CONGO**

****

**FACULTE DES SCIENCES INFORMATIQUES**

Département de Génie Informatique

BP. 4745 KINSHASA II

Kinshasa-lingwala

**Développement d’une API de paiement en attente**

**Cas d’un site d’e-commerce (Marianustyle)**

**Par**

** ANDEDI ANDEDI JOEL**

Travail de fin d’études présenté en vue de l’obtention du Grade de gradué en Sciences informatiques

**Directeur**: Chef des travaux MATE LANDRY

**Rapporteur** : Assistant SAKASAKA JOSMARD

**AOUT 2023**

EPIGRAPHE

***“***

*La confiance est la colle qui tient tout ensemble. C'est l'élément essentiel de toute relation, personnelle ou professionnelle*

**”**.

**Stephen R. Covey, 1989**

# DEDICACES

Nous dédions ce travail à :

* Nos très chers parents Jean ANDEDI et Mariam ANUNGA, pour leur amour indéfectible, leur soutien moral, mental et financier.
* Nos frères et sœurs, Johnny YETSHA, Chris AYOKA et Rachel OSAKO, pour leur présence et leur soutien.
* Nos cousins et cousines, Alice Asha, Jean AYOKA, Micheline MUKANGA, Junior et Valencia ANDEDI.
* Toute la famille ANDEDI, à qui ce travail est dédié.

Nous vous remercions pour votre amour, votre soutien et votre encouragement. Vous avez rendu possible l'accomplissement de ce travail.

**ANDEDI Andedi Joel**

# REMERCIEMENTS

Ce travail est le fruit consenti non seulement des auteurs de la rédaction de ce présent document, mais également des braves hommes et femmes dans l’ombre qui ont contribué à l’aboutissement de celui-ci, directement ou indirectement. Nous ne pouvons rédiger ce travail sans ouvrir une page pour leurs adresser nos remerciements.

Nos remerciements particuliers à notre directeur, Monsieur MATE Landry, et à notre rapporteur, Monsieur SAKASAKA Josmard, qui ont accepté de nous encadrer et nous démontrer une démarche scientifique à suivre pour bien arriver à terme de ce travail.

Nous remercions également le corps professoral de la faculté des sciences informatiques de l’Université Protestante au Congo, à vous tous qui avez contribué à notre formation.

Nous ne pouvons omettre de mentionner les amis et camarades qui nous ont accompagnés durant ce parcours avec leurs encouragements et motivations : NZALINGOMO Merdi, WASHEMA Emmanuel, LUAMBUA Bossworth, LUSAMBA Alice et MIANDA Sarah.

Nous aimerions écrire une ligne spéciale pour remercier celui qui est arrivé il y a peu de temps mais est devenu un mentor et une source d’inspiration, Monsieur Omer Pitou, pour ses conseils et orientations.

Nous ne pourrons clôturer cette page sans adresser notre vif remerciement à notre groupe de travail BAS MASTA, qui est devenu une famille prête à soutenir et nous permettant d’accomplir ce parcours. Nous pensons à : BACKELANT’S Joseph, KUKWABANTU Jonathan, HYBANI Hans, MANZOMBI Benedicte, MBALA Devernay, OTEMA Elie et TSHIMBALANGA Jean.

Un grand merci à tous nos très chers collègues de promotion pour votre parfaite collaboration.

Enfin, tous ceux qui ont contribué moralement ou financièrement à la réalisation de ce travail, du voisin de l’avenue jusqu’au service de restauration de l’université, trouvent par ces mots notre profonde reconnaissance.

# LISTE DES ABBREVIATIONS, ACRONYMES ET SIGLES

API : Application Programming interface

BtoB : Business to Business

BtoC : Business to Consumer

CGU : Conditions générales d’utilisations

CtoC : Consumer to Consumer

PERT : Program Evaluation Review Technique

UML : Unified Modeling Language

UP : Unified Process

REST : Representational state Transfer

SOAP : Simple Object Access Protocol

# LISTE DES TABLEAUX

[Tableau 2-1 identification des taches et durée des opérations du projet 39](#_Toc142517835)

[Tableau 2‑2 le cout de réalisation du projet par tache 42](#_Toc142517836)

[Tableau 2‑3 Description textuelle du Package Client 49](#_Toc142517837)

[Tableau 2‑4 Fiche de description du package Marchand 50](#_Toc142517838)

[Tableau 2‑5 Fiche de description du package Administrateur 51](#_Toc142517839)

[Tableau 3‑1 Règles de gestion 53](#_Toc142517840)

# LISTE DES FIGURES

[Figure 1‑1 Fonctionnement du commerce BtoC](#_Toc142580087) 17

[Figure 1‑2 Schéma de communication d'une API 24](#_Toc142580088)

[Figure 1‑3 Déroulement d’un acte de fiducie 32](#_Toc142580089)

[Figure 2-1 Schéma de fonctionnement de l'architecture MVC dans une application Spring Boot 36](#_Toc142580090)

[Figure 2-2 Réseau PERT 39](#_Toc142580091)

[Figure 2-3 Diagramme de contexte de l'API 44](#_Toc142580092)

[Figure 2-4 Diagramme de cas d'utilisation globale de l'API 45](#_Toc142580093)

[Figure 2-5 Diagramme de séquence de l'API 46](#_Toc142580094)

[Figure 2-6 illustration d'une application interagissant avec les API des tiers 47](#_Toc142580095)

[Figure 2-7 illustration de l'architecture à trois niveaux(3-tiers) 48](#_Toc142580096)

[Figure 2-8 Package de cas d'utilisation Client 49](#_Toc142580097)

[Figure 2-9 Package de cas d'utilisations Marchand 50](#_Toc142580098)

[Figure 3-1 Présentation du Diagramme de classe 54](#_Toc142580100)

[Figure 3-2 Présentation du diagramme de composants 55](#_Toc142580101)

[Figure 3‑3 Présentation du diagramme d'activités 56](#_Toc142580102)

[Figure 3‑4 Présentation du diagramme d'état-transition 57](#_Toc142580103)

[Figure 3‑5 Description du flow architecture 60](#_Toc142580104)

[Figure 3‑6 Présentation du diagramme de déploiement 64](#_Toc142580105)

# TABLE DES MATIERES

[EPIGRAPHE 1](#_Toc142652728)

[DEDICACES 2](#_Toc142652729)

[REMERCIEMENTS 3](#_Toc142652730)

[LISTE DES ABBREVIATIONS, ACRONYMES ET SIGLES 4](#_Toc142652731)

[LISTE DES TABLEAUX 5](#_Toc142652732)

[LISTE DES FIGURES 6](#_Toc142652733)

[TABLE DES MATIERES 7](#_Toc142652734)

[INTRODUCTION GENERALE 10](#_Toc142652735)

[0.1. MISE EN CONTEXTE 10](#_Toc142652736)

[0.2. PROBLEMATIQUE 11](#_Toc142652737)

[0.3. HYPOTHESES 12](#_Toc142652738)

[0.4. Méthodes et techniques 13](#_Toc142652739)

[a. Méthodes 13](#_Toc142652740)

[b. Techniques 13](#_Toc142652741)

[0.5. Objectifs de la recherche 14](#_Toc142652742)

[0.6. Contribution de l’étude 14](#_Toc142652743)

[0.7. Délimitation du travail 14](#_Toc142652744)

[a. Délimitation dans le temps 14](#_Toc142652745)

[b. Délimitation dans l’espace 15](#_Toc142652746)

[0.8. Division du travail 15](#_Toc142652747)

[Chapitre 1 REVUE DE LA LITTERATURE 16](#_Toc142652748)

[1.1. REVUE DE LA LITTERATURE EMPIRIQUE 16](#_Toc142652749)

[1.2. REVUE DE LA LITTERATURE THEORIQUE 20](#_Toc142652750)

[1.2.1. E-commerce 20](#_Toc142652751)

[a. Définition de l’e-commerce 20](#_Toc142652752)

[b. Formes de e-commerce 20](#_Toc142652753)

[c. Objectifs d’un e-commerce 22](#_Toc142652754)

[d. Avantages du e-commerce 23](#_Toc142652755)

[e. Inconvénients du commerce électronique 24](#_Toc142652756)

[f. Risques du E-commerce 24](#_Toc142652757)

[Exemples de célèbre e-commerce 25](#_Toc142652758)

[1.2.2. API 26](#_Toc142652759)

[1.2.2.1. L'histoire des API en quelques mots 26](#_Toc142652760)

[1.2.2.2. Qu'est-ce qu'une API ? 26](#_Toc142652761)

[1.2.2.3. Fonctionnement d’une API 26](#_Toc142652762)

[1.2.2.4. Styles architecturaux des apis 27](#_Toc142652763)

[1.2.2.5. Qu'est-ce qu'une API REST ? 29](#_Toc142652764)

[1.2.2.6. Qu'est-ce qu'une intégration d'API ? 29](#_Toc142652765)

[1.2.2.7. Quels sont les avantages de l'API REST ? 29](#_Toc142652766)

[1.2.2.8. Quels sont les différents types d'API ? 30](#_Toc142652767)

[1.2.2.9. Que sont les points de terminaison d'API et pourquoi sont-ils importants ? 30](#_Toc142652768)

[1.2.2.10. Comment sécuriser une API REST ? 31](#_Toc142652769)

[1.2.3. Escrow System 31](#_Toc142652770)

[1.2.3.1. Qu'est-ce qu'un acte de fiducie ? 32](#_Toc142652771)

[1.2.3.2. Gestion de l'acte de fiducie 32](#_Toc142652772)

[1.2.3.3. Considérations particulières 33](#_Toc142652773)

[Chapitre 2 CAPTURE DES BESOINS ET ELABORATION 33](#_Toc142652774)

[2.1. CONTEXTE DE L’ETUDE 33](#_Toc142652775)

[2.1.1. Présentation de l’application Marianu Style 33](#_Toc142652776)

[2.1.1.1. Spécification fonctionnelle de Marianu style 34](#_Toc142652777)

[2.1.1.2. Spécifications techniques 34](#_Toc142652778)

[2.1.1.3. Outils de développement 34](#_Toc142652779)

[2.1.1.4. Architecture Logicielle 35](#_Toc142652780)

[2.2. Analyse de l’existant 37](#_Toc142652781)

[2.2.1. Propositions de solutions 37](#_Toc142652782)

[2.3. ETUDES DE FAISABILITES 38](#_Toc142652783)

[2.3.1. Faisabilité Opérationnelle 38](#_Toc142652784)

[a. Identification et dénombrement des taches 38](#_Toc142652785)

[b. Réseau PERT 39](#_Toc142652786)

[c. Calcul de dates 40](#_Toc142652787)

[d. Calcul de marges 41](#_Toc142652788)

[1. Marge libre 41](#_Toc142652789)

[2. Marge Total 41](#_Toc142652790)

[3. Détermination du chemin critique 42](#_Toc142652791)

[2.3.2. Faisabilité Financière 42](#_Toc142652792)

[2.4. CAPTURE DES BESOINS 42](#_Toc142652793)

[2.4.1. Capture des besoins fonctionnels 43](#_Toc142652794)

[2.4.1.1. Enoncé sur le fonctionnement du nouveau système 43](#_Toc142652795)

[2.4.2. Captures des besoins techniques 47](#_Toc142652796)

[2.5. Elaboration 48](#_Toc142652797)

[2.5.1. Package Client 49](#_Toc142652798)

[2.5.2. Package Marchand 50](#_Toc142652799)

[2.5.3. Package Administrateur 51](#_Toc142652800)

[Chapitre 3 CONSTRUCTION ET TRANSITION 53](#_Toc142652801)

[3.1. CONSTRUCTION DU SYSTEME 53](#_Toc142652802)

[3.1.1. Modélisation statique 53](#_Toc142652803)

[3.1.2. Modélisation dynamique 55](#_Toc142652804)

[3.2. Implémentation du système 57](#_Toc142652805)

[3.2.1. Besoins techniques 58](#_Toc142652806)

[3.2.2. Présentation des codes sources 60](#_Toc142652807)

[3.2.3. Présentation d’interfaces 62](#_Toc142652808)

[3.3. TRANSITION VERS LE NOUVEAU SYSTEME 64](#_Toc142652809)

[CONCLUSION 65](#_Toc142652810)

[BIBLIOGRAPHIE 66](#_Toc142652811)

INTRODUCTION GENERALE

## MISE EN CONTEXTE

Les humains ont toujours cherché à échanger des biens et des services avec les autres. Au fil du temps, les modes de paiement ont évolué pour répondre aux besoins changeants de la société. Aujourd'hui, nous avons plus d'options de paiement à notre disposition que jamais auparavant, des espèces et des chèques aux cartes de crédit et aux crypto-monnaies.

Les méthodes de paiement se diversifient de nos jours. Les paiements en ligne deviennent de plus en plus populaires, notamment avec l'essor des services de paiement tels que PayPal et Stripe. Les paiements mobiles connaissent également une croissance rapide avec l'utilisation de services tels que Apple Pay et Google Wallet, et de divers services de réseaux de télécommunications tels que MPESA, Airtel Money et Orange Money.

Les méthodes de paiement modernes offrent de nombreux avantages tels que la commodité, la rapidité, la sécurité et l'anonymat. Mais cela cause aussi des problèmes et des inquiétudes chez les consommateurs, les entreprises et les gouvernements.

Voici quelques-uns des principaux enjeux des modalités de paiement modernes :

**Sécurité des transactions**. Les escroqueries et les piratages se multiplient avec les méthodes de paiement modernes. Les consommateurs et les entreprises doivent donc être vigilants pour protéger leurs informations financières et éviter les transactions frauduleuses.

**La protection de la vie privée.** Lorsqu'ils effectuent des paiements en ligne ou mobiles, les consommateurs partagent souvent des informations personnelles sensibles. Par conséquent, les entreprises doivent s'assurer que les données des clients sont stockées et traitées de manière sécurisée et respectueuse de la vie privée.

**Accessibilité.** Certaines méthodes de paiement modernes, telles que les paiements mobiles, nécessitent un accès à des technologies telles que les smartphones et la connectivité Internet. Cela peut représenter un défi pour ceux qui n'ont pas les moyens d'acquérir ou d'utiliser ces technologies.

**Coût.** Certaines méthodes de paiement modernes comportent des frais pour les consommateurs ou les entreprises, ce qui peut représenter un fardeau financier pour certains.

**Lois**. Les méthodes de paiement modernes traversent souvent les frontières et utilisent une technologie sophistiquée, qui peut être difficile à réglementer pour les gouvernements. Cela peut poser des problèmes en matière de protection des consommateurs, de lutte contre la criminalité financière et de garantie de la stabilité financière.

La dématérialisation de ces nouveaux systèmes de paiement était essentielle pour permettre leur déploiement à grande échelle. Auparavant, il fallait passer des heures à la banque pour effectuer un paiement. Et pour les vendeurs qui reçoivent et encaissent des chèques de se déplacer. Face à ce défi, l'utilisation d'une API était une solution innovante.

La technologie API (*Application Programming Interface*) agit comme un intermédiaire entre deux applications, essentiellement une passerelle qui permet aux logiciels de communiquer avec d'autres logiciels. Nous utilisons des API tous les jours sans même nous en rendre compte. Nous pouvons consulter la météo quotidienne sur nos smartphones. Il est utilisé par les marchands de commerce électronique qui fournissent des sites Web ou des applications permettant aux clients d'acheter des biens. Pour que les marchands de commerce électronique reçoivent de l'argent des clients payants, la technologie API facilite le processus de paiement.

En ce qui concerne les problèmes transactionnels, les API ont considérablement amélioré l'expérience client, les rendant plus rapides, plus sûres et moins chères.

## PROBLEMATIQUE

Le marché des paiements numériques continue de croître. En 2021, cette valeur est de 7,36 billions de dollars américains et devrait atteindre 15,27 billions de dollars américains en 2027. [[1]](#footnote-1)

Et les ventes mondiales du commerce électronique, le principal canal par lequel ces systèmes de paiement sont utilisés, dépasseront pour la première fois les 5 000 milliards de dollars, ce qui représente un cinquième des ventes au détail [[2]](#footnote-2).

Ces chiffres montrent clairement que le secteur du e-commerce est rentable. Il est clair que la pénétration de ce domaine dans la société africaine a été jusqu'à présent très faible. Le cabinet Statistia la prévoit à 40% d'ici 2024, le plus bas par rapport aux autres continents.

L'une des raisons de la lenteur de l'adoption du commerce électronique dans la société africaine est la crise de confiance entre les vendeurs et les acheteurs.

D'un côté, les internautes sont intéressés par l'achat des différents produits proposés sur la plateforme, mais craignent de se faire arnaquer, ne voudront pas régler des commandes qui ne seront jamais livrées. D'autre part, certains vendeurs craignent que des utilisateurs fictifs puissent commander des biens en saisissant une adresse inexistante afin de tromper. Ces frais de déplacement "inutiles" peuvent amener le vendeur à dépenser pour des services qui ne seront pas payés. Face à cette situation, nous sommes appelés à réfléchir sur cette question : Quelle solution informatique déployer pour résoudre la crise de confiance entre acheteurs et vendeurs dans le secteur du e-commerce africain ?

## HYPOTHESES

Parmi les recommandations énoncées par **Alain DUCASS et Jean-Marc KWADJANE** dans leur ouvrage sur *le commerce électronique en Afrique,* pour une meilleure intégration du commerce électronique en Afrique, on peut citer « Le développement des services de paiement inspirées des meilleurs pratiques ».[[3]](#footnote-3)

La mise en place d’une API pour le système de paiement en attente accessible à tout le e-commerce pourrait diminuer voir dissipée cette méfiance qui existe entre les vendeurs et les acheteurs.

Ce système est basé sur le principe juridique du dépôt fiduciaire ou « *escrow agreement* ». Un escrow est une construction juridique dans le cadre de laquelle un actif (actions, fonds, acte de titre, etc.) est remis à un tiers indépendant (escrow agent), qui le détiendra en fiducie jusqu’à ce qu’une condition contractuelle spécifique soit remplie ou qu’un ou d’autres événements surviennent. Lorsque la condition spécifiée est remplie ou l’événement survient, en fonction de ce qui est défini dans l’escrow agreement, l’escrow agent débloque l’actif pour le destinataire approprié.

Ce système permettra de rassurer d’une part l’acheteur qui va payer en amont que le fournisseur ne pourrait recevoir l’argent qu’après confirmation de la réception de la commande et peut obtenir son remboursement sans problème et d’autre part au vendeur que c’est un acheteur sérieux car il a déjà payé la commande en avance bien qu’il n’ayant pas encore effectivement encaissés l’argent.

## Méthodes et techniques

## Méthodes

Une méthode de recherche représente un ensemble de règles ou de procédures permettant d'atteindre un but dans les meilleures conditions (Introduction à la Recherche Scientifique, 2022).

La méthode est donc une ligne à suivre pour atteindre un but poursuivi. Voici donc les méthodes utilisées dans notre travail :

**La méthode UP ( Unified Process )**

Le processus unifié (UP) est une méthode de développement logiciel itératif et incrémental. Il est basé sur l'idée que le développement logiciel est un processus continu d'amélioration et d'évolution. Le UP est divisé en quatre phases principales : la phase d'initiation, la phase de conception, la phase de construction et la phase de test.

Le UP est une bonne méthode de développement pour notre API car il est flexible et adaptable.

## Techniques

Diverses techniques de recherche ont été utilisées dans la conception de ce travail, notamment :

* **Technique documentaire** :

Divers articles et documents de recherche sur l'économie numérique, la conception et le développement des systèmes de paiement et des API ont été collectés, analysés, efficacement intégrés et utilisés dans la recherche.

* **Technique d'observation** :

Grâce à cette technique, nous avons pu constater les faits et constater que les internautes des communautés africaines se méfient des applications e-commerce.

## Objectifs de la recherche

Ci-dessus, nous avons déploré la faible pénétration du commerce électronique en Afrique. A travers cette recherche, nous visons à développer des solutions innovantes qui feront de l'Afrique un environnement de choix pour les leaders du e-commerce, un *Eldorado*.

En outre ce travail permettra de mettre en place une API de paiement qui va intégrer divers services de paiement mobile disponibles en République Démocratique du Congo qui, une fois intégrée dans une plateforme/application ou de e-commerce, offre nativement la possibilité d’utiliser les services de mobile money disponible pour régler des paiements de manière électronique.

Il s’agit d’offrir :

* Au client final (ici le contributeur payeur) une expérience de paiement homogène et simple ;
* Aux développeurs une intégration facilitée ;

## Contribution de l’étude

L'apport principal de cette recherche dans le domaine du paiement porte sur l'application du principe d’entiercement (escrow) lors des transactions. Ce système aidera les différents acteurs à augmenter leur confiance auprès des parties prenantes et à dissiper les soupçons d’anarque ou d’escroquerie qui est implanté depuis l'avènement de ces technologies.

Du point de vue académique, ce travail pourrait démontrer l’utilisation de l’informatique dans d’autres domaines tel que l’économie. Mais aussi, la manière dont l’outil informatique peut être utilise pour faciliter l’applications des principes juridiques dans notre quotidien.

## Délimitation du travail

Les recherches effectuées pour la réalisation de ce travail s’est délimité en temps et en espace.

### Délimitation dans le temps

Les recherches ont été effectué sur une période allant du mois de janvier au mois d’aout 2023.

### Délimitation dans l’espace

Nous avons travaillé sur l’application de ce nouveau système de paiement dans le secteur de la vente des vêtements en ligne pour le moment. Bien qu’à terme, nous aimerons bien couvrir tout le secteur e-commerce.

## Division du travail

En plus de l'introduction générale et de la conclusion générale, ledit travail est divisé en trois chapitres organisés comme suit :

CHAPITRE PREMIER : REVUE DE LA LITTÉRATURE

Ce chapitre explique les différents termes clés utilises dans notre travail et expose les travaux de recherche antérieurs relatifs à notre sujet, lesquels servent de fondement théorique à notre étude.

CHAPITRE DEUXIÈME : CAPTURE DE BESOIN ET ÉLABORATION

Le deuxième chapitre décrit la méthodologie de notre étude, notamment la manière dont nous avons collecté et analysé les données afin de déterminer les besoins en matière de développement de notre API pour le système de paiement en attente.

CHAPITRE TROISIÈME : CONSTRUCTION ET TRANSITION

Le troisième et dernier chapitre présente la conception et la mise en œuvre de notre API, ainsi que les résultats obtenus. Nous y exposons également les perspectives de développement.

REVUE DE LA LITTERATURE

Ce chapitre a pour objectif de présenter une revue de la littérature empirique et théorique relative à la thématique abordée. La revue de la littérature empirique permettra de synthétiser les travaux antérieurs réalisés par différents chercheurs. Cette étape sera utile pour identifier les lacunes dans la recherche existante et pour situer notre contribution dans le contexte scientifique. En outre, la revue de la littérature théorique consistera en une définition des concepts clés ou terminologiques relatifs à notre thème d'étude.

Cette revue de la littérature théorique permettra de clarifier les concepts utilisés dans notre recherche, ainsi que d'approfondir notre compréhension du cadre théorique dans lequel s'inscrit notre travail. Cette étape est essentielle pour garantir la rigueur scientifique de notre recherche.

## REVUE DE LA LITTERATURE EMPIRIQUE

Comme mentionnée ci-haut, cette section permettra de synthétiser les travaux antérieurs réalisés par différents chercheurs sur la conception et développement des Api pour le système de paiement. Cette étape sera utile pour identifier les lacunes dans la recherche existante afin de pouvoir situer notre contribution dans le contexte scientifique.

1. **Développement d’une API Java de paiement en ligne sur les sites e-commerce, Osias Noël Nicodème Finagnon TOSSOU**

L’objet principale de son travail était de construire un système de paiement en ligne qui s’accorde à la situation de la population africaine en générale et béninoise en particulier. Cette api pourrait aider les détenteurs des e-commerces à intégrer facilement une solution de paiement dans leurs plateformes de commerce électronique.

L'auteur a développé cette solution en utilisant la méthodologie Scrum et la modélisation UML

Pour résoudre ce problème, l'auteur a implémenté une API sécurisée en utilisant le langage de programmation Java EE et POSTGRES comme SGBD. Cette API vous permet de gérer les comptes marchands, de gérer les paiements, etc. Cette API utilise les méthodes de paiement bancaires existantes : VISA, Mastercard...

1. **MISE EN ŒUVRE D'UNE SOLUTION DE PAIEMENT EN LIGNE POUR LE COMMERCE ELECTRONIQUE (E-COMMERCE), M. GUIGUEMDE G Louis Patrick M. TRAORE Sy Ali**

L’objet principale de sa recherche était de mettre en place de solutions de paiement e-commerce afin d’élargir les offres de services ainsi que d’améliorer les performances de la Banque Burkinabé de l'Agriculture et de la Banque Commerciale (BACB).

Après avoir observé la croissance du commerce électronique, les auteurs a remarqué que la question de la dématérialisation du commerce électronique était devenue une priorité. A cet effet, il a fallu chercher des moyens pour rester en contact avec les clients malgré la distance. En outre les questions liées aux couts ainsi que la sécurité des paiements à surgir car le risque que les transactions soient interceptées par des parties malveillantes devient de plus en plus élevé.

Afin d’analyser ce système, la méthode UP avec le langage de modélisation UML a été utilisée.

Il a clairement identifié le domaine de recherche et collecté des informations auprès des utilisateurs pour avoir une compréhension claire du flux d'informations.

Pour résoudre ce problème, les auteurs ont amélioré le système existant en mettant en œuvre une solution beaucoup plus rapide et plus sûre tout en tenant compte des diverses limitations existantes. Ce nouveau système a apporté plusieurs améliorations, notamment : Authentification à chaque connexion, ajout d'acteurs pour maintenir le système, etc...

Bien que ces systèmes aient permis d’avoir une avancée importante dans le domaine du paiement électronique, mais nous voulons clairement que les problèmes évoqués dans notre problématique persistent toujours. A cet effet, nous croyons sincèrement que notre travail pourrait aider d’une manière significative à combler ce manque.

1. **REALISATION D’UN AGREGATEUR DE PAIEMENT : KKIAPAY, MELAINE KIOSSOU**

Le travail propose une solution d'agrégation de divers services de paiement mobile disponibles en République du Bénin. Cette solution, une fois intégrée dans une plateforme/application de collecte d'argent ou de e-commerce, offre nativement la possibilité d'utiliser aussi bien MTN Mobile Money, Moov Money, les cartes bancaires Visa ou MasterCard ou ECOBANK Xpress pour régler des paiements de manière électronique.

Le travail a été réalisé en utilisant la méthode Scrum et le langage UML. La méthode Scrum est une pratique agile qui permet de développer des produits complexes de manière itérative et incrémentale. Le langage UML est un langage de modélisation graphique qui permet de représenter des systèmes complexes.

La solution proposée a été testée avec succès sur une plateforme de e-commerce. Les résultats des tests ont montré que la solution est facile à utiliser et sécurisée.

Le travail a permis de développer une solution innovante qui répond aux besoins des populations et des entreprises du Bénin. La solution proposée permet de faciliter les paiements en ligne et de réduire les coûts pour les entreprises.

La solution proposée a le potentiel de révolutionner le secteur du paiement en ligne au Bénin. Elle permettra de favoriser l'inclusion financière et de stimuler l'économie du pays.

1. **SERVICE DE GESTION DE PAIEMENT GETTIME, DAMIEN ULRICH**

Le travail de Ulrich a pour but de choisir le meilleur fournisseur de service de paiement pour une entreprise. Il est difficile de trouver soi-même le fournisseur qui correspond le mieux à son entreprise car ils ne sont pas tous compatibles avec tous les langages et tous ne proposent pas toutes les fonctionnalités de paiement. L'auteur a donc retranscrit de manière harmonisée les capacités des dix fournisseurs de service de paiement les plus connus. Grâce à ce listing de fournisseurs, il a créé une solution afin de déterminer les fournisseurs qui correspondent à votre entreprise. Il termine ce document en expliquant les marches à suivre pour mettre en place un système de paiement de deux manières différentes. Premièrement avec une application et deuxièmement avec un site web utilisant WordPress

Les dix fournisseurs de service de paiement les plus connus sont :

* Amazon Pay
* Datatrans
* HiPay
* Mangopay
* Mollie
* Payline
* PayPal
* PayZen
* SecurionPay
* Stripe

L'auteur a sélectionné ces fournisseurs en fonction de leur popularité, de leur offre de services et de leur compatibilité avec les besoins des entreprises. Il a ensuite regroupé les informations sur chacun des fournisseurs dans un tableau, afin de faciliter la comparaison.

Le tableau comprend les informations suivantes :

* Le nom du fournisseur
* La description du service
* Les fonctionnalités disponibles
* Les pays dans lesquels le fournisseur est disponible
* Les frais de transaction

L'auteur a également expliqué les étapes à suivre pour mettre en place un système de paiement avec chacun des fournisseurs. Il a illustré ses explications avec des schémas.

Le travail de Ulrich est un outil précieux pour les entreprises qui cherchent à choisir le meilleur fournisseur de service de paiement. Il permet de comparer facilement les différents fournisseurs et de trouver celui qui correspond le mieux aux besoins de l'entreprise.

En résumé, les travaux existants ont contribué au domaine des systèmes de paiement, mais ils n'ont pas encore abordé les questions spécifiques de notre problématique. Pour combler ce vide et amener les utilisateurs qui ont encore une certaine méfiance par rapport aux systèmes de paiement proposés par les e-commerces, nous allons mettre en place notre API de paiement en attente.

Nous pensons que notre API de paiement en attente sera un atout précieux pour les e-commerces et leurs clients. Elle permettra aux e-commerces de gagner de nouveaux clients et aux clients de se sentir plus en sécurité lorsqu'ils effectuent des achats en ligne.

## REVUE DE LA LITTERATURE THEORIQUE

Dans cette section, il s’agira de présenter les concepts clés liés à notre travail tels que : E-commerce, API et Escrow system.

### E-commerce

### Définition de l’e-commerce

D’après, le Professeur Pescie H. Mampuya, « le e-commerce consiste à utiliser internet et le web pour effectuer des transactions commerciales. De manière plus formelle, le e-commerce définit les transactions commerciales numériques qui ont lieu entre les entreprises et les particuliers »

### Formes de e-commerce

Il existe trois formes principales de commerce électronique, différenciées par la clientèle et la source de revenus : "De l'entreprise au consommateur" (BtoC), "De l'entreprise à l'entreprise" (BtoB),"Du consommateur au consommateur" (CtoC).

* **Business to Consumer (BtoC)**

Business-to-Consumer (B2C) est le commerce électronique traditionnel d'entreprise à consommateur, c'est-à-dire la vente de biens et de services à des clients individuels via des sites Web.

Le commerce électronique B2C peut prendre la forme de pages de vente uniquement Web ou de compléments virtuels aux magasins physiques. L'objectif est de développer la clientèle et la notoriété grâce à la vente en ligne. Les revenus B2C proviennent des ventes au détail.

Les sites dits de vente aux particuliers sont de type B2C, et le principe est de proposer des remises importantes sur des produits pendant une durée limitée.

Le dropshipping est un site de vente en ligne qui ne tient pas d'inventaire et n'expédie pas de produits. Concrètement, les détaillants vendent des marchandises pour le compte de tiers et externalisent toute la logistique à des sous-traitants. Les clients reçoivent leurs achats directement du grossiste ou du fabricant.

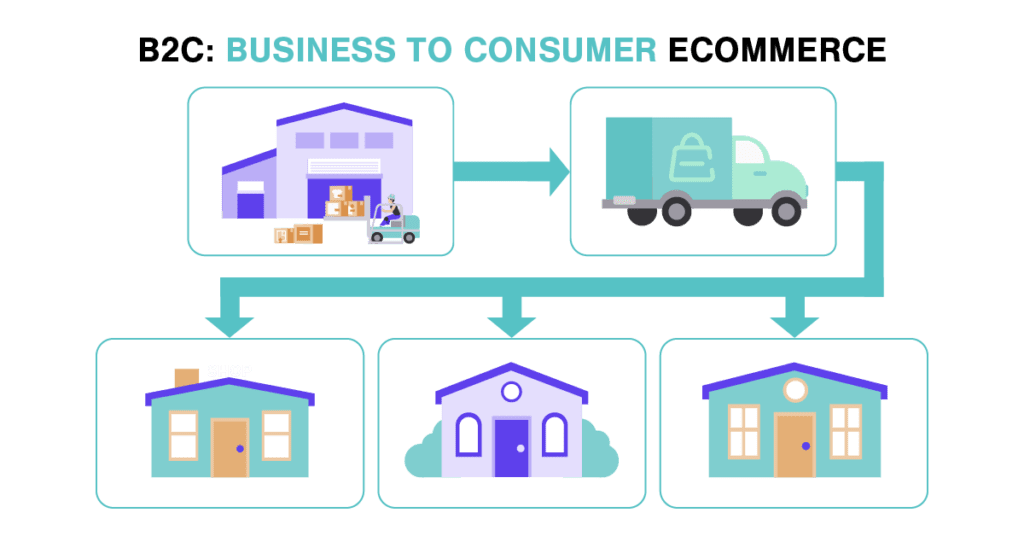


Figure ‑ Fonctionnement du commerce BtoC

* **Business to Business (BtoB)**

Business to Business (B2B) est le commerce électronique d'une entreprise à une autre. Il s'agit par exemple d'un échange entre deux concessionnaires ou entre un fournisseur et une entreprise. Les revenus B2B proviennent de la vente de biens ou de services aux entreprises.

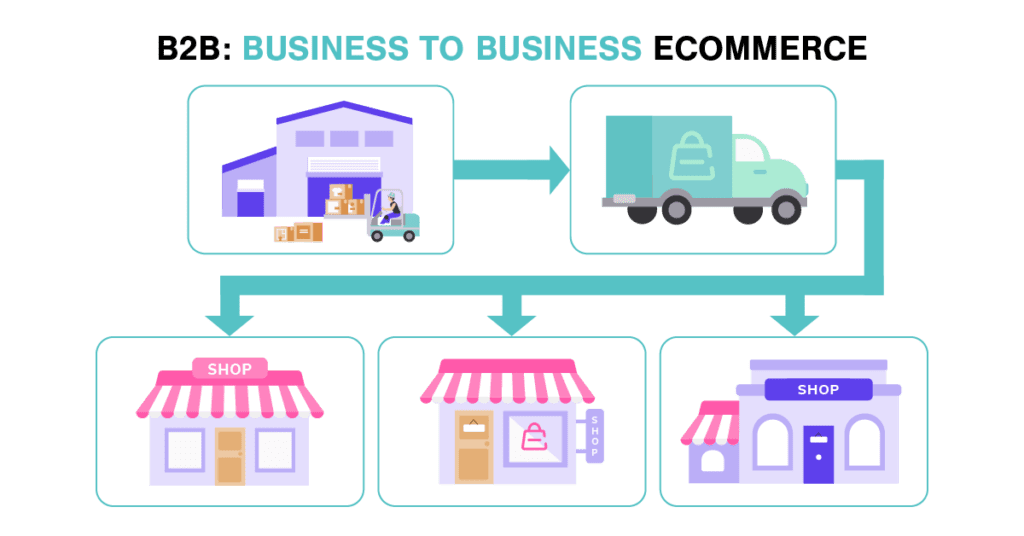


Figure ‑2 Fonctionnement du commerce BtoB

Figure ‑ business

* **Consumer to Consumer (CtoC)**

Consumer to Consumer (C2C) fait référence à une plateforme d'échange de consommateur à consommateur, l'équivalent internet des petites annonces. Ce site se limite à fournir une interface et un système d'échange de biens ou services.

Les revenus C2C peuvent provenir soit :

1. **Pourcentage des ventes**. Les grands sites d'enchères taxent les vendeurs lorsqu'une transaction a lieu entre deux clients (par exemple 1% à 5% du prix de vente final).

2. **Publicité**. Les sites de vente directe de consommateur à consommateur génèrent des récompenses grâce à la publicité et à la publicité payante. Les vendeurs de marchandises peuvent choisir de payer des frais de publicité pour augmenter la visibilité de leurs annonces.

**3. Une combinaison des deux.**

Figure ‑3 Fonctionnement du commerce CtoC

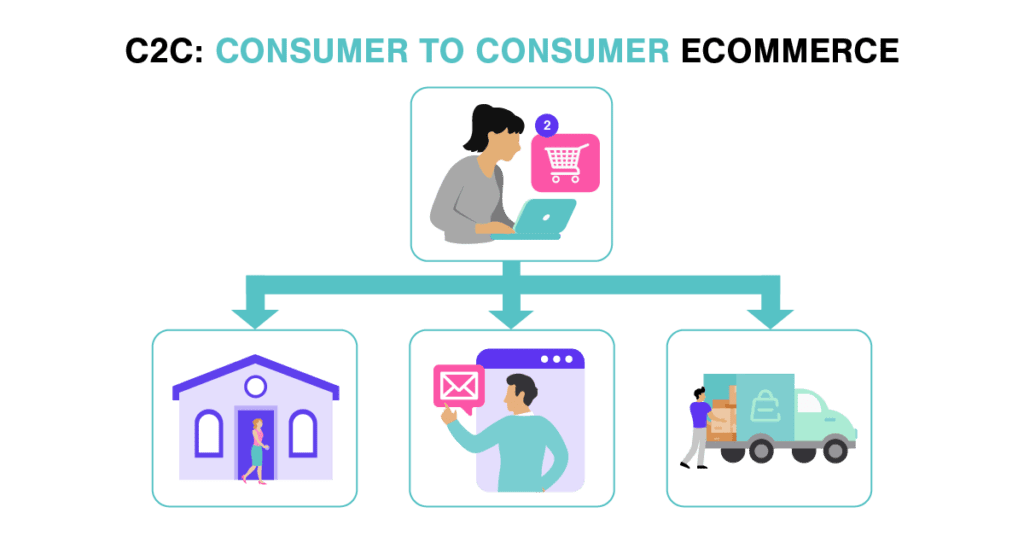


Figure ‑ Fonctionnement du commerce CtoC

### Objectifs d’un e-commerce

Aujourd'hui, les sites e-commerce sont un moyen de se simplifier la vie et de permettre l'achat de produits dans toutes les catégories. Mais surtout, sentez-vous à l'aise dans votre propre environnement à travers un écran. En effet, grâce aux sites e-commerce, il est possible d’effectuer les achats rapidement et en toute sécurité.

Parmi les objectifs du e-commerce, nous citons :

* **Acquisition et collecte de prospects**. Les sites de vente e-commerce ont un objectif précis : augmenter le nombre de visites et augmenter le temps que les visiteurs passent dans votre boutique en ligne. Étant donné que le commerce électronique est encore un produit relativement nouveau sur le marché en ligne, il faut plusieurs visites pour se faire connaître ou reconnaître dans le monde entier.
* **Fidélisation du processus d'achat pour les mêmes clients**. Le principal objectif d'un site Web marchand est de mieux connaître la clientèle, de renforcer leur intérêt et de mettre en valeur cet avantage dans lors des campagnes marketing.
* **Convertir les visiteurs en clients**. Pour qu'un site de vente prospère dans le monde virtuel, il a besoin d'une plus grande visibilité pour attirer les visiteurs. Plusieurs critères sont pris en considération, comme la qualité du design, qui contribue à améliorer la perception esthétique du site et de son contenu. Un contenu et des fonctionnalités bien organisés permettent aux clients d'accéder facilement aux informations qu'ils recherchent.

### Avantages du e-commerce

* **Les clients achètent plus rapidement**. Pour les clients, le commerce électronique leur permet d'acheter n'importe quand, n'importe où. Cela aide les acheteurs à obtenir plus rapidement les produits dont ils ont besoin.
* **Les entreprises peuvent facilement acquérir de nouveaux clients**. Le commerce électronique permet également aux entreprises d'atteindre plus facilement de nouveaux clients dans le monde entier. Les sites de commerce électronique ne sont pas associés à un seul emplacement géographique. En fait, il est ouvert et accessible à tous les clients qui nous visitent en ligne. Les marques, d'autre part, ont la possibilité de se connecter avec un large public pertinent qui est prêt à acheter.
* **Réduction des coûts d'exploitation.** De plus, les détaillants de commerce électronique peuvent ouvrir des succursales avec des coûts d'exploitation minimes. En effet, il ne nécessite pas de magasin physique (et de personnel qui y soit stationné). À mesure que les ventes augmentent, les marques peuvent facilement développer leur activité. De plus, il n'est pas nécessaire d'investir dans de grands biens immobiliers ou d'embaucher un grand nombre de travailleurs. Cela signifie que la marge bénéficiaire globale sera plus élevée.
* **Expérience personnalisée.** Grâce à l'automatisation et à l'utilisation des données, les entreprises peuvent offrir des expériences en ligne hautement personnalisées à leurs clients de commerce électronique.

### Inconvénients du commerce électronique

* **Interaction limitée avec les clients.** Sans contact en face à face, il peut être plus difficile de comprendre les souhaits, les besoins et les préoccupations de vos clients e-commerçants.
* **Les défaillances technologiques peuvent affecter la capacité de vente**. Si le site Web de commerce électronique d'une entreprise est lent, en panne ou indisponible pour les clients, cela signifie que les ventes sont impossibles. Les pannes de site et les défaillances technologiques peuvent perturber les relations avec les clients et avoir un impact négatif sur les résultats d'une entreprise.
* **Ne peut pas être testé ou utilisé**. Pour les acheteurs qui souhaitent voir et toucher des produits, en particulier dans le secteur des biens physiques tels que les vêtements, les chaussures et les produits de beauté, l'expérience de commerce électronique peut être limitante. Mais avec l'aide de vidéos, d'images de produits et même de la technologie de réalité virtuelle, les entreprises trouvent de nouvelles façons de transcender cet aspect de l'expérience d'achat en ligne. Toute personne souhaitant se lancer dans le commerce électronique doit être consciente des risques du commerce électronique.

### Risques du E-commerce

**La sécurité des infrastructures est compromise.** Les données peuvent être exposées à des attaques de pirates et de hackers. Dans certains cas, ils piratent des sites Web pour voler des informations personnelles aux visiteurs et aux propriétaires. Pour éviter d'être victime de ce danger, l’ajout d’une couche supplémentaire au site Web via les formulaires de contact et les champs d'inscription ainsi qu’une bonne politique de sécurité s’avère nécessaire.

**Résoudre les litiges entre clients et vendeurs**. Un client peut décider d'annuler un produit sélectionné parce que ce n'est pas ce qu'il attendait. Certaines descriptions perfectionnent le produit dans le but d'attirer des milliers de clients. De plus, il se peut que l’acheteur soit amené à effectuer plusieurs paiements ou qu’il ne reçoive pas les articles qu’il a acheté et payé.

**Trafic par le biais de cartes de crédit volées.** Les propriétaires de sites de commerce électronique doivent être en mesure d'identifier les transactions suspectes. La qualité de la sécurité doit être suffisamment élevée pour identifier si une transaction est effectuée par le titulaire de la carte de crédit ou par un pirate informatique.

**Vol d'identité.** La méthode d'identification doit être très complexe et ne doit pas être déchiffrable par un tiers. Cela ne doit pas être facile. Deux ou plusieurs systèmes d’authentification sont possibles. Ces méthodes peuvent améliorer la sécurité du système de l’entreprise et empêcher l'intrusion d'utilisateurs malveillants, réduisant ainsi le risque de transactions de commerce électronique.

**Manque de sécurité du côté client du commerce électronique.** Le paiement des marchandises est effectué dans une entreprise de commerce électronique, tout comme l'achat en ligne. Cela n'est souvent pas rassurant pour les clients qui craignent la perte ou la propagation de données. Pire encore, ils estiment que l'utilisation des cartes de crédit risque de les faire grimper dans les rangs des victimes de la fraude électronique.

### Exemples de célèbre e-commerce

* **Amazon** : Amazon est le plus grand site e-commerce au monde. Il a été fondé en 1994 par Jeff Bezos. Amazon propose une large gamme de produits, de l'électronique aux vêtements, en passant par les produits alimentaires et les articles ménagers. Amazon est connu pour son excellent service client et ses prix bas.
* **eBay** : eBay est un site de vente aux enchères en ligne. Il a été fondé en 1995 par Pierre Omidyar. eBay permet aux particuliers et aux entreprises de vendre des produits aux enchères. eBay est connu pour sa large sélection de produits et ses prix attractifs.
* **Walmart** : Walmart est une chaîne de magasins de détail. Elle a été fondée en 1962 par Sam Walton. Walmart propose une large gamme de produits, de l'alimentation aux produits ménagers, en passant par l'électronique et les vêtements. Walmart est connu pour ses prix bas et sa large sélection de produits.
* **Target** : Target est une chaîne de magasins de détail. Elle a été fondée en 1962 par George Dayton. Target propose une large gamme de produits, de l'alimentation aux produits ménagers, en passant par l'électronique et les vêtements. Target est connu pour ses prix abordables et son design moderne.
* **Etsy** : Etsy est un site de vente en ligne qui met en relation des artisans et des acheteurs du monde entier. Il a été fondé en 2005 par Rob Kalin, Chris Wanstrath et Haim Schoppik. Etsy propose une large gamme de produits faits à la main, tels que des bijoux, des vêtements, des objets de décoration, des œuvres d'art et des produits alimentaires. Etsy est connu pour sa communauté d'artisans passionnés et pour la qualité de ses produits.

### API

### L'histoire des API en quelques mots

Les API sont apparues aux débuts de l'informatique, avant l'avènement des ordinateurs personnels. À l'époque, il était principalement utilisé comme bibliothèque de système d'exploitation. Ils transportaient parfois des messages entre les mainframes, mais presque tous étaient locaux au système sur lequel ils s'exécutaient. Près de 30 ans plus tard, les API ont quitté l'environnement sur site. Au début des années 2000, il est devenu célèbre pour l'intégration de données à distance.

### Qu'est-ce qu'une API ?

Une API est un mécanisme qui permet à deux composants logiciels de communiquer entre eux à l'aide d'un ensemble de définitions et de protocoles. Par exemple, le système logiciel d'une station météorologique contient des données météorologiques quotidiennes. Une application météo pour téléphone mobile "communique" avec ce système via une API et affiche des mises à jour météo quotidiennes sur le téléphone.

API, d'autre part, fait référence à une interface de programmation d'application. Dans le contexte des API, le terme « application » fait référence à un logiciel qui exécute une fonction spécifique. Cette interface peut être vue comme un contrat de service entre deux applications. Ce contrat définit la manière dont les deux parties communiquent entre elles par le biais de demandes et de réponses. La documentation de l'API fournit des informations sur la façon dont les développeurs construisent ces demandes et réponses.

### Fonctionnement d’une API

L'architecture de l'API est généralement décrite en termes de clients et de serveurs. Par conséquent, l'application qui envoie la requête s'appelle le client et l'application qui envoie la réponse s'appelle le serveur. Ainsi, dans notre exemple météo, la base de données météo de bureau est le serveur et l'application mobile est le client.

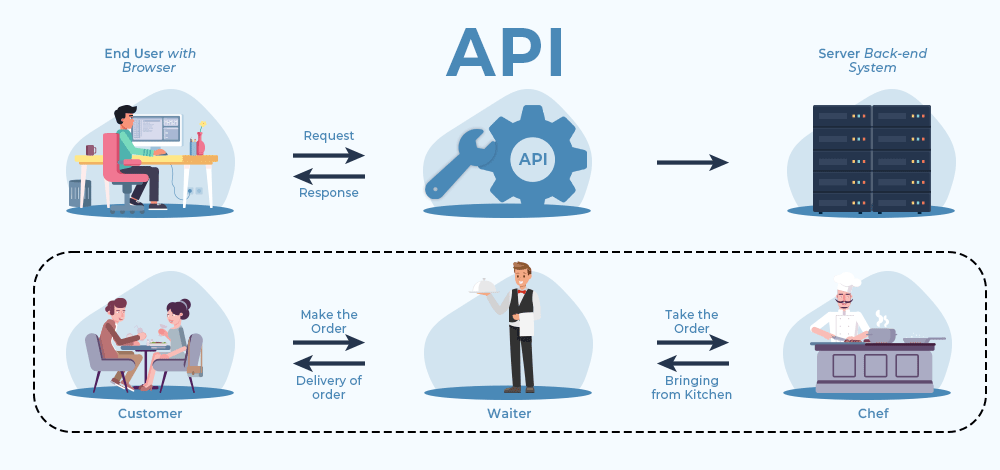


Figure ‑ schéma de communication d'une API

### Styles architecturaux des apis

Un style architectural d'API est un ensemble de principes et de pratiques qui guident la conception d'une API. Il comprend des éléments tels que la structure, les méthodes, les formats de données et les protocoles utilisés par l'API.

Nous allons passer en revue, les styles architecturaux le plus utilisés.

* SOAP (Simple Object Access Protocol) est un style architectural qui utilise XML pour structurer les données et SOAP pour invoquer des services distants. Il est couramment utilisé dans les environnements d'entreprise et est réputé pour sa sécurité et sa fiabilité.
* REST (Representational State Transfer) est un style architectural qui utilise le protocole HTTP pour invoquer des services distants. Il est plus léger que SOAP et est considéré comme plus facile à utiliser et à développer.
* GraphQL est un style architectural qui permet aux clients de demander des données spécifiques au serveur. Il est considéré comme plus efficace que REST car il n'envoie que les données dont le client a besoin.
* gRPC (Google Remote Procedure Call) est un style architectural qui utilise le protocole HTTP/2 pour invoquer des services distants. Il est considéré comme plus performant que SOAP et REST.
* WebSockets est un protocole de communication bidirectionnel qui permet aux clients d'envoyer et de recevoir des données du serveur en temps réel. Il est souvent utilisé pour des applications telles que la messagerie en direct et la diffusion de vidéos.
* Webhooks sont des déclencheurs qui sont appelés lorsqu'un événement se produit sur un serveur. Ils sont souvent utilisés pour envoyer des notifications aux clients ou pour déclencher d'autres actions.
* MQTT (Message Queuing Telemetry Transport) est un protocole de messagerie léger qui est souvent utilisé dans les applications IoT. Il permet aux appareils de s'abonner à des thèmes et de recevoir des messages lorsqu'un nouveau message est publié sur un thème.
* AMQP (Advanced Message Queuing Protocol) est un protocole de messagerie plus complexe que MQTT. Il offre un ensemble de fonctionnalités plus riche, telles que la capacité de gérer des transactions et d'assurer la fiabilité de la livraison des messages.

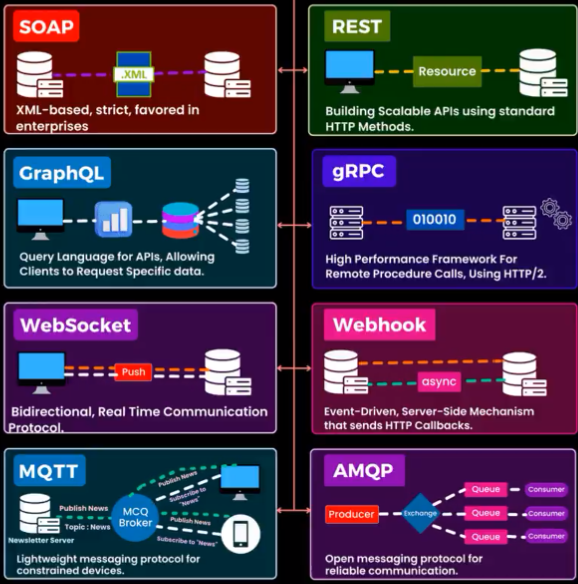


Figure 1‑‑. Styles architectural des APIs

Nous allons faire une série des questions-réponses afin de mieux explorer ce vaste domaine des apis.

### Qu'est-ce qu'une API REST ?

REST signifie *Representational State Transfer*. REST définit un ensemble de fonctions telles que GET, PUT et DELETE qui permettent aux clients d'accéder aux données du serveur. Les clients et les serveurs échangent des données à l'aide du protocole HTTP.

Une caractéristique clé des API REST est qu'elles sont sans état. Sans état signifie que le serveur ne stocke pas les données client entre les requêtes. La demande d'un client à un serveur est similaire à l'URL que vous entrez dans votre navigateur pour accéder à un site Web. La réponse du serveur consiste en des données simples qui n'ont pas la représentation graphique classique d'une page Web.

### Qu'est-ce qu'une intégration d'API ?

Une intégration d'API est un composant logiciel qui met automatiquement à jour les données entre un client et un serveur. Des exemples d'intégration API incluent la synchronisation automatique des données de la galerie d'images de votre téléphone vers le cloud, ou la synchronisation automatique de l'heure et de la date de votre ordinateur portable lorsque vous changez de fuseau horaire.

### Quels sont les avantages de l'API REST ?

Les API REST présentent quatre avantages principaux :

**1. Intégration**

Les API sont utilisées pour intégrer de nouvelles applications dans des systèmes logiciels existants. Cela permet d'accélérer le développement car vous n'avez pas à réécrire chaque fonctionnalité à partir de zéro. Tirez parti du code existant à l'aide d'API.

**2. Innovation**

L'introduction de nouvelles applications peut changer des industries entières. De plus, les entreprises doivent réagir rapidement et soutenir l'introduction rapide de services innovants. Pour y parvenir, les modifications au niveau de l'API doivent être effectuées sans réécrire l'intégralité du code.

**3. Expansion**

Les API offrent aux entreprises un moyen unique de répondre aux besoins des clients sur toutes les plateformes. Par exemple, l'API Maps permet l'intégration d'informations cartographiques sur des sites Web, Android, iOS, etc. Toute entreprise peut utiliser des API gratuites ou payantes pour fournir un accès similaire aux bases de données internes.

**4. Entretien facile**

Les API agissent comme des passerelles entre deux systèmes. Chaque système est obligé d'apporter des modifications internes afin que l'API ne soit pas compromise. De cette façon, les modifications futures du code par une partie n'affecteront pas l'autre partie.

### Quels sont les différents types d'API ?

Les API sont classées à la fois par architecture et par portée d'utilisation. Nous avons déjà couvert les principaux types d'architectures d'API, passons donc à leur portée.

**API privée.** Ils sont internes à l'entreprise et ne sont utilisés que pour connecter les systèmes et données internes.

**API publique**. Ils sont accessibles au public et peuvent être utilisés par n'importe qui. Ces types d'API peuvent ou non avoir des autorisations ou des coûts associés.

**API partenaire.** Celles-ci ne sont accessibles qu'aux développeurs externes autorisés pour faciliter les partenariats entre entreprises. Exemple : API de paiement avec MPESA[[4]](#footnote-4)

**API composite**. Ils combinent deux ou plusieurs API différentes pour gérer les exigences et les comportements complexes du système.

### Que sont les points de terminaison d'API et pourquoi sont-ils importants ?

Un point de terminaison API est le point de contact final d'un système de communication API. Cela inclut les URL des serveurs, des services et d'autres emplacements numériques spécifiques où les informations sont envoyées et reçues entre les systèmes. Les points de terminaison d'API sont importants pour l’entreprise pour deux raisons principales :

**1. Sécurité**

Les points de terminaison d'API rendent le système vulnérable aux attaques. La surveillance des API est essentielle pour prévenir les abus.

**2. Performances**

Les points de terminaison d'API, en particulier ceux à fort trafic, peuvent provoquer des goulots d'étranglement et avoir un impact sur les performances du système.

### Comment sécuriser une API REST ?

Toutes les API doivent être protégées par une authentification et des contrôles appropriés. Il existe deux manières principales de sécuriser votre API REST :

**1. Jeton d'authentification**

Ceux-ci sont utilisés pour autoriser les utilisateurs à effectuer des appels API. Un jeton d'authentification vérifie qu'un utilisateur est bien celui qu'il prétend être et qu'il a accès à un appel d'API particulier. Par exemple, lors de la connexion à un serveur de messagerie, les clients de messagerie utilisent des jetons d'authentification pour sécuriser l'accès.

**2. Clé API**

Les clés API sont utilisées pour vérifier le programme ou l'application qui a effectué l'appel API. Ils identifient l'application et vérifient qu'elle dispose des autorisations nécessaires pour des appels d'API spécifiques. Les clés d'API ne sont pas aussi sécurisées que les jetons, mais vous pouvez surveiller vos API et collecter des données sur leur utilisation. Lorsque vous visitez divers sites Web, vous remarquerez peut-être que les URL de votre navigateur contiennent de longues chaînes et des chiffres. Cette chaîne est la clé API que votre site Web utilise pour effectuer des appels API internes.

### 1.2.3. Escrow System

Un compte en fiducie est un accord contractuel entre les différentes parties à une transaction par lequel des tiers indépendants et de confiance reçoivent et versent des fonds ou des documents au nom des différentes parties impliquées dans la transaction.

Un dépôt fiduciaire d'origine anglo-saxonne est un mode de paiement destiné à réduire le risque lié aux transactions entre deux parties qui se méfient l'une de l'autre. Ce mode de paiement est de plus en plus utilisé dans le commerce électronique, y compris dans des domaines tels que l'immobilier et la propriété intellectuelle.

### 1.2.3.1. Qu'est-ce qu'un acte de fiducie ?

Un accord de fiducie est un contrat qui définit les termes de l'accord entre les parties concernées et les responsabilités de chaque partie. Les actes de fiducie impliquent généralement un tiers indépendant, appelé agent fiduciaire, qui détient des actifs de valeur en garde jusqu'à ce que les conditions contractuelles spécifiées soient respectées. Cependant, les conditions doivent être pleinement expliquées à toutes les parties concernées.



Figure 1‑‑ Déroulement d’un acte de fiducie

### 1.2.3.2. Gestion de l'acte de fiducie

Dans un acte de fiducie, une partie (généralement le déposant) laisse des fonds ou des actifs à un agent fiduciaire jusqu'à ce que l'acte soit exécuté. Une fois les termes du contrat remplis, le liquidateur libère les fonds ou autres actifs aux bénéficiaires. Les accords d'entiercement sont couramment utilisés pour une variété de transactions financières, en particulier les transactions importantes telles que l'immobilier et les ventes en ligne.

L'accord d'entiercement doit détailler les termes entre toutes les parties impliquées. La mise en place d'un tel système garantit que toutes les obligations des parties sont remplies et que les transactions sont effectuées en toute sécurité.

Les actes de fiducie incluent généralement des informations telles que :

* + L’identité du destinataire prévu ;
  + Définitions de tous les termes liés au contrat ;
  + Modalités et conditions détaillées du fonds fiduciaire et du déblocage de ce fonds.
  + L’utilisation acceptable des fonds par les fiduciaires ;
  + Obligations et responsabilités fiduciaires ;
  + Frais et dépenses du bénéficiaire
  + Compétence et lieu en cas de litige ;

La plupart des accords d'entiercement sont conclus lorsqu'une partie souhaite s'assurer que l'autre partie a satisfait à certaines conditions ou obligations avant de procéder à l'accord. Par exemple, les vendeurs peuvent conclure des accords d'entiercement pour s'assurer que les acheteurs potentiels sont en mesure de lever des fonds avant la conclusion de la vente. Si l'acheteur ne parvient pas à lever des fonds, la transaction peut être annulée et l'acte fiduciaire résilié.

Un acte de fiducie offre une sécurité en confiant des actifs à un fiduciaire pour la garde jusqu'à ce que chaque partie ait rempli ses obligations en vertu du contrat.

### 1.2.3.3. Considérations particulières

Dans les transactions commerciales, il peut être dans l'intérêt d'une partie de procéder uniquement si l'autre partie est absolument certaine qu'elle peut respecter ses obligations. C'est là que l'utilisation d'un accord de fiducie peut aider.

Par exemple, une entreprise qui achète des marchandises à l'international veut être sûre que sa contrepartie peut livrer les marchandises. À l'inverse, le vendeur veut s'assurer que le paiement est effectué lorsque les marchandises sont expédiées à l'acheteur. Les deux parties peuvent conclure un accord d'entiercement pour garantir la livraison et le paiement. Vous pouvez convenir que l'Acheteur demandera irrévocablement à l'Agent de détenir les fonds sous séquestre et de remettre les fonds au Vendeur une fois les marchandises arrivées.

# CAPTURE DES BESOINS ET ELABORATION

## 2.1. CONTEXTE DE L’ETUDE

### 2.1.1. Présentation de l’application Marianu Style

Nous allons déployer notre api de paiement dans une application e-commerce de ventes de vêtements. Dans cette section, nous allons parler de cette application dénommé « Marianu style ». Nous allons détailler ces fonctionnalités, son architecture ainsi que les différentes technologies utilisées pour son développement.

### 2.1.1.1. Spécification fonctionnelle de Marianu style

Marianu style est un site e-commerce qui permet l’achat des vêtements en ligne. Cette application offre une gamme de choix à toute catégorie d’utilisateurs (adulte, enfants, homme et femme).

Le but principal de cette application est de permettre à ceux qui n’ont pas forcément la capacite de se déplacer pour acheter les articles dans des magasins, de pouvoir en avoir à n’importe quel endroit et moment. Elle offre un service de livraison à domicile.

### 2.1.1.2. Spécifications techniques

* Type d’application : Application web
* Environnement concernes : Web
* Architecture : Architecture client-serveur 2 tiers
* Type de communication : Communication synchrone
* Système de gestion de base de données : MySQL
* Environnement de développement :
* Spring Tool suite 4
* Visual Studio code
* Java (Spring boot)
* Reactjs

### 2.1.1.3. Outils de développement

1. **Spring Tool suite 4**

Spring Tool Suite (STS) est un ensemble complet d'outils d'IDE gratuit et open source pour développer des applications Spring.

STS est basé sur Eclipse et comprend un ensemble d'outils et d'extensions conçus pour aider à développer des applications Spring plus rapidement et plus efficacement. Il prend en charge toutes les dernières fonctionnalités de la plate-forme Spring, y compris Spring Boot, Spring Cloud et Spring Security.

1. **Visual studio code**

Visual Studio Code (VS Code) est un éditeur de code gratuit et open source développée par Microsoft. Il est disponible pour Windows, MacOs et Linux. VS Code est un éditeur de code léger et puissant qui peut être utilisé pour développer des applications dans une variété de langages, notamment JavaScript, Type Script, C++, C#, Python, Java et Go.

1. **Spring boot**

Spring Boot est un Framework open source qui aide les développeurs à créer des applications Spring plus rapidement et plus facilement. Il fournit un ensemble de fonctionnalités et de conventions qui simplifient le développement d'applications Spring, telles que la configuration automatique, le démarrage en un clic et la prise en charge de la production.

1. **ReactJS**

Réacs est une bibliothèque JavaScript open source utilisée pour créer des interfaces utilisateur interactives. Elle a été créée en 2011 par Jordan Walke, un ingénieur logiciel chez Facebook. ReactJS est devenu très populaire au cours des dernières années et est maintenant utilisé par de nombreuses grandes entreprises, telles que Airbnb, Netflix et Uber.

1. **Base de données MySQL**

MySQL est un système de gestion de base de données relationnelles open source. Ce SGBDR d’Oracle est basé sur le langage SQL et fonctionne sur pratiquement toutes les plateformes comme Linux, UNIX et Windows.

MySQL est basé sur un modèle client-serveur, où le serveur MySQL gère toutes les instructions. Le but originel de ce SGBDR est d’administrer efficacement de grandes bases de données.

### 2.1.1.4. Architecture Logicielle

L’application utilise l’architecture MVC (Model View Controller). L'architecture MVC est un modèle d'architecture logicielle qui divise une application en trois parties distinctes : le modèle, la vue et le contrôleur.

**Le modèle** représente les données de l'application. Il est responsable de la gestion des données et de l'accès aux données.

**La vue** représente l'interface utilisateur de l'application. Elle est responsable de l'affichage des données à l’utilisateur.

**Le contrôleur** est responsable de la gestion de l'interaction entre le modèle et la vue. Il reçoit les entrées de l'utilisateur et les transmet au modèle. Il reçoit également les données du modèle et les transmet à la vue.

L'architecture MVC est un modèle d'architecture populaire pour les applications Web. Elle est conçue pour faciliter le développement, la maintenance et la mise à l'échelle des applications Web.

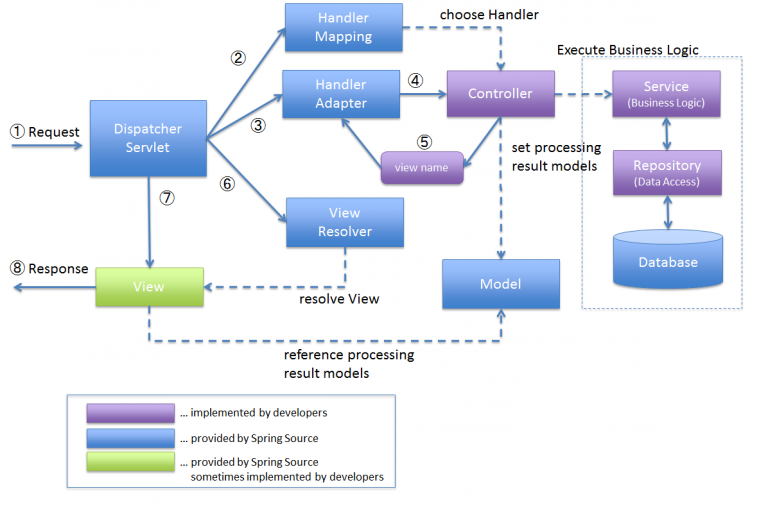


Figure 2‑1 Schéma de fonctionnement de l'architecture MVC dans une application Spring Boot

Le modèle représente les données de l'application. Il est responsable de la gestion des données et de l'accès aux données.

La vue représente l'interface utilisateur de l'application. Elle est responsable de l'affichage des données à l’utilisateur.

Le contrôleur est responsable de la gestion de l'interaction entre le modèle et la vue. Il reçoit les entrées de l'utilisateur et les transmet au modèle. Il reçoit également les données du modèle et les transmet à la vue.

## 2.2. Analyse de l’existant

Dans ce sous point, nous avons abordé la partie de l’application qui concerne notre travail, le paiement.

A ce jour, l’application à mise en place un système de paiement en utilisant l’api de Stripe.

Stripe est une API de paiement qui permet aux entreprises d'accepter des paiements en ligne. Elle est utilisée par des millions d'entreprises dans le monde entier, des petites entreprises aux grandes entreprises.

Ce système permet de payer en utilisant les cartes bancaires tels que VISA et MasterCard

### 2.2.1. Propositions de solutions

Comme évoqué dans la problématique, ce système de paiement ne s’adapter pas aux réalités de notre environnement d’où la nécessite de mettre en place un nouveau système.

Lorsque deux personnes ne se font pas confiance et doivent effectuer une transaction, ils font recourir à une personne tierce de confiance qui pourra agir comme intermédiaire. Nous allons nous positionner comme ce tiers personne de confiance

En pratique, lorsque le client voudra passer une commande, la somme payée sera versée dans notre compte en attendant que le vendeur puisse livrer la marchandise. Dès que le client aura bel et bien reçu sa marchandise, le montant de la commande sera verse au vendeur.

Le client aura la possibilité d’exiger un remboursement si la livraison ne s’effectue pas ou on ne lui livre pas la commande exacte.

De ce fait, le vendeur aura confiance que c’est un client sérieux d’une part et l’acheteur pourrait se rassurer que l’argent déboursé ne soit versé que lorsqu’il pourrait réceptionner sa commande en bonne et due forme.

En outre, le système de paiement prendra en compte les mobiles money[[5]](#footnote-5) (MPESA, …) parce que c’est un portefeuille que la grande majorité de la population africaine possède à la place des cartes bancaires dont la proportion des détenteurs est estimée à 12% en RDC[[6]](#footnote-6).

## 2.3. ETUDES DE FAISABILITES

Dans cette section, nous allons procéder à l’étude de faisabilité. L'étude de faisabilité est une étape importante dans le développement d'un projet. Elle permet d'évaluer si le projet est réalisable et rentable. L'étude de faisabilité prend en compte différents aspects, notamment :

* La faisabilité fonctionnelle : Est-ce que le projet est techniquement faisable ? Est-ce que les ressources nécessaires sont disponibles ?
* La faisabilité financière : Est-ce que le projet est rentable ? Est-ce que le projet est financé ?

L'étude de faisabilité doit être soigneusement réalisée afin d'évaluer la faisabilité globale du projet. Cela permettra de prendre des décisions éclairées quant à la poursuite ou non du projet, ainsi que d'identifier les ajustements éventuels à apporter pour maximiser ses chances de succès.

Nous allons soigneusement analyser les 2 parties de l’étude de faisabilité qui sont : La faisabilité opérationnelle et financière.

### 2.3.1. Faisabilité Opérationnelle

L’analyse de la faisabilité opérationnelle est l'évaluation de la possibilité de mettre en œuvre un projet ou un programme de manière efficace et efficiente. Elle consiste à analyser les ressources et les capacités nécessaires pour réaliser le projet, et à identifier les risques potentiels qui pourraient entraver sa mise en œuvre.

Pour réaliser cette analyse, nous allons utiliser la méthode PERT (Program Evaluation and Research Task). PERT est une méthode crée par les Américains à la fin des années 50 dans l’objectif de gagner du temps dans la réalisation de leurs projets phares. Il permet de déterminer le chemin critique : les taches pour lesquelles le moindre retard entraine un retard sur l’ensemble du projet.

### Identification et dénombrement des taches

Le tableau suivant comprend toutes les taches identifies pour la réalisation de ce projet ainsi que les contraintes de réalisation de chaque tache.

Tableau 2‑ identification des taches et durée des opérations du projet

|  |  |  |  |
| --- | --- | --- | --- |
| **Code Tache** | **Libelle Tache** | **Tache antérieure** | **Durée** |
| A | Définition des exigences | - | 7 |
| B | Rédaction du travail | A | 50 |
| C | Conception de l’architecture technique | A | 10 |
| D | Développement du backend | C | 14 |
| E | Intégrations des apis mobiles money | D | 5 |
| F | Tests unitaires | E | 5 |
| G | Documentation de l’API | F | 4 |
| H | Révision et correction | F | 5 |
| I | Déploiement de l’API | H | 2 |
| J | Test en production (Beta Test) | I | 5 |
| K | Validation du travail | B, J | 7 |
| TOTAL | |  | 144 |

### Réseau PERT

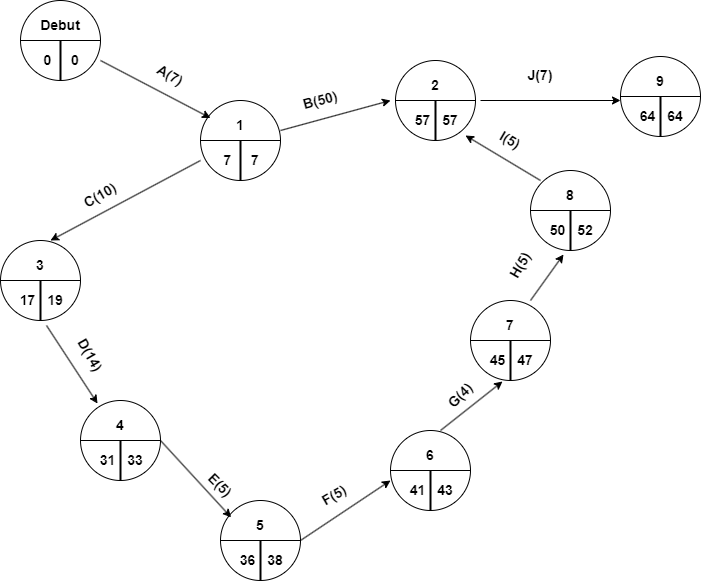


Figure 2-2 Réseau PERT

### Calcul de dates

* **Date au plus tôt (Dto)**

Le calcul des dates au plus tôt permet connaitre la date de début pour chaque tache afin de d’estimer aussi la date le plus tôt pour la fin du projet

Dto(début) = 0

Dto(1) = 0+7

Dto(2) = 7 + 50 et 50 + 5 (Maximum 57)

Dto(3) = 7+10 = 17

Dto(4) = 17+14 = 31

Dto(5) = 31+5 = 36

Dto(6) = 36+5 = 41

Dto(7) = 41+4=45

Dto(8) = 45+5 = 50

Dto(9) = 57+7 = 64

* **Date au plus tard (Dta)**

Le calcul des dates au plus tard permet connaitre la date de fin pour chaque tache afin de d’estimer aussi la date la plus tard pour la fin du projet

Dta(9) = 64

Dta(2) = 64-7=57

Dta(8) = 57-5 = 52

Dta(7) = 52-5 = 47

Dta(6) = 47-4 = 43

Dta(5) = 43-5 = 38

Dta(4) = 38-5 = 33

Dta(3) = 33-14 = 19

Dta(1) = 57 – 50 et 19 -10 (minimum 7)

Dta(début) = 7-7 = 0

### Calcul de marges

Les marges représentent le délai de mise en route d’une tache sans compromettre l’autre. Nous distinguons deux types de marges : Marge libre et marge totale

### Marge libre

Elle représente le délai de la mise en route de la tache(i) sans compromettre la date au plus tôt de l’étape(y).

ML(A) = 7-0-7=0

ML(B)=57-7-50=0

ML(C)=17-7-10=0

ML(D)=31-17-14=0

ML(E)=36-31-5=0

ML(F)=41-36-5=0

ML(G)=45-41-4=0

ML(H)=50-45-5=0

ML(I)=57-50-5=2

ML(J)=64-57-7=0

### Marge Total

Elle représente le délai pour la mise en route de la tache(i) sans modifier la date au plus tard(y).

MT(A) = 7-7-0=0

MT(B)=57-50-7=0

MT(C)=19-10-7=2

MT(D)=33-14-17=2

MT(E)=38-5-31=2

MT(F)=43-5-36=2

MT(G)=47-4-41=2

MT(H)=52-5-45=2

MT(I)=57-5-50=2

MT(J)=64-7-57=0

1. Détermination du chemin critique

Une tache critique est celle dont sa date de début au plus tôt correspond à sa date de début au plus tard.

Le chemin critique est déterminé en regroupant toutes les taches critiques

Dans le cas de notre travail, le chemin critique est : **A-B-K**

## 2.3.2. Faisabilité Financière

La mise en place de tout projet engage des dépenses. La faisabilité financière est une phase qui permet de déterminer tous les couts liés à la réalisation du projet. Dans cette section, nous avons dressé un tableau où nous répertorions d’une part l’ensemble des taches et leurs durées et d’autres les couts engagés journalièrement ou globalement.

Tableau ‑2 le cout de réalisation du projet par tache

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **N°** | **Taches** | **Durée en jours** | **Cout/Jour** | **Cout total** |
| 1 | Définition des exigences (Connexion internet) | 7 | 2$ | 14$ |
| 2 | Développement backend (Connexion internet) | 14 | 2$ | 28$ |
| 3 | Intégration des apis de mobile money (souscription + connexion internet) | 5 | 2 | 100$+10$ |
| 4 | Déploiement API (Hébergement) | 2 | 2 | 100$ + 4$ |
| 5 | Imprévus |  |  | 100$ |
| TOTAL | | 28 jours |  | 356$ |

## 2.4. CAPTURE DES BESOINS

La capture des besoins consiste à identifier les fonctionnalités majeures à développer dans le nouveau système et à décrire les spécifications techniques à respecter pour un meilleur développement.

Elle est subdivisée en deux parties : la capture des besoins fonctionnels ainsi que la capture des besoins techniques

### 2.4.1. Capture des besoins fonctionnels

La capture des besoins fonctionnels fait intervenir deux diagrammes essentiellement :

* Diagramme de séquences
* Diagramme de cas d’utilisation

Afin d’avoir une vue globale de l’interaction des acteurs et le système, nous allons aussi modéliser le diagramme de contexte

### 2.4.1.1. Enoncé sur le fonctionnement du nouveau système

Un marchand intéressé par l’instauration de notre système de paiement procèdera à l’inscription dans le site de l’entreprise. Après cette inscription, nous donnerons les procédures afin de lui permettre d’intégrer l’api dans son site e-commerce.

Lorsqu’un de ses clients veut procéder au paiement d’une commande, il soumet un formulaire qui comprend son numéro de compte ainsi que l’opérateur de mobile money désiré. Le système procède à la vérification du solde auprès de cette operateur. Une fois la transaction valide, ce paiement est transmis temporairement dans un compte transit. Apres réception de sa livraison, un client peut soit refuse la livraison et demande un rembourser ou accepter et valider définitivement sa transaction. Si le client accepte sa livraison, le système effectue un transfert du compte de transit vers le compte du marchand et envoie une commission auprès de l’administrateur. Au cas de refus, le système effectue un transfert du compte de transit vers le compte du client. Tout en notifiant à chaque étape du processus, toutes les parties prenantes.

1. **Diagramme de contexte**

Le diagramme de contexte est un diagramme de niveau supérieur qui visualise les interactions entre un système et ses acteurs externes. Il est utilisé pour fournir une vue d'ensemble du système et de ses interactions avec son environnement.

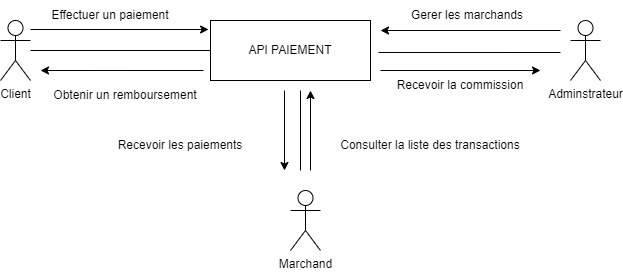


Figure 2‑3 Diagramme de contexte de l'API

1. **Diagramme de cas d’utilisation globale**

Le diagramme de cas d’utilisation est un diagramme orienté utilisateurs. Il permet de recueillir, d’analyser et d’organiser les besoins, de recenser les grandes fonctionnalités du système à mettre en place. Il est utilisé pour capturer les exigences fonctionnelles d'un système et pour communiquer les exigences aux parties prenantes.

**Acteurs**

* **Client** : représente un possible acheté dans un site e-commerce qui voudrait procéder au paiement de sa commande.
* **Marchand** : Représente un propriétaire d’un e-commerce, qui aimerait intégrer notre système de paiement dans sa plateforme.
* **Administrateur** : Représente l’entreprise qui gère ce système de paiement.

**Cas d’utilisations**

* Payer commande
* Demander remboursement
* Confirmer commande
* Recevoir paiement
* Voir liste transactions
* Voir état transaction
* Recevoir commission
* S’authentifier

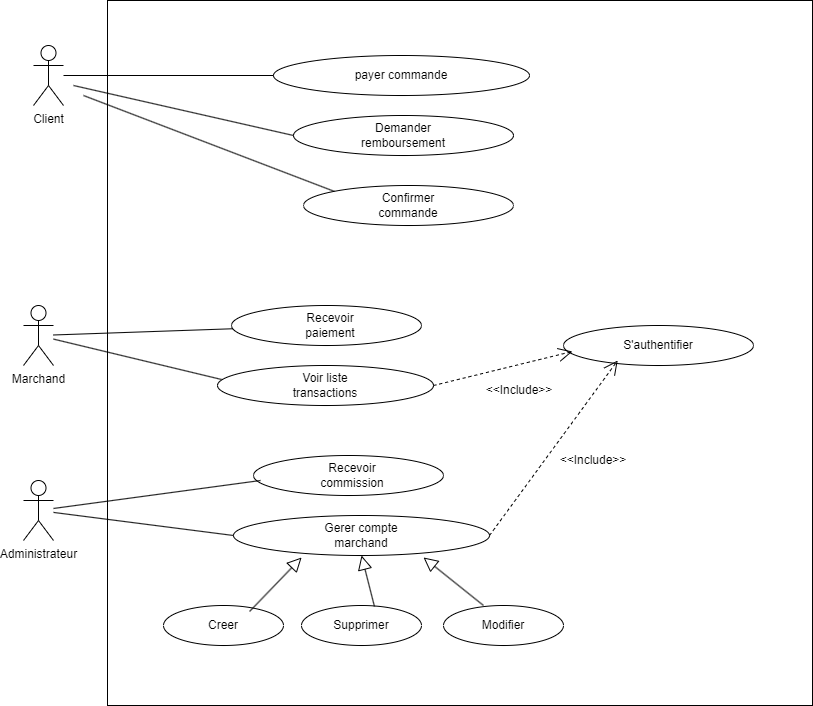


Figure 2‑4 Diagramme de cas d'utilisation globale de l'API

1. **Diagramme de séquence**

Un diagramme de séquence représente les interactions entre les objets d'un système en fonction du temps. Il est utilisé pour visualiser les interactions entre les objets et pour comprendre comment les objets communiquent entre eux.

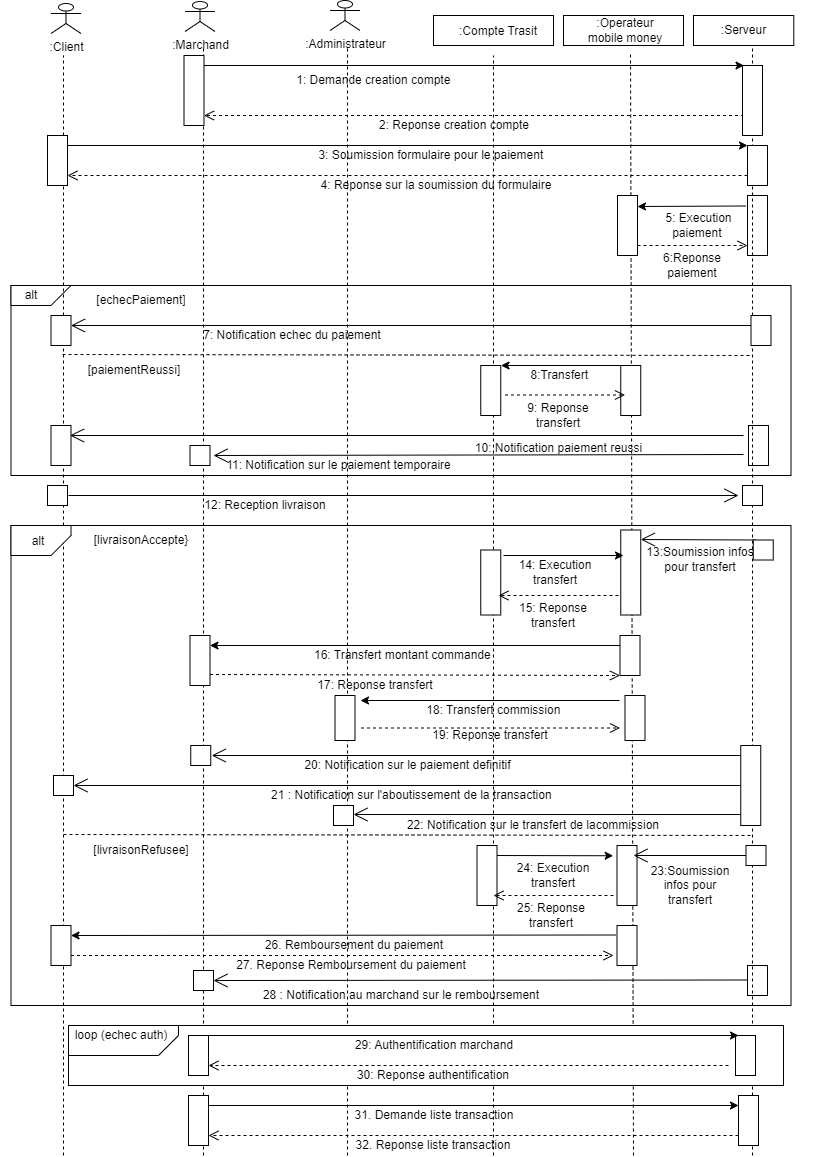


Figure 2‑5 Diagramme de séquence de l'API

## 2.4.2. Captures des besoins techniques

* **Environnement du système**

La particularité du secteur e-commerce réside du fait que toutes les transactions se font en ligne grâce à l’internet. Pour ce fait notre api sera accessible par tous les systèmes connectes à internet car cela pourrait permettre la communication et la réalisation des transactions en temps réel.

Donc l’api sera dans un serveur et les autres applications pourraient à partir de leurs serveurs communique avec celle-ci.

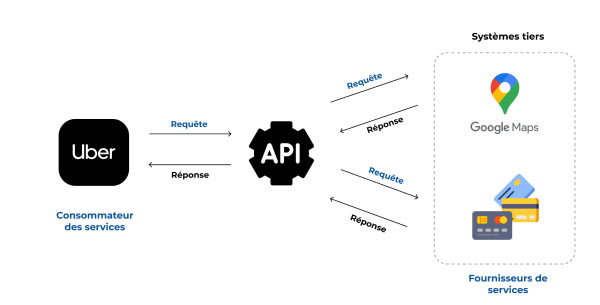


Figure 2‑6 illustrations d'une application interagissant avec les API des tiers

* **Style d’architecture de déploiement**

Pour la réalisation de notre api, nous avons choisi l’architecture a 3 niveaux (appelée architecture 3-tier).

L’architecture 3-tiers est une architecture logicielle qui divise une application en trois couches logiques :

* **La couche de présentation** : Cette couche est responsable de l'affichage de l'application à l'utilisateur. Elle se compose généralement d'un navigateur Web ou d'une application mobile.
* **La couche d'application** : Cette couche est responsable du traitement des demandes de l'utilisateur et du retour des résultats. Elle se compose généralement d’un ou plusieurs serveurs Web ou d'applications.
* **La couche de données** : Cette couche est responsable du stockage des données de l'application. Elle se compose généralement d'une base de données ou d'un entrepôt de données.

Nous avons choisi cette architecture pour la réalisation de notre api car la séparation de la couche d’application et de données permet une meilleure sécurité en effectuant une gestion des accès non autorisé efficace.

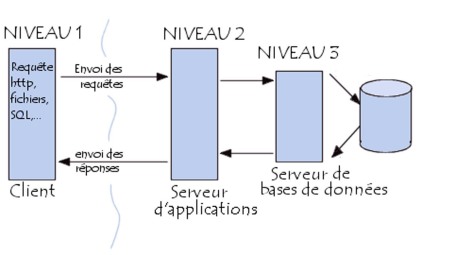


Figure 2‑7 illustrations de l'architecture à trois niveaux(3-tiers)

## 2.5. Elaboration

La phase d’élaboration reprend les éléments de la phase de la capture des besoins et les précise pour chaque acteur afin d’arriver à une spécification détaille de la solution à mettre en œuvre.

Pour faciliter la maitrise des fonctionnalités lors de la capture des besoins, Le diagramme de cas d’utilisation globale sera découpes en sous-systèmes appelles packages.

Cette découpe en sous-systèmes se fera en fonction des acteurs. Pour chaque acteur, nous aurons ses cas d’utilisations ainsi que la partie du diagramme de séquence ou il joue un rôle.

### 2.5.1. Package Client

* **Cas d’utilisation**

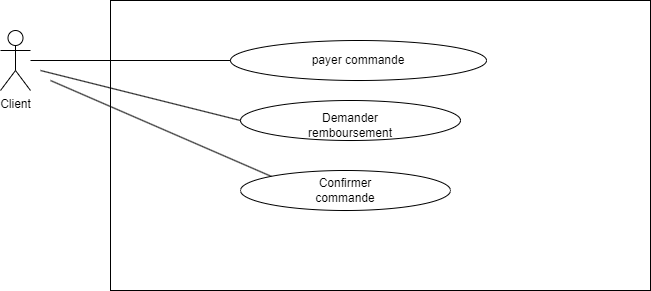


Figure 2‑8 Package de cas d'utilisation Client

Tableau 2‑3 Description textuelle du Package Client

|  |  |
| --- | --- |
| Acteur | Client |
| Cas d’utilisations | 1. Payer commande  2. Demander remboursement  3. Confirmer commande |
| Objectifs | 1. Procéder au paiement des commandes du client  2. Demander un remboursement lorsqu’il n’est pas satisfait de la livraison  3. Confirmer son paiement lorsqu’il est satisfait de sa livraison |
| Précondition | 1. Passer la commande auprès d’un marchand  2. Avoir effectué un payment et avoir un numéro de mobile money agrée.[[7]](#footnote-7)  3. Avoir effectué un payment |
| Scenario nominal | 1. Le client d’un site e-commerce, après avoir passé une commande voudrait procéder au paiement pour se faire livrer  2. Apres réception de sa commande, si le client n’est pas satisfait, il peut demander un remboursement  3. Apres réception de sa commande, si le client est satisfait, il peut confirmer son paiement pour que l’argent puisse être verser au marchand. |
| Scenario alternatif | 1. Si les informations entres sur le formulaire sont incorrectes, le système le notifie afin de le guider à corriger  2. Vérifier d’avoir eu la notification que le paiement a été accepté où rafraichir la page  3. Vérifier d’avoir eu la notification que le paiement a été accepté où rafraichir la page |

### 2.5.2. Package Marchand

* **Cas d’utilisation**

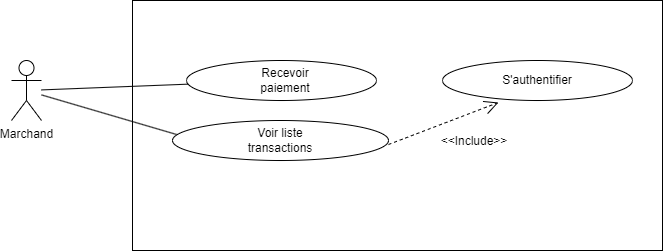


Figure 2‑9 Package de cas d'utilisations Marchand

Tableau 2-4 Fiche de description du package Marchand

|  |  |
| --- | --- |
| Acteur | Marchand |
| Cas d’utilisations | 1. Recevoir paiement  2. Voir liste transactions  3. S’authentifier |
| Objectifs | 1. Permettre au Marchand de recevoir le paiement de ses clients  2. Avoir la liste complète des paiements qu’il a reçus  3. Être identifier dans le système afin d’avoir les informations précises liées à son compte |
| Précondition | 1. Avoir un compte dans le système et intégrer le formulaire de paiement connecté à l’api dans sa plateforme et posséder un numéro de mobile money agrée comme marchand par ce mobile money.  2. S’authentifier  3. Avoir créer un compte |
| Scenario normal | 1. Apres qu’un client ait confirmer son paiement définitif, le marchand est notifié pour un paiement sur son compte  2. Le marchand se connecte sur son compte et affiche la liste de transactions  3. Dès qu’il a un compte valide, le marchand s’authentifier pour accéder à son tableau de bord |
| Scenario alternatif | 1. Vérifier que le formulaire de paiement se retrouve bien dans la partie réserve au paiement de commande dans son site e-commerce  2. Rafraichir la page dans la section réserves aux transactions  3. Vérifier que les identifiants entres sont corrects |

### 2.5.3. Package Administrateur

* **Cas utilisation**

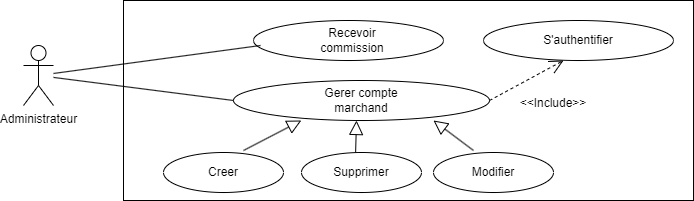


Figure 2‑10 Fiche de description du package Marchand

Tableau 2‑5 Fiche de description du package Administrateur

|  |  |
| --- | --- |
| Acteur | Administrateur |
| Cas d’utilisations | 1. Recevoir commission  2. Gérer compte marchand  3. S’authentifier |
| Objectifs | 1. Recevoir la commission sur toutes les transactions effectuées dans sa plateforme  2. Gérer tous les marchands qui veulent utiliser ce système de paiement.  3. Ne permettre qu’a ceux qui sont autorisées à effectuer ces taches si délicates |
| Précondition | 1. Avoir défini un compte qui va réservoir le paiement  2. S’authentifier  3. Avoir les identités de connexion valide |
| Scenario normal | 1. Pour chaque paiement confirmé, une certaine somme calcule en pourcentage sous forme de frais de service ou commission est verse à l’administrateur  2. L’administrateur se connecte pour voir la liste des tous les marchands et bannir ceux qui ne respectent pas les règles ou CGU  3. L’administrateur se connecte avec les identifiants fournis par le système afin d’effectuer son travail |
| Scenario alternatif | 1. Vérifier que le moyen de paiement défini est valide ou correct  2. Vérifier la connexion internet et rafraichir la page.  3. Vérifie que les bons identifiants ont été entrés pour la connexion. |

# CONSTRUCTION ET TRANSITION

## CONSTRUCTION DU SYSTEME

La phase de la construction du système est une étape très importante et complexe. Cette phase comprend les étapes ci-après :

* La modélisation statique
* La modélisation dynamique
* La présentation d’environnement de développement et du SGBD
* La codification
* Le test
* La présentation des interfaces et structures des tables

### Modélisation statique

La modélisation statique est une technique utilisée pour représenter la structure d'un système logiciel. Elle permet de visualiser les éléments d'un système, tels que les classes, les objets, les interfaces et les relations entre ces éléments.

1. **Diagramme de classes**

Tableau 3‑ Règles de gestion

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| N° | Association | Classe d’objet | | Multiplicité | | Règles de gestion |
| Source | Cible | Source | Cible |
| 1 | Posséder | Marchand | Moyen paiement | 1 | 1.. \* | Un marchand peut posséder des moyens de paiement |
| 2 | Gérer | Administrateur | Marchand | 1 | 1.. \* | Un administrateur peut gérer des marchands |
| 3 | Effectuer | Client | Transaction | 1 | 1.. \* | Un client peut effectuer des transactions |

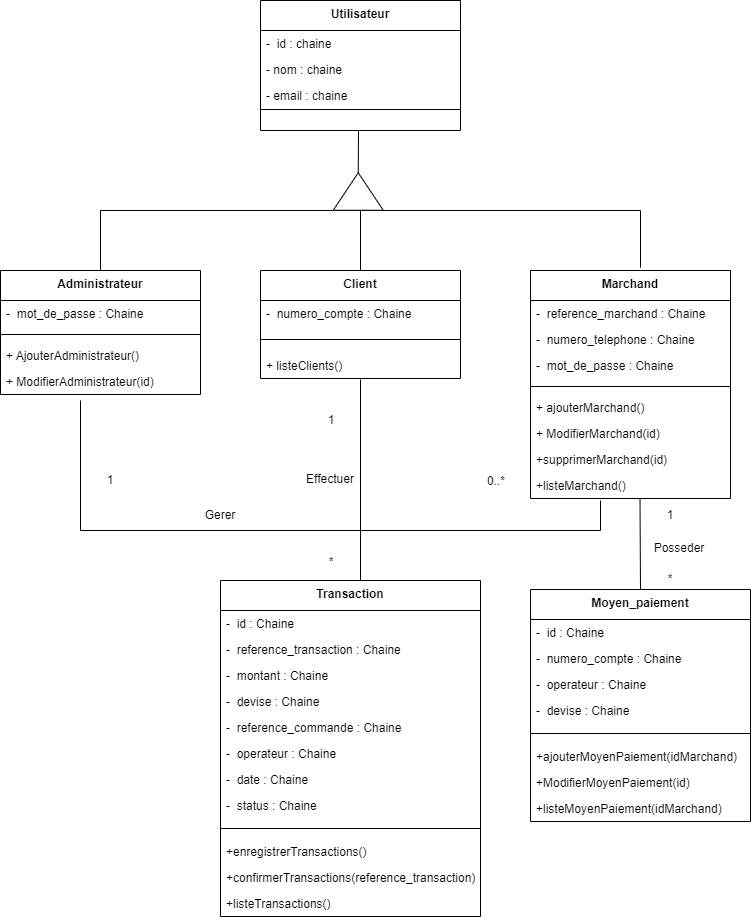


Figure 3-1 Présentation du Diagramme de classe

1. **Diagramme des composants**

Un diagramme de composants représente les différents composants d'un système logiciel et leurs relations. Il est utilisé pour visualiser la structure du système logiciel et pour comprendre comment les différents composants communiquent entre eux.

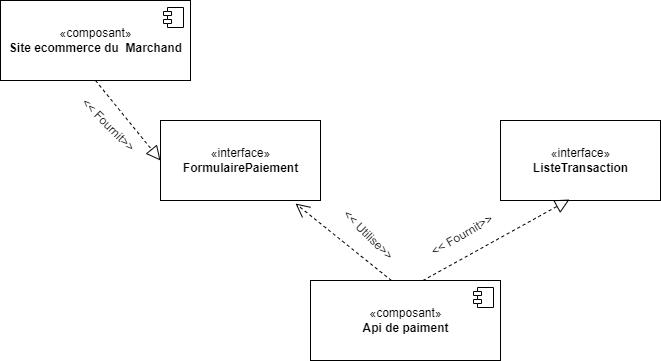


Figure 3-2 Présentation du diagramme de composants

### Modélisation dynamique

La modélisation dynamique est utilisée pour représenter le comportement d'un système au fil du temps. Elle permet de visualiser les interactions entre les différents éléments d'un système et de comprendre comment le système évolue en fonction des entrées.

Dans cette section, nous allons présenter deux diagrammes importants tels que :

* Diagramme d’activités et
* Diagramme d’état-transition

1. **Diagramme d’activités**

Un diagramme d'activités représente les différentes activités d'un système et les relations entre ces activités. Il est utilisé pour visualiser le flux de travail d'un système et pour comprendre comment les différentes activités sont liées entre elles.

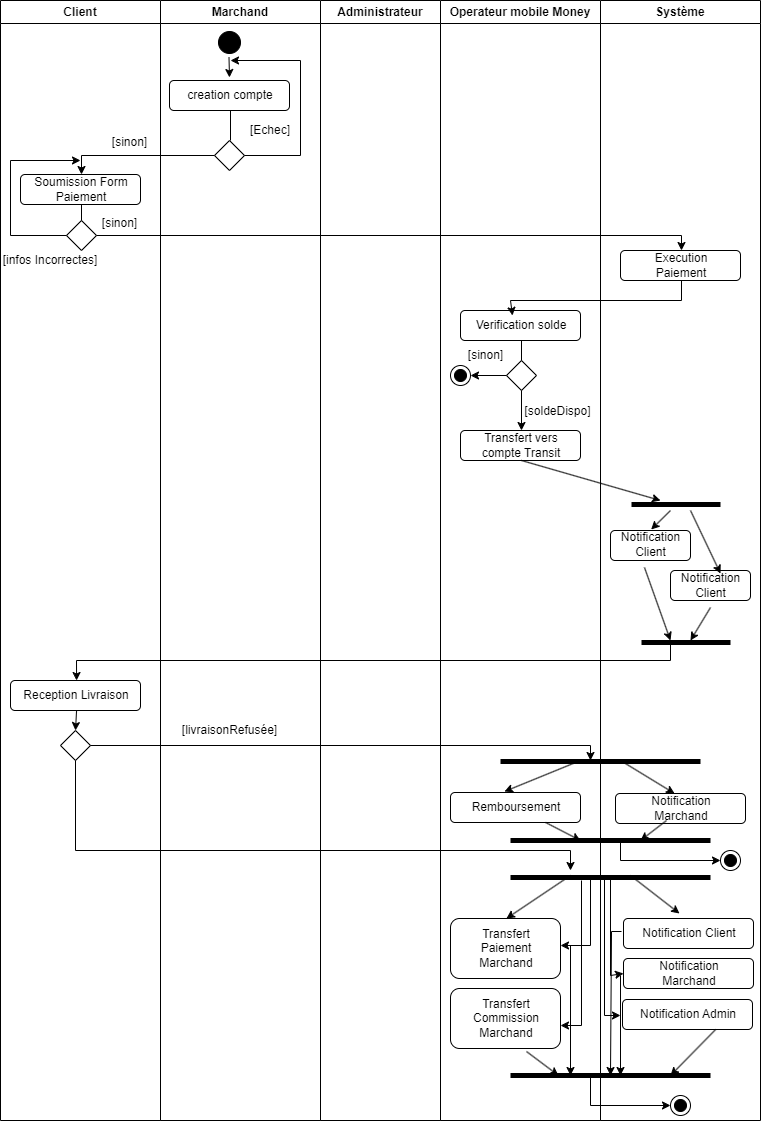


Figure 3-3 Présentation du diagramme d'activités

1. **Diagramme d’état-transition**

Un diagramme d'état-transition représente les différents états d'un objet et les transitions entre ces états. Il est utilisé pour visualiser le comportement d'un objet au fil du temps et pour comprendre comment l'objet réagit aux événements.

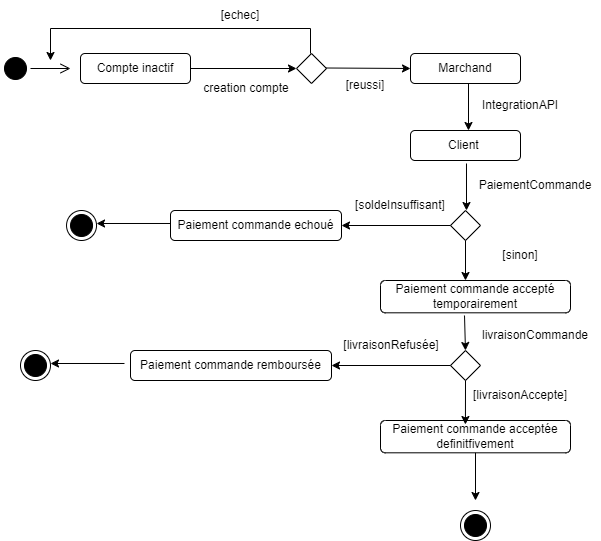


Figure 3-4 Présentation du diagramme d'état-transition

## Implémentation du système

### Besoins techniques

* + - 1. **Outils de développement**
* **Choix du langage de programmation**

Pour la réalisation de notre api, nous avons utilisé le langage java avec son Framework Spring boot.

Le Framework Spring Boot a été choisi pour plusieurs raisons entre autres sa maturité. On peut clairement constater sur le marché que grand nombre des entreprises utilisent ce Framework pour développer un système stable et complexe. En outre, avec sa documentation bien détaillé, nous a permis de rapidement développer nos fonctionnalités.

Afin d’implémenter la partie administrateur, nous avons utilisé la bibliothèque javascript ReactJS.

Pour ajouter assurer la sécurité de notre API, nous avons utilisé Spring Security.

Spring Security est un Framework de sécurité open source pour Java qui fournit une API simple et flexible pour sécuriser les applications. Il est basé sur le modèle de sécurité de la défense en profondeur, qui consiste à utiliser plusieurs couches de sécurité pour protéger les applications contre les attaques.

Afin de garantir la sécurité de ses applications, Spring Security fournit les fonctionnalités telles que : Authentification, l’autorisation, le chiffrement grâce aux jetons JWT( *JSON Web Token*).

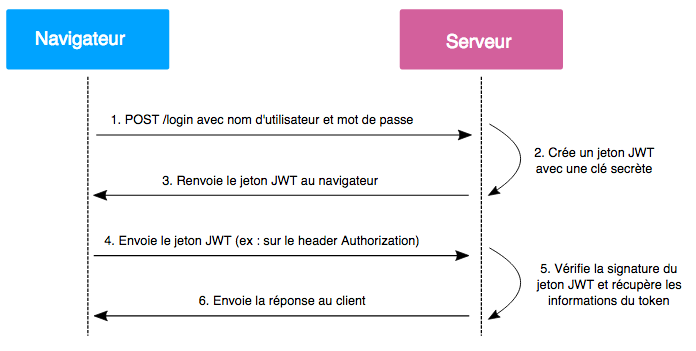


Figure 3‑‑. Utilisation du JWT lors de l'authentification

* **Environnement de développement**

Nous avons utilisé deux environnements de développement puissant afin de développer notre api qui sont : Spring Tool suite 4 et Visual studio Code.

En outre, la plateforme Postman, nous a servi à tester notre API pendant le développement.

* **Choix du SGBD**

Pour la gestion de notre base de données, nous avons choisi le SGBD MySQL pour sa simplicité et sa facilité d’accès et aussi parce qu’il offre une offre gratuite.

* **Architecture Logicielle**

Notre api suit l’architecture par défaut de Spring boot appelle le « *flow architecture* » dans laquelle l’application est subdivisé en plusieurs couches telles que :

Spring Boot suit une architecture en couches dans laquelle chaque couche communique avec la couche directement au-dessous ou au-dessus (structure hiérarchique) de celle-ci.

Avant de comprendre l'architecture Spring Boot, nous devons connaître les différentes couches et classes qui y sont présentes. Il existe quatre couches dans Spring Boot :

* Couche de présentation
* Couche métier
* Couche de persistance
* Couche de base de données

1. **Couche de présentation**

La couche de présentation gère les requêtes HTTP, traduit les paramètres JSON en objets, authentifie la requête et la transmet à la couche métier. En bref, elle consiste en des vues, c'est-à-dire la partie frontale.

1. **Couche métier**

La couche métier gère toute la logique métier. Elle est composée de classes de service et utilise les services fournis par les couches d'accès aux données. Elle effectue également l'authentification et la validation.

1. **Couche de persistance**

La couche de persistance contient toute la logique de stockage et traduit les objets métier des lignes de base de données.

1. **Couche de base de données**

Dans la couche de base de données, les opérations CRUD (créer, lire, mettre à jour, supprimer) sont effectuées.

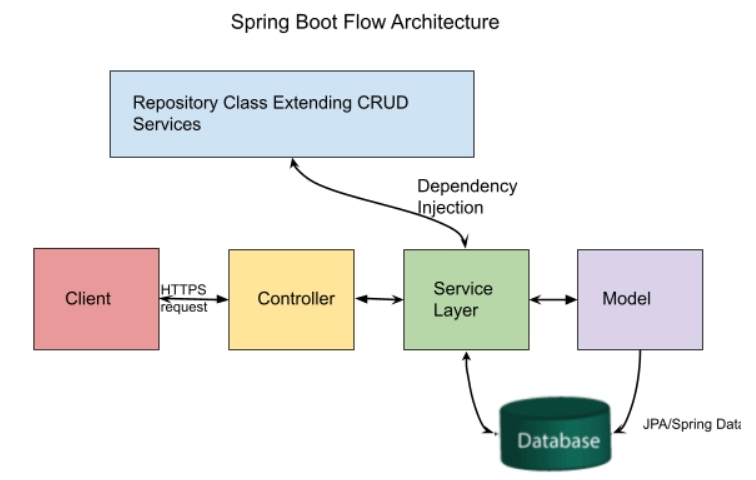
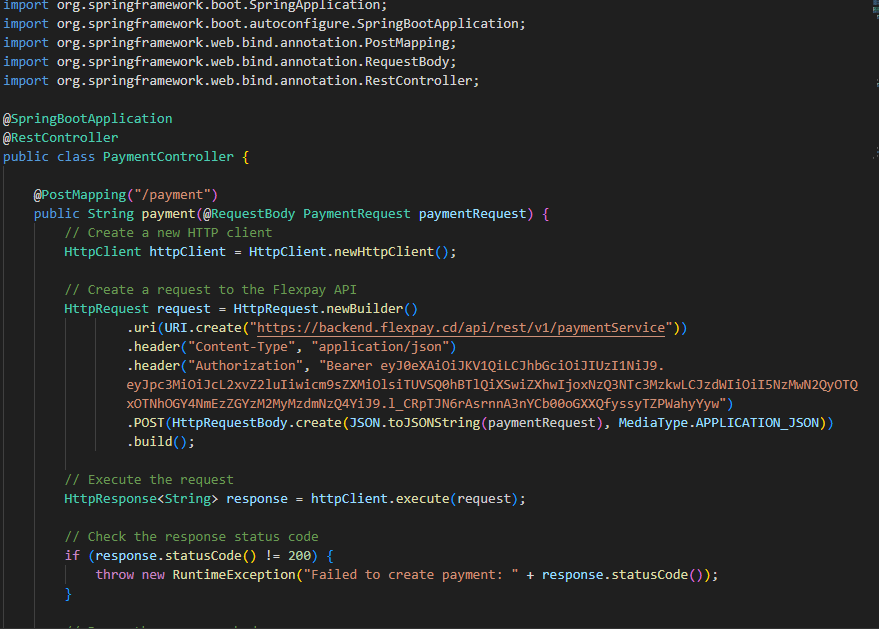
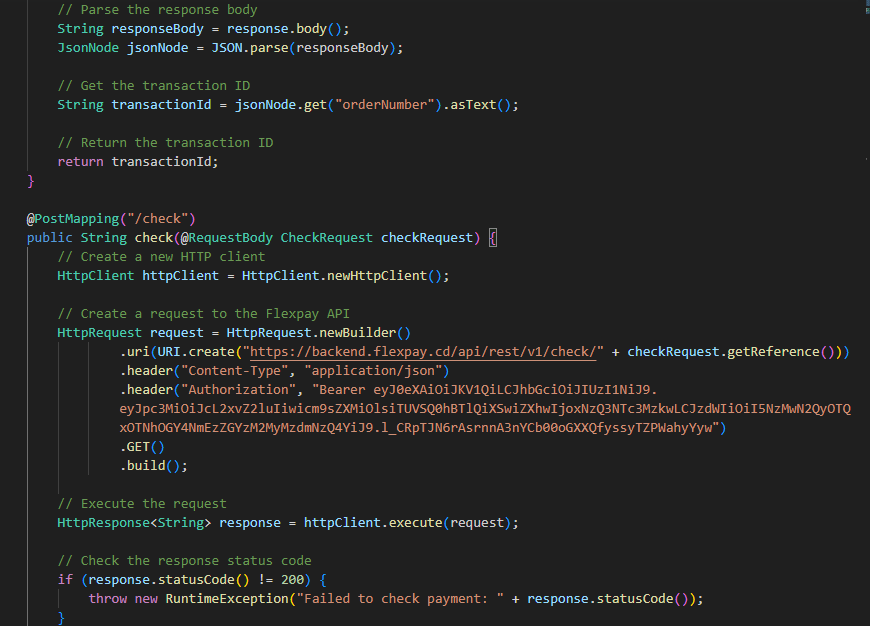


Figure 3-6 Description du flow architecture

## Présentation des codes sources

Cette section nous permettra de présenter les codes sources des fonctionnalités clés de notre application.

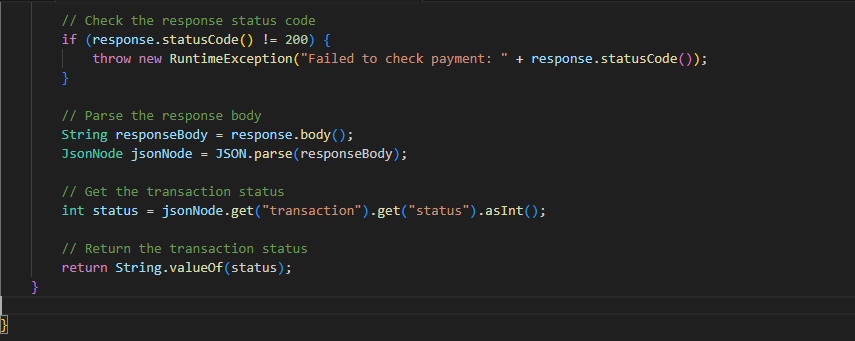


Figure 3‑‑. Codes sources du Controller de Paiement

## Présentation d’interfaces

Cette section nous permettra de présenter les principales interfaces de notre api ou la manière dont notre API agit auprès des clients.

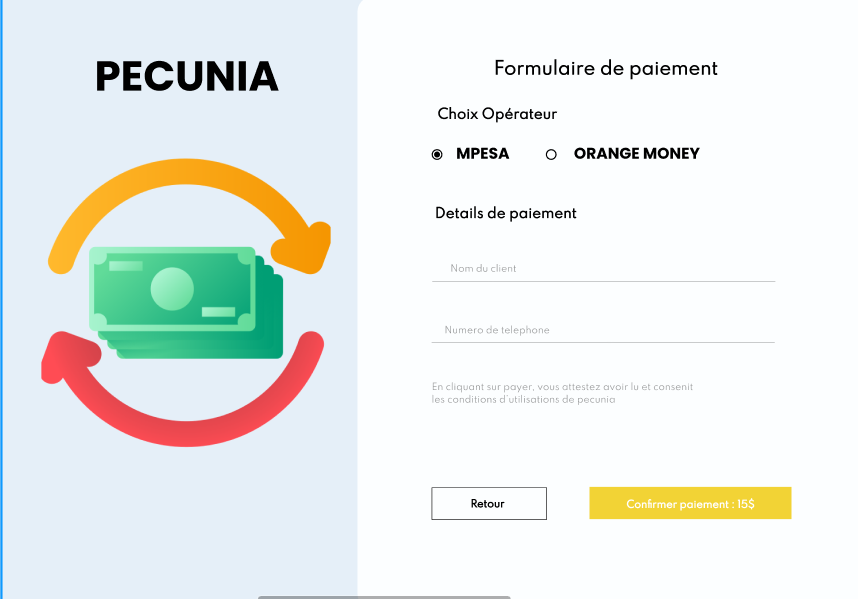


Figure 3‑‑.Interface de paiement pour le client

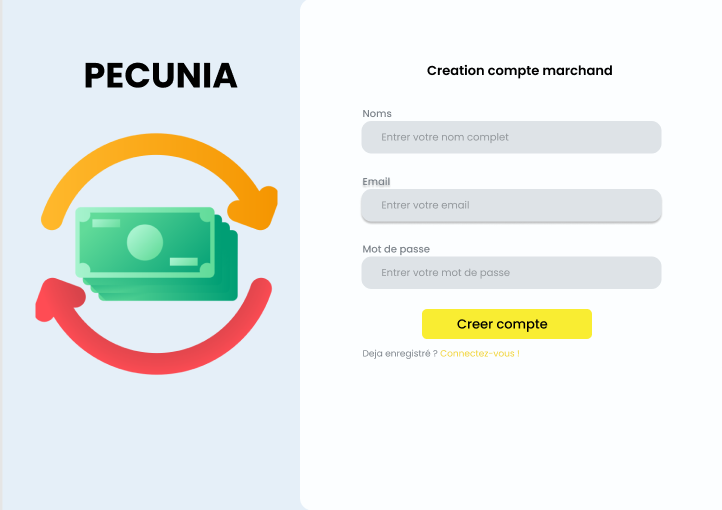


Figure 3‑‑. Interface de création de compte marchand

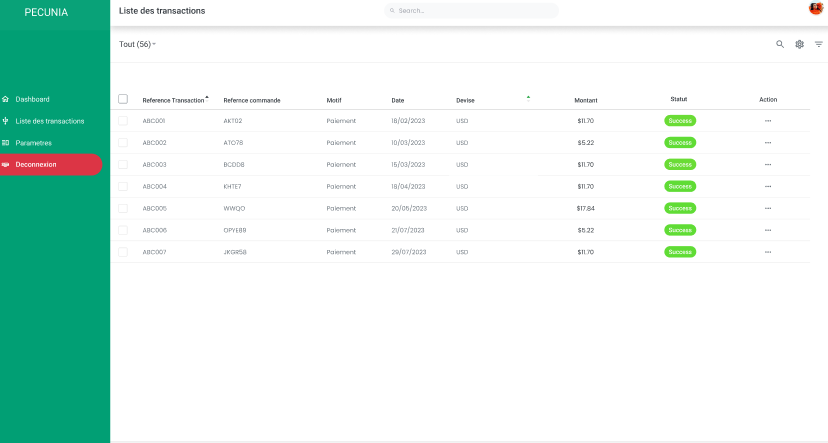


Figure 3‑‑. Interface liste des transactions pour le marchand

## TRANSITION VERS LE NOUVEAU SYSTEME

Cette dernière phase, est important car elle permet de démontrer comment le nouveau système sera mise en place. Pour ce faire, nous allons utiliser le diagramme de déploiement

**Diagramme de déploiement**

Un diagramme de déploiement représente la distribution des composants d'un système sur différentes machines. Il est utilisé pour visualiser l'architecture physique d'un système et pour comprendre comment les différents composants communiquent entre eux.

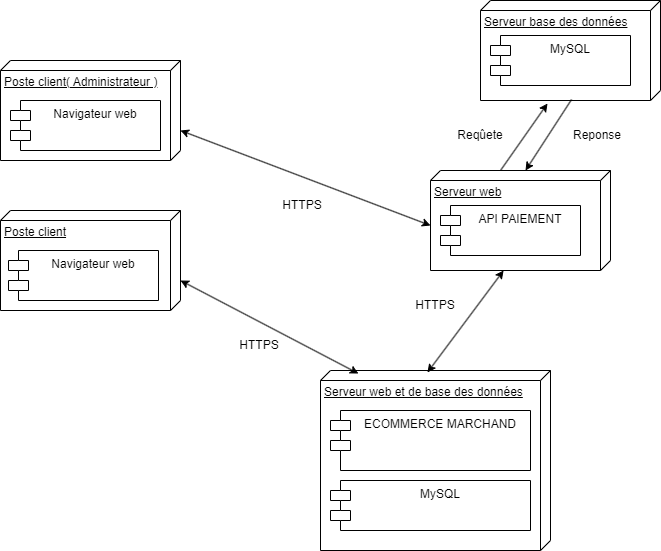


Figure 3-7 Présentation du diagramme de déploiement

# CONCLUSION

Le paiement a toujours été au cœur du quotidien de l’homme. Il lui permet d’obtenir ce qu’il veut, en échange d’argent. Auparavant, les paiements se faisaient en espèces. Cependant, avec le développement du commerce électronique, les systèmes de paiement ont dû s’adapter.

Nous avons exploré les raisons pour lesquelles il est nécessaire d’améliorer nos systèmes de paiement. Nous avons constaté que le commerce électronique ne cesse de se développer et que les systèmes de paiement traditionnels ne sont pas toujours adaptés à ce nouveau mode de consommation.

Nous avons donc développé une nouvelle approche de paiement basée sur les API. Cette technologie permet de dématérialiser l’ensemble de l’infrastructure de paiement. Cela rend les paiements plus accessibles et plus faciles à utiliser, quel que soit l’endroit où se trouve l’acheteur ou le vendeur.

L’un des défis du commerce électronique est la confiance. Les acheteurs doivent pouvoir se fier aux vendeurs et les vendeurs doivent pouvoir se fier aux acheteurs. Nous avons pensé à une solution qui pourrait à la fois faciliter le paiement et augmenter la confiance entre les deux parties.

Notre API de paiement a été conçue en respectant le principe juridique d’entiercement. Ce principe permet d’ajouter une tierce personne de confiance pendant le processus du paiement. Le client peut payer via cette personne de confiance et est assuré que le paiement ne sera pas transféré au vendeur tant que celui-ci n’aura pas confirmé la commande. Ce paiement en avance permet de mettre le vendeur en confiance que c’est une commande sérieuse.

Nous croyons que le commerce électronique est un secteur à fort potentiel. Nous pensons qu’avec la mise en place de notre solution, l’Afrique, où le défi de la confiance est fort, pourrait enfin profiter de ce secteur.

Comme tout œuvre humaine étant imparfaite, nous sommes disposés à recevoir des remarques et critiques lies à ce travail et différentes recommandations en vue de son amélioration.

# BIBLIOGRAPHIE

Alain DUCASS, J.-M. K. (2015). Le commerce électronique en Afrique, Recommandation pour l'intégration en Méditerrané. Paris: IPEMED.

API de paiement en ligne pour les développeurs. (n.d.). Square. Récupéré de https://developer.squareup.com/fr/fr/online-payment-apis

Conception Et Réalisation D’une Application Web E-Commerce Sous Java EE | PDF | Commerce électronique | Développement de logiciel. (n.d.). Scribd. Récupéré de https://fr.scribd.com/document/472533665/01-Conception-et-realisation-d-une-application-Web-E-Commerce-sous-Java-EE

DEBRAUWER, L. (2016). UML 2.5 Initiation, exemples et exercices corrigés. Paris: ENI.

E-Commerce en Afrique francophone : le paiement, la “première difficulté” pour les entrepreneurs du digital - BBC News Afrique. (n.d.). BBC News Afrique. Récupéré de https://www.bbc.com/afrique/region-57460615

Ethan C. (Fevrier,2022). Global Ecommerce Forecast & Growth Projections, Insider Intelligence. Récupéré https://www.insiderintelligence.com/content/global-ecommerce-forecast-2022.

G. (2020, Février 16). Qu’est-ce qu’une API et pourquoi vous devez en concevoir ? | Apprendre la programmation. Apprendre La Programmation. Récupéré de https://apprendre-la-programmation.net/definition-api/

H. (2015, septembre 15). Petite histoire des moyens de paiements. BNP Paribas. https://histoire.bnpparibas/troc-argent-liquide-sans-contact-petite-histoire-des-moyens-de-paiements/

Jensen, C. T. (2015). Les API POUR LES NULS. New York: John Wiley & Sons. Inc.

MAMPUYA KINKANI Pescie H., P. J. (2023). E-commerce : Guide pour sa réussite. Kinshasa: Editions de l'Université Protestante au Congo.

MAMPUYA KINKANI PESCIE HONORE, M. F. (2023). Modélisation Objet avec le processus Unifié et UML. Kinshasa: Editions de l'Université Protestante au Congo.

Modor Intelligence. (2023). Digital payments market size& share analysis-growth trends & Forecasts (2023-2028). Récupéré de https://www.mordorintelligence.com/industry-reports/digital-payments-market.

Schaullum Pierre, O. I. (2021). Les systèmes de paiement mobile a l'ère de la covid-19 : sécurité, vie privée et confiance numérique. London: ISTE OpenScience.

TOSSOU, O. N. (2017). Développement d’une API Java de paiement en ligne sur les sites e-commerce. Cotonou: UNIVERSITÉ D’ABOMEY-CALAV.

S. (n.d.). Pourquoi les APIs sont essentielles à l’essor de l’Embedded Finance. Skaleet. Récupéré de https://skaleet.com/blog/pourquoi-les-apis-sont-essentielles-a-l-essor-de-l-embedded-finance/

1. Modor Intelligence. (2023). Digital payments market size& share analysis-growth trends & Forecasts (2023-2028). Récupéré de https://www.mordorintelligence.com/industry-reports/digital-payments-market. [↑](#footnote-ref-1)
2. Ethan C. (Fevrier,2022). Global Ecommerce Forecast & Growth Projections, Insider Intelligence. Récupéré https://www.insiderintelligence.com/content/global-ecommerce-forecast-2022. [↑](#footnote-ref-2)
3. Alain DUCASS, J.-M K (2015), *Le commerce électronique en Afrique*. Paris : Institut de prospective économique du monde méditerranéen [PDF]. Récupéré de https://www.ipemed.coop/adminIpemed/media/fich\_article/1460888627\_ipemede-commerce-en-afriquebd.pdf [↑](#footnote-ref-3)
4. MPESA : M-Pesa est un produit de microfinance et de transfert d'argent lancé par Vodacom en 2012 permettant aux abonnés d'accéder à une large gamme de services financiers à partir de leur propre téléphone mobile [↑](#footnote-ref-4)
5. Mobile Money : Le mobile money est un service financier qui permet aux utilisateurs de stocker, d'envoyer et de recevoir de l'argent en utilisant leurs téléphones portables [↑](#footnote-ref-5)
6. Target RDC. Banque : 12% des congolais vivant en milieux urbains possèdent un compte, selon une étude Target. Récupère de https://www.target-sarl.cd/fr/content/banque-12-des-congolais-vivant-en-milieux-urbains-possedent-un-compte-selon-une-etude-target [↑](#footnote-ref-6)
7. A la période où nous rédigeons ce travail, seul le numéro de téléphone qui est agrée comme marchand peut recevoir un remboursement ou un paiement, c’est l’unes des restrictions imposées par les entreprises de mobile money afin de se protéger contre le risque du blanchissement d’argent. [↑](#footnote-ref-7)