

IIR

INGÉNIERIE INFORMATIQUE & RÉSEAUX

ANNÉE UNIVERSITAIRE

2025/2026

OPTION

**MÉTHODES INFORMATIQUES APPLIQUÉES
À LA GESTION DES ENTREPRISES (MIAGE)**

PROJET DE FIN D'ANNÉE

THÈME

Aether Wallet Multi device wallet blockchaine

RÉALISÉ PAR

**Salaheddine MCHICH
Ayoub Ouaha
Mohamed Fatihe
Amal famous
Hajar OU-Raho**

ENCADRÉ PAR

SOUTENU LE

À nos chers parents

*Que ce modeste travail soit l'aboutissement
de vos vœux tant formulés, le fruit de vos
innombrables sacrifices, bien que nous ne
vous en acquitterons jamais assez.*

À mes chers amis

*À mes amis pour leur sympathie, fidélité
et soutien.*

À nos chers professeurs

*Ce projet n'aurait probablement pas la
forme actuelle sans l'aide de nos professeurs.*

Remerciements

Je tiens tout d'abord à remercier tout le corps enseignant de mon établissement, qui m'a accompagné tout au long de ma formation académique.

Je souhaiterais remercier particulièrement toutes les personnes qui m'ont épaulé et conseillé durant la réalisation de ce projet, et qui m'ont transmis leur expertise dans le domaine du développement mobile et de la blockchain.

Ce projet de fin d'études m'a permis d'acquérir des compétences précieuses dans le développement d'applications mobiles cross-platform, l'intégration blockchain, et l'architecture logicielle moderne.

Je n'oublie pas non plus mes proches qui m'ont sans cesse soutenu dans l'élaboration de ce projet et m'ont aidé à chaque étape de sa réalisation.

Abstract

*As part of my engineering training, I completed this end-of-studies project focused on the design and implementation of **Aether Wallet**, a production-grade cryptocurrency wallet application developed using Flutter for cross-platform mobile deployment.*

*The functional module I developed provides a comprehensive solution for managing digital assets with a sophisticated **Deep Glassmorphism** user interface. More specifically, I worked on the integration of real-time blockchain interactions with the Ethereum Mainnet. The application features critical modules including wallet generation using BIP39 mnemonic phrases, secure transaction signing, and live market data visualization via WebSocket connections. The system also incorporates a holographic-styled QR scanner and an AI-powered chat assistant to guide users through the complexities of the crypto ecosystem.*

*My work ensures that the management of cryptocurrency holdings is carried out in a secure, immersive, and performant manner, guaranteeing full data persistence via PostgreSQL and high-speed caching with Redis. The architecture follows **Clean Architecture** principles combined with the **BLoC** state management pattern to ensure scalability and maintainability.*

The success of this project relies on adopting a development process that adheres to the Agile methodology, and for the architectural design, I utilized UML modeling. To succeed in this project, I applied the expertise acquired during my academic formation in mobile development and blockchain technology.

The technical environment in which I worked included Flutter/Dart, Node.js, Express, TypeScript, PostgreSQL, Redis, Web3, and Angular for the administration dashboard.

Keywords : Cryptocurrency Wallet, Flutter, Ethereum, Blockchain, Clean Architecture, BLoC, Deep Glassmorphism, Web3, Mobile Development

Résumé

*Dans le cadre de ma formation d'ingénieur, j'ai réalisé ce projet de fin d'études portant sur la conception et la réalisation d'**Aether Wallet**, une application de portefeuille de crypto-monnaies de niveau production, développée avec Flutter pour un déploiement mobile multi-plateforme.*

*Le module fonctionnel que j'ai développé offre une solution complète pour la gestion d'actifs numériques, dotée d'une interface utilisateur sophistiquée de type **Deep Glassmorphism**. Plus précisément, j'ai travaillé sur l'intégration d'interactions réelles avec la blockchain Ethereum (Mainnet). L'application intègre des fonctionnalités critiques telles que la génération de portefeuilles via phrases mnémoniques BIP39, la signature sécurisée de transactions et la visualisation de données de marché en temps réel via des connexions WebSocket. Le système inclut également un scanner QR holographique et un assistant IA pour guider l'utilisateur.*

*Mon travail garantit que la gestion des avoirs en crypto-monnaies s'effectue de manière sécurisée, immersive et performante, assurant une persistance fiable des données via PostgreSQL et une mise en cache rapide avec Redis. L'architecture respecte les principes de la **Clean Architecture** combinés au pattern de gestion d'état **BLoC** pour assurer l'évolutivité.*

La réussite de ce projet repose sur l'adoption d'un processus de développement adhérant à la méthodologie Agile, et pour la modélisation architecturale, j'ai choisi UML. Pour mener à bien ce projet, j'ai mobilisé l'ensemble des compétences acquises durant mon cursus en développement mobile et technologies blockchain.

L'environnement technique dans lequel j'ai travaillé comprenait Flutter/Dart, Node.js/Express, PostgreSQL, Redis, Web3 et Angular pour le tableau de bord d'administration.

Mots-clés : Portefeuille Crypto-monnaie, Flutter, Ethereum, Blockchain, Clean Architecture, BLoC, Deep Glassmorphism, Web3, Développement Mobile

Table des figures

1.1	Cycle de développement Agile adopté	6
1.2	Planning GANTT du projet	7
2.1	Diagramme de cas d'utilisation : Gestion des Portefeuilles	11
2.2	Diagramme de cas d'utilisation : Transactions	12
2.3	Diagramme de séquence : Processus d'envoi d'ETH	13
2.4	Diagramme de classe - Entités du domaine Aether Wallet	15
4.1	Écran de démarrage Aether Wallet	26
4.2	Interface de connexion sécurisée	26
4.3	Tableau de bord principal	27
4.4	Liste des crypto-monnaies	28
4.5	Détail et graphique d'un token	29
4.6	Formulaire d'envoi de transaction	30
4.7	Scanner QR Code intelligent	31
4.8	Historique des transactions	32
4.9	Vue d'ensemble du Dashboard Admin	33
4.10	Gestion des utilisateurs et rôles	33

Liste des tableaux

1.1	Fiche technique du projet Aether Wallet	4
1.2	Planning prévisionnel du projet Aether Wallet	6
2.1	Description des cas d'utilisation – Gestion Portefeuille	12
2.2	Description des cas d'utilisation – Transactions	12
2.3	Description du diagramme de séquence – Transaction	14
2.4	Description des entités - Aether Wallet	16

Liste des abréviations

API	Application Programming Interface
BIP	Bitcoin Improvement Proposal
BLoC	Business Logic Component
CI/CD	Continuous Integration / Continuous Delivery
EMSI	Ecole Marocaine des Sciences de l'Ingénieur
ETH	Ethereum
EVM	Ethereum Virtual Machine
IDE	Integrated Development Environment
JSON	JavaScript Object Notation
JWT	JSON Web Token
ORM	Object-Relational Mapping
PFE	Projet de Fin d'Etudes
RPC	Remote Procedure Call
SDK	Software Development Kit
SGBD	Système de Gestion de Base de Données
SQL	Structured Query Language
UI	User Interface
UML	Unified Modeling Language
UX	User Experience
Web3	Web 3.0 (Decentralized Web)

Table des matières

Introduction générale	1
1 Contexte général du projet	3
1.1 Introduction	3
1.2 Cadre du projet	3
1.2.1 Contexte Académique	3
1.2.2 Problématique	3
1.3 Présentation du projet : Aether Wallet	4
1.3.1 Vision du projet	4
1.3.2 Périmètre fonctionnel	4
1.4 Objectifs du projet	5
1.5 Processus de développement	5
1.5.1 Méthodologie Agile	5
1.5.2 Planning prévisionnel	6
1.5.3 Diagramme de GANTT	7
1.6 Conclusion	7
2 Analyse et Conception	9
2.1 Introduction	9
2.1.1 Modélisation UML	9
2.2 Identification des acteurs	9
2.3 Besoin de spécification	10
2.3.1 Besoins fonctionnels	10
2.3.2 Besoins non fonctionnels	10
2.4 Diagrammes de cas d'utilisation	10
2.4.1 Définition	10
2.4.2 Diagrammes de cas d'utilisation : Gestion des Portefeuilles	11
2.4.3 Diagrammes de cas d'utilisation : Transactions et Marché	12
2.5 Diagrammes de séquence	13
2.5.1 Définition	13
2.5.2 Diagramme de séquence : Envoi d'une Transaction	13
2.6 Diagramme de classe	14
2.6.1 Définition	14
2.6.2 Diagramme de classe : Entités Métier Aether Wallet	15
2.7 Conclusion	16
3 Téchnologie et outils de développement	18
3.1 Introduction	18
3.2 Architecture du système	18

3.3	Environnement logiciel et outils de développement	19
3.3.1	Technologies Frontend Mobile	19
3.3.2	Backend et Blockchain	20
3.3.3	Bases de données et cache	21
3.3.4	Outils de développement	22
3.4	Conclusion	23
4	Réalisation de l'application	25
4.1	Introduction	25
4.2	Implémentation de l'Interface Graphique de l'Application	25
4.2.1	Workflow Mobile (Flutter)	25
4.2.2	Tableau de Bord Admin (Angular)	32
4.3	Conclusion	34
	Perspective	35
	Conclusion générale	36

Introduction générale

Dans le cadre de la formation en ingénierie informatique et réseaux, spécialité MIAGE, un stage a été conduit au sein de l'entreprise Zynerator, située à Marrakech, du 1^{er} juillet 2025 au 1^{er} septembre 2025.

L'apparition du besoin d'une solution de gestion des congés et de leur validation découle de la complexité des processus administratifs et RH, nécessitant un suivi structuré et automatisé. La problématique principale réside dans l'optimisation du traitement des demandes de congé tout en garantissant la conformité aux règles de validation hiérarchique.

Le projet a porté sur le développement d'une application basée sur une architecture microservices, permettant la gestion des demandes de congé et leur circuit de validation. Un module fonctionnel dédié à cette gestion a été conçu et implémenté, assurant un traitement transparent et structuré des demandes.

Le rapport est structuré en quatre chapitres principaux. Le premier chapitre présente le contexte général du projet et fournit une vue d'ensemble du sujet traité. Le deuxième chapitre détaille l'analyse et la conception du projet, en exposant les étapes de sa structuration. Le troisième chapitre présente les technologies et outils de développement utilisés, en justifiant les choix techniques. Enfin, le quatrième chapitre illustre la réalisation concrète de l'application, ses fonctionnalités et les résultats obtenus.

*Chapitre 1 :
Contexte général du projet*

Chapitre 1

Contexte général du projet

1.1 Introduction

La révolution des crypto-monnaies et de la technologie blockchain a transformé la façon dont nous concevons les transactions financières. Cependant, l'adoption massive de ces technologies se heurte souvent à la complexité des outils existants. C'est dans ce contexte que naît le projet ****Aether Wallet****.

Ce premier chapitre établit le cadre de notre Projet de Fin d'Études. Il présente d'abord le contexte académique et la problématique liée à la sécurité et l'ergonomie des portefeuilles actuels. Ensuite, il détaille la solution proposée en précisant ses objectifs et son périmètre fonctionnel. Enfin, il expose la démarche méthodologique Agile adoptée pour mener à bien le développement de cette application mobile.

1.2 Cadre du projet

1.2.1 Contexte Académique

Ce projet s'inscrit dans le cadre de l'obtention du diplôme d'Ingénieur d'État en Ingénierie Informatique et Développement Mobile. Il représente l'aboutissement du cursus de formation universitaire 2024-2025, permettant de synthétiser les compétences techniques acquises (Flutter, Blockchain, Architecture Logicielle) et les compétences de gestion de projet.

1.2.2 Problématique

Le développement d'une application de portefeuille crypto-monnaie moderne fait face à plusieurs défis majeurs identifiés lors de l'analyse préliminaire :

- **Sécurité des actifs** : Comment garantir un stockage ultra-sécurisé des clés privées sur un appareil mobile exposé aux attaques (phishing, malwares) ?

- **Complexité utilisateur** : La technologie blockchain (frais de gas, adresses hexadécimales, délais de confirmation) est souvent inaccessible pour le grand public.
- **Performance et Réactivité** : La nécessité de gérer des flux de données en temps réel (prix du marché) et des opérations cryptographiques lourdes sans dégrader l'expérience utilisateur.

1.3 Présentation du projet : Aether Wallet

1.3.1 Vision du projet

Aether Wallet est une application mobile de niveau production conçue pour offrir une expérience utilisateur immersive tout en garantissant une sécurité maximale. Le nom "Aether" fait référence à la nature éthérée des actifs numériques. Le projet se distingue par son interface **Deep Glassmorphism** optimisée pour les écrans OLED et son architecture modulaire.

Attribut	Détail
Nom du projet	Aether Wallet
Type	Projet académique
Plateformes	iOS, Android, Web (Flutter Cross-platform)
Blockchain	Ethereum Mainnet (Web3 Integration)
Architecture	Clean Architecture + BLoC Pattern
Design System	Deep Glassmorphism (Neon/Dark)

TABLE 1.1 – Fiche technique du projet Aether Wallet

1.3.2 Périmètre fonctionnel

Le projet couvre trois axes fonctionnels principaux pour répondre aux besoins d'un utilisateur de crypto-monnaies moderne :

- **Gestion des Actifs (Wallet)**
 - Génération de portefeuilles via mnémonique BIP39.
 - Importation sécurisée de portefeuilles existants.
 - Consultation de solde en temps réel et envoi de transactions ETH.
- **Market Intelligence**
 - Suivi des cours en temps réel via WebSocket (Binance/CoinGecko).
 - Graphiques interactifs et historiques de prix.
 - Système d'alertes de prix personnalisables.
- **Outils Avancés**
 - Scanner QR holographique pour les adresses.
 - Galerie NFT pour la visualisation des collections.
 - Assistant Chat IA intégré pour l'aide contextuelle.

1.4 Objectifs du projet

Les objectifs du projet sont classés en deux catégories distinctes :

- **Objectifs Fonctionnels :**
 - Permettre la création et la restauration de portefeuilles sans tiers de confiance.
 - Assurer l'exécution de transactions sur le réseau Ethereum Mainnet.
 - Fournir une visualisation claire et instantanée des données de marché.
- **Objectifs Non-Fonctionnels :**
 - Sécurité : Chiffrement local AES-256 et authentification biométrique.
 - Performance : Interface fluide à 60 FPS (Flutter Skia).
 - Maintenabilité : Code structuré selon la Clean Architecture pour faciliter les évolutions futures.

1.5 Processus de développement

1.5.1 Méthodologie Agile

Pour mener à bien ce projet complexe impliquant blockchain et mobile, nous avons adopté une méthodologie **Agile (Scrum adapté)**. Cette approche itérative permet de valider chaque brique technique (infrastructure, domaine, UI) avant de passer à la suivante, réduisant ainsi les risques liés à l'intégration blockchain.

Le développement a été découpé en Sprints logiques correspondant aux couches de l'architecture logicielle :

- **Product Backlog :** Liste des fonctionnalités (Wallet, Swap, NFT, Chat).
- **Sprints :** Phases de développement de 2 à 3 semaines.
- **Revue :** Validation des fonctionnalités sur simulateurs et appareils réels.

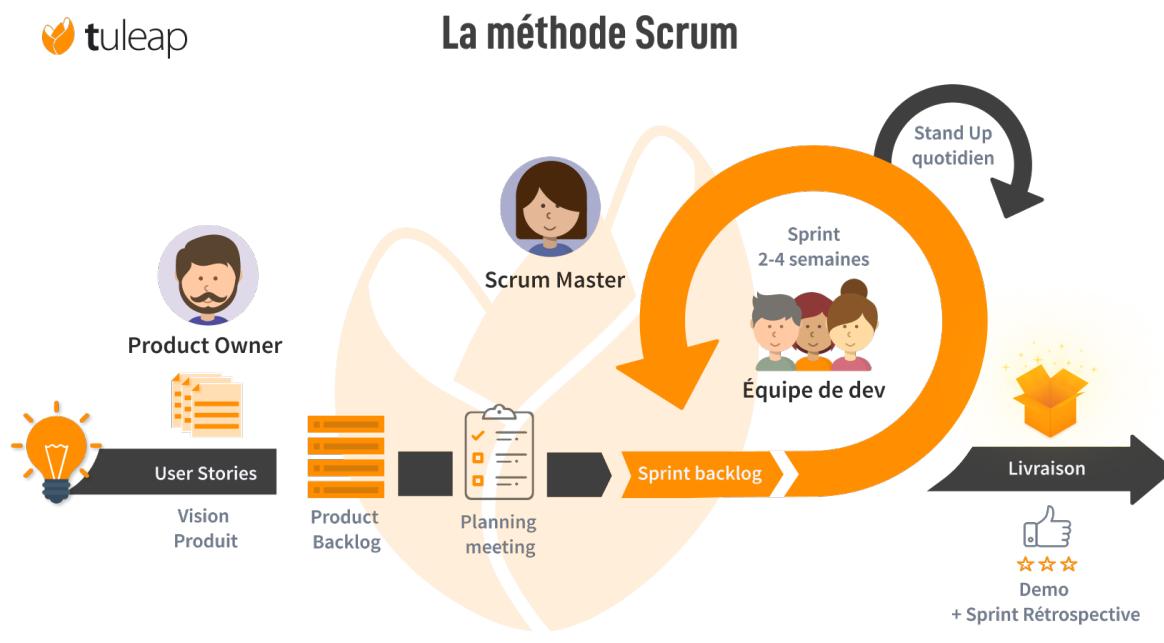


FIGURE 1.1 – Cycle de développement Agile adopté

1.5.2 Planning prévisionnel

Le projet a été planifié sur une durée de 15 semaines, réparties en 6 phases clés :

Phase	Focus	Activités Principales	Durée
1	Infrastructure	<ul style="list-style-type: none"> - Initialisation Flutter, Configuration CI/CD, Linter. - Mise en place de la Clean Architecture. 	1 sem.
2	Domain & Data	<ul style="list-style-type: none"> - Implémentation des Entités et Repositories. - Configuration BDD locale (Hive) et Sécurité. 	1 sem.
3	Presentation (BLoC)	<ul style="list-style-type: none"> - Gestion d'état avec BLoC. - Logique métier (WalletBloc, MarketBloc). 	1 sem.
4	UI / UX	<ul style="list-style-type: none"> - Intégration du design Deep Glassmorphism. - Développement des écrans et animations. 	1 sem.
5	Blockchain	<ul style="list-style-type: none"> - Intégration Web3dart (Ethereum Mainnet). - Signature de transactions et WebSocket. 	1 sem.
6	Tests & Livraison	<ul style="list-style-type: none"> - Tests d'intégration et optimisation. - Rédaction du rapport final. 	1 sem.

TABLE 1.2 – Planning prévisionnel du projet Aether Wallet

1.5.3 Diagramme de GANTT

Le diagramme de GANTT ci-dessous illustre l'enchaînement temporel des phases de développement, assurant une visibilité claire sur l'avancement du projet.

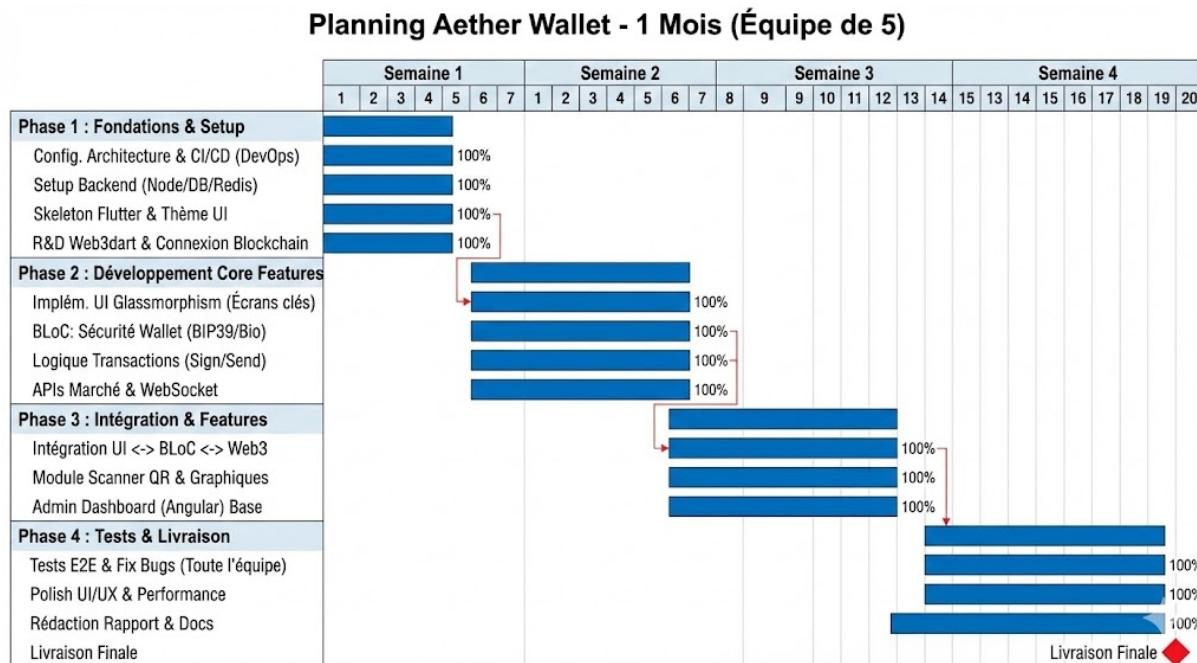


FIGURE 1.2 – Planning GANTT du projet

1.6 Conclusion

Ce chapitre a posé les bases contextuelles du projet **Aether Wallet**. Nous avons défini la problématique liée à la complexité et à la sécurité des portefeuilles crypto actuels, et présenté notre solution : une application mobile alliant sécurité industrielle et design innovant. La méthodologie Agile adoptée et le planning structuré garantissent une exécution rigoureuse des phases de développement. Ces éléments constituent le socle sur lequel s'appuiera l'analyse et la conception technique détaillée dans le chapitre suivant.

*Chapitre 2 :
Analyse et conception*

Chapitre 2

Analyse et Conception

2.1 Introduction

Ce deuxième chapitre se divise en deux volets principaux : le premier est consacré à l'identification des acteurs et à la spécification des besoins, tandis que le second aborde la modélisation UML du système à travers les diagrammes de cas d'utilisation, de séquence, ainsi que le diagramme de classes.

2.1.1 Modélisation UML

UML (Unified Modeling Language) est un langage de modélisation graphique standardisé utilisé pour visualiser, spécifier, construire et documenter les artefacts d'un système logiciel. Il offre une méthode standard pour écrire les plans d'un système, incluant les aspects conceptuels tels que les processus métier et les fonctions du système, ainsi que les aspects concrets comme les expressions de langages de programmation, les schémas de bases de données et les composants logiciels.

Dans le cadre de ce projet, UML nous permet de structurer l'architecture complexe d'un portefeuille crypto-monnaie, en modélisant les interactions entre l'utilisateur, l'application mobile et la blockchain Ethereum.

2.2 Identification des acteurs

L'identification des acteurs est une étape essentielle car elle permet de définir les différentes entités externes qui interagissent avec le système Aether Wallet.

— **Acteurs du système :**

- **Utilisateur** : Propriétaire du portefeuille, il crée des comptes, signe des transactions, consulte ses actifs et gère ses paramètres de sécurité.
- **Blockchain Ethereum (Mainnet)** : Système externe décentralisé qui valide les transactions et stocke l'état global des soldes (Ledger).
- **API de Marché (CoinGecko/Binance)** : Fournisseur de données tiers utilisé pour récupérer les prix des crypto-monnaies en temps réel.

2.3 Besoin de spécification

2.3.1 Besoins fonctionnels

Les besoins fonctionnels décrivent les fonctionnalités que le système doit fournir pour répondre aux attentes des utilisateurs.

— **Fonctionnalités principales :**

- **Gestion de Portefeuille** : Génération de mnémonique (BIP39), importation de clés privées, et sauvegarde sécurisée.
- **Transactions** : Envoi et réception d'ETH, estimation des frais de gas, et scan de QR Code.
- **Suivi de Marché** : Visualisation des prix en temps réel et graphiques interactifs.
- **Sécurité** : Authentification biométrique et chiffrement local des données.

2.3.2 Besoins non fonctionnels

Les besoins non fonctionnels définissent les contraintes de qualité et de performance du système Aether Wallet.

— **Contraintes de qualité et de performance :**

- **Performance** : L'interface doit être fluide (60 FPS) et les données de marché mises à jour en temps réel.
- **Sécurité** : Les clés privées ne doivent jamais quitter l'appareil de l'utilisateur (chiffrement AES-256).
- **Disponibilité** : Le backend doit assurer une haute disponibilité pour l'historique des transactions et les alertes.
- **Ergonomie** : L'interface Deep Glassmorphism doit être intuitive et optimisée pour les écrans OLED.

2.4 Diagrammes de cas d'utilisation

2.4.1 Définition

Les diagrammes de cas d'utilisation décrivent les fonctionnalités du système du point de vue de l'utilisateur. Ils aident à délimiter le périmètre fonctionnel de l'application mobile.

2.4.2 Diagrammes de cas d'utilisation : Gestion des Portefeuilles

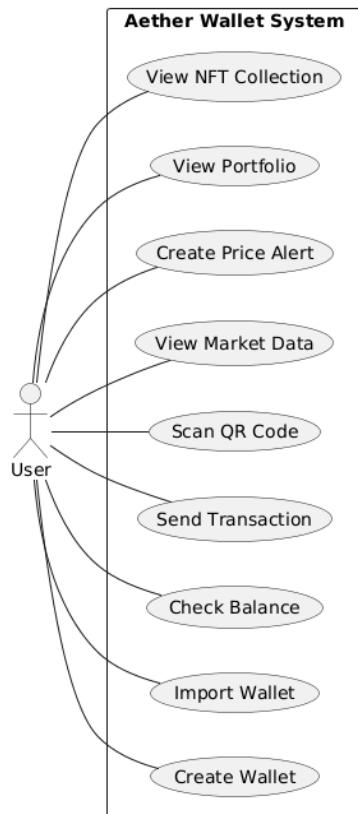


FIGURE 2.1 – Diagramme de cas d'utilisation : Gestion des Portefeuilles

Cas d'utilisation	Description
S'authentifier (Biométrie)	Obligatoire. L'utilisateur sécurise l'accès à l'application via FaceID ou TouchID pour protéger ses actifs sensibles.
Créer un portefeuille	Génération d'une nouvelle phrase mnémonique de 12 mots (BIP39) et dérivation des clés cryptographiques. L'utilisateur doit sauvegarder cette phrase.
Importer un portefeuille	Restauration d'un compte existant en saisissant une phrase mnémonique valide. Le système vérifie la validité de la phrase.
Consulter le solde	Interrogation de la blockchain Ethereum pour afficher le solde ETH disponible en temps réel.

Cas d'utilisation	Description
Sauvegarder Mnémonique	L'utilisateur peut visualiser sa phrase de récupération après une authentification biométrique renforcée.

TABLE 2.1 – Description des cas d'utilisation – Gestion Portefeuille

2.4.3 Diagrammes de cas d'utilisation : Transactions et Marché

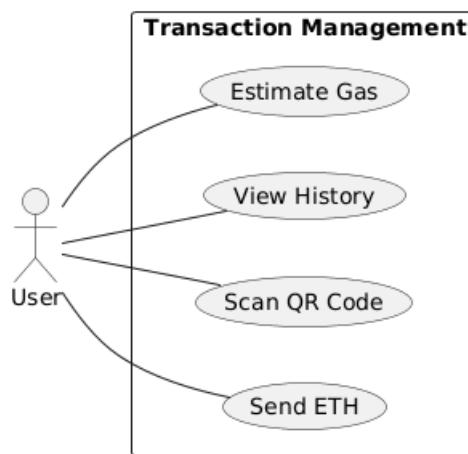


FIGURE 2.2 – Diagramme de cas d'utilisation : Transactions

Cas d'utilisation	Description
Envoyer des fonds	L'utilisateur initie un transfert d'ETH vers une adresse tierce. Le système estime les frais de gas avant confirmation.
Scanner QR Code	Utilisation de la caméra pour lire instantanément une adresse de destinataire au format standard Ethereum (0x...).
Consulter le marché	Visualisation des prix en direct et des graphiques d'évolution des cryptomonnaies via WebSocket.
Estimer les frais (Gas)	Calcul automatique du coût de la transaction en fonction de la congestion du réseau Ethereum.
Voir l'historique	Consultation de la liste des transactions passées (envoyées et reçues) avec leurs statuts (Confirmé/Échoué).

TABLE 2.2 – Description des cas d'utilisation – Transactions

2.5 Diagrammes de séquence

2.5.1 Définition

Le diagramme de séquence illustre comment les objets interagissent entre eux selon un ordre chronologique. Il est crucial ici pour montrer le flux sécurisé d'une transaction blockchain, de l'interface utilisateur jusqu'au réseau Ethereum.

2.5.2 Diagramme de séquence : Envoi d'une Transaction

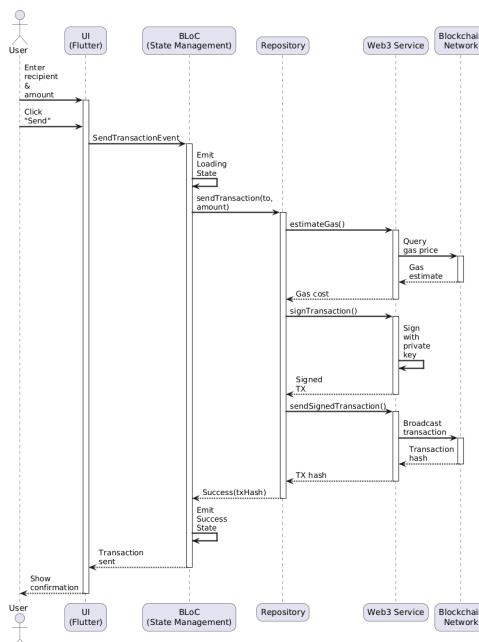


FIGURE 2.3 – Diagramme de séquence : Processus d'envoi d'ETH

Élément	Description
Acteurs	- Utilisateur (Initiateur), Application UI (Interface), TransactionBloc (Logique métier), Web3Service (Connектор), Blockchain Ethereum.
Processus Principal	<ul style="list-style-type: none"> - 1. L'utilisateur saisit l'adresse et le montant. - 2. Le système estime les frais de Gas via RPC. - 3. L'utilisateur confirme et signe la transaction localement (clé privée). - 4. La transaction signée est diffusée sur le réseau.

Élément	Description
Scénarios conditionnels	<ul style="list-style-type: none"> - Si succès : Hash de transaction retourné, solde mis à jour localement. - Si échec (Gas insuffisant) : Message d'erreur affiché à l'utilisateur.
Mise à jour système	<ul style="list-style-type: none"> - Une fois confirmée par les mineurs, la transaction est ajoutée à l'historique local et le solde est synchronisé.

TABLE 2.3 – Description du diagramme de séquence – Transaction

2.6 Diagramme de classe

2.6.1 Définition

Le diagramme de classes représente la structure statique du système, détaillant les entités, leurs attributs et leurs relations. Il reflète l'architecture "Domain Layer" de notre Clean Architecture, indépendante des détails d'implémentation.

2.6.2 Diagramme de classe : Entités Métier Aether Wallet

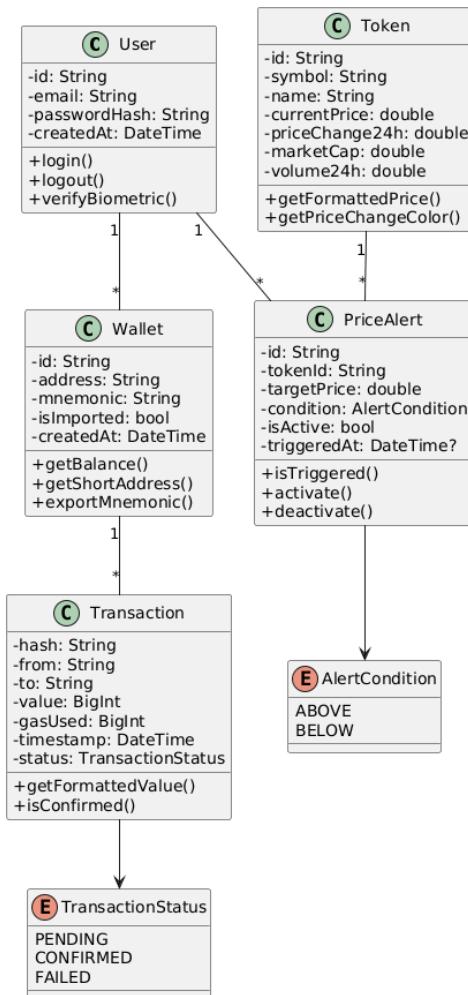


FIGURE 2.4 – Diagramme de classe - Entités du domaine Aether Wallet

Entité	Description
User (Collaborateur)	- Représente l'utilisateur de l'application, lié à ses préférences de sécurité (Biométrie, PIN), ses paramètres de notification et ses devises préférées (USD, EUR).
Wallet (Portefeuille)	- Entité centrale contenant l'adresse publique, la clé privée (cryptée), la phrase mnémonique et le solde actuel. Relation 1-1 avec l'Utilisateur.

Entité	Description
Transaction	- Enregistre les détails d'un transfert : hash unique, adresse expéditeur, adresse destinataire, montant, frais de gas, timestamp et statut (Pending, Confirmed, Failed).
Token	- Représente un actif numérique (ETH ou ERC-20) avec son symbole, son nom, son prix actuel, sa variation sur 24h et son logo.
PriceAlert	- Définit une règle de surveillance de prix créée par l'utilisateur (ex : Notifier si ETH > 3000\$), liée à un Token spécifique.

TABLE 2.4 – Description des entités - Aether Wallet

2.7 Conclusion

Au fil de ce chapitre, nous avons exploré en détail la conception de l'application Aether Wallet en nous appuyant sur les standards UML. Les diagrammes de cas d'utilisation ont clarifié le périmètre fonctionnel (Gestion de portefeuille, Transactions), les diagrammes de séquence ont sécurisé la logique des flux blockchain, et le diagramme de classes a structuré les données métier. Cette analyse constitue la base solide pour l'implémentation technique qui sera présentée dans le chapitre suivant.

*Chapitre 3 :
Téchnologie et outil de développement*

Chapitre 3

Téchnologie et outils de développement

3.1 Introduction

Ce chapitre présente les technologies, frameworks et outils utilisés pour la conception et le développement de l'application mobile **Aether Wallet**. Il décrit l'architecture logicielle adoptée, l'environnement de développement ainsi que les choix techniques effectués afin de garantir la sécurité, la performance et la maintenabilité du système, notamment dans un contexte blockchain.

3.2 Architecture du système

L'application **Aether Wallet** repose sur une **Clean Architecture** combinée à une architecture client–serveur. Cette approche vise à assurer une séparation claire des responsabilités entre les différentes couches de l'application, facilitant ainsi l'évolution, les tests et la maintenance du code.

— Clean Architecture :

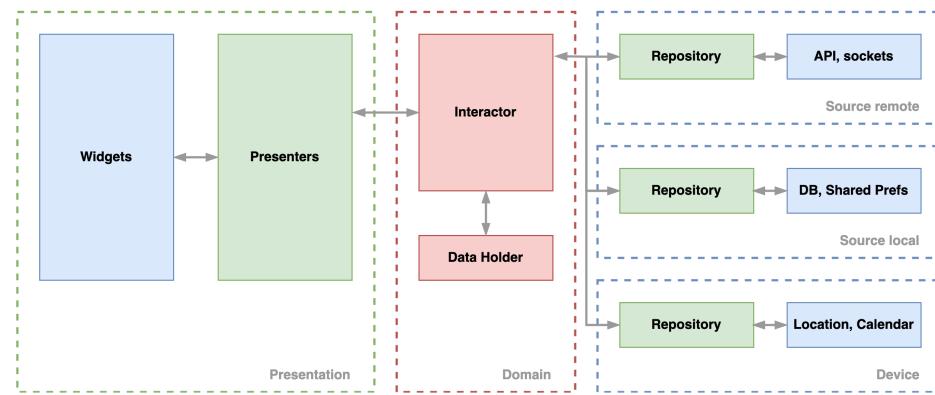
- **Couche Présentation** : Interface utilisateur Flutter et gestion d'état via le pattern BLoC.
- **Couche Domaine** : Logique métier, entités et cas d'utilisation indépendants de toute technologie.
- **Couche Données** : Accès aux API backend, à la blockchain Ethereum et aux services de cache.

— Communication système :

- Communication sécurisée entre l'application mobile et le backend Node.js via des API REST.
- Interaction directe avec la blockchain Ethereum pour la gestion des portefeuilles et transactions.

— Avantages de l'architecture :

- Forte maintenabilité et testabilité.
- Indépendance entre l'interface utilisateur et la logique métier.
- Facilité d'intégration des services blockchain.



Clean Architecture appliquée à Aether Wallet

3.3 Environnement logiciel et outils de développement

3.3.1 Technologies Frontend Mobile

Flutter :

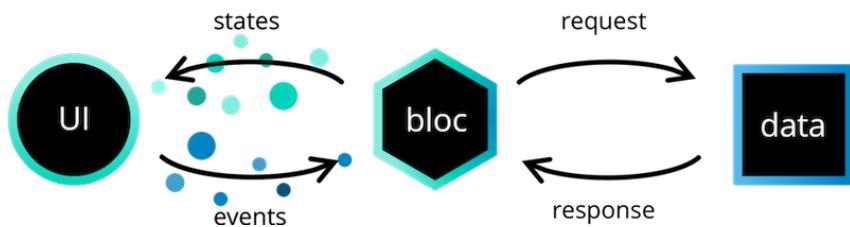


Flutter

Flutter est un framework open source développé par Google destiné à la création d'applications multiplateformes à partir d'un seul code source. Il permet de cibler simultanément Android et iOS tout en offrant des performances proches du natif grâce à son moteur de rendu graphique intégré.

Dans le cadre du projet **Aether Wallet**, Flutter a été choisi pour sa rapidité de développement, sa richesse en composants graphiques et sa capacité à fournir une interface utilisateur fluide, sécurisée et homogène sur différentes plateformes mobiles.

BLoC Pattern :



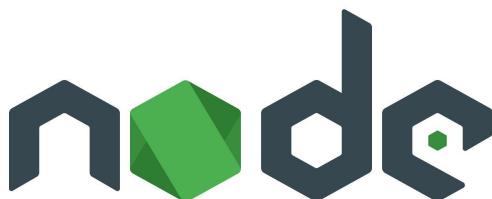
Pattern BLoC

Le pattern BLoC (Business Logic Component) est un modèle de gestion d'état reposant sur un flux de données unidirectionnel. Les événements déclenchés par l'utilisateur sont traités par la logique métier, puis traduits en états observables par l'interface utilisateur.

Ce pattern est utilisé dans Aether Wallet pour gérer les processus critiques tels que l'authentification, la consultation des soldes et l'exécution des transactions, garantissant une séparation claire entre logique métier et interface graphique.

3.3.2 Backend et Blockchain

Node.js :

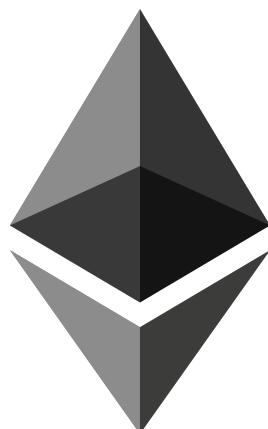


Node.js

Node.js est un environnement d'exécution JavaScript côté serveur basé sur le moteur V8. Il adopte un modèle asynchrone non bloquant, particulièrement adapté aux applications nécessitant une forte capacité de montée en charge.

Dans Aether Wallet, Node.js est utilisé pour implémenter l'API backend assurant la gestion des utilisateurs, la validation des transactions et la communication sécurisée avec les services blockchain.

Ethereum :



Ethereum

Ethereum est une plateforme blockchain décentralisée permettant l'exécution de contrats intelligents et la gestion d'actifs numériques. Elle garantit la transparence, l'immutabilité et la sécurité des transactions grâce à son architecture distribuée.

Dans le projet Aether Wallet, Ethereum constitue le réseau principal pour la gestion des portefeuilles, l'envoi et la réception de transactions, ainsi que l'interaction avec des tokens standards tels que ERC-20.

3.3.3 Bases de données et cache

PostgreSQL :



PostgreSQL

PostgreSQL est un système de gestion de base de données relationnelle open source reconnu pour sa robustesse et sa conformité aux propriétés ACID. Il assure la persistance sécurisée et cohérente des données.

Dans Aether Wallet, PostgreSQL est utilisé pour stocker les informations utilisateur, l'historique des transactions et les métadonnées associées aux portefeuilles.

Redis :

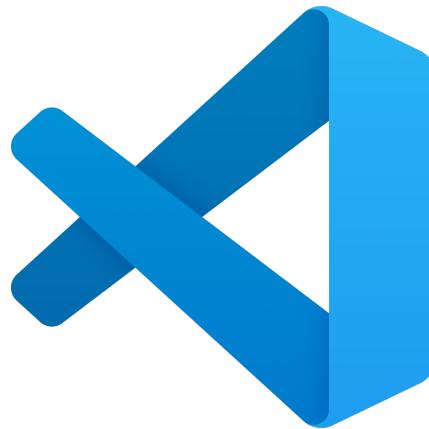


Redis

Redis est un système de stockage de données en mémoire utilisé comme cache afin d'améliorer les performances globales de l'application. Il permet de réduire la latence et la charge sur la base de données principale.

3.3.4 Outils de développement

Visual Studio Code :



Visual Studio Code

Visual Studio Code est l'environnement de développement principal utilisé pour le développement Flutter, Node.js et Angular. Il offre des outils avancés de débogage et une large extensibilité.

GitHub :



GitHub

GitHub est utilisé pour la gestion de version, la collaboration entre développeurs et le suivi de l'évolution du projet tout au long du cycle de développement.

Postman :



Postman

Postman est utilisé pour tester et valider les API REST développées dans le backend Node.js.

3.4 Conclusion

Ce chapitre a présenté les technologies et outils utilisés pour le développement de l'application **Aether Wallet**. L'association de Flutter, Node.js et Ethereum, soutenue par une architecture propre et des outils modernes, constitue une base technique solide garantissant la performance, la sécurité et l'évolutivité du système. Cette base permet d'aborder la phase d'implémentation détaillée dans le chapitre suivant.

Chapitre 4 :
Réalisation de l'application

Chapitre 4

Réalisation de l'application

4.1 Introduction

L'implémentation représente l'étape cruciale qui suit directement la phase de conception. Elle consiste à traduire le modèle conceptuel élaboré auparavant en composants logiciels concrets, qui ensemble forment le système final. Cette étape marque la concrétisation des analyses et des plans en un produit fonctionnel et opérationnel.

4.2 Implémentation de l'Interface Graphique de l'Application

4.2.1 Workflow Mobile (Flutter)

Écran de démarrage (Splash Screen) :



FIGURE 4.1 – Écran de démarrage Aether Wallet

L'écran de démarrage introduit l'identité visuelle de l'application avec le logo Aether animé. En arrière-plan, le système initialise les services critiques et vérifie si une session est déjà active.

Authentification Biométrique :

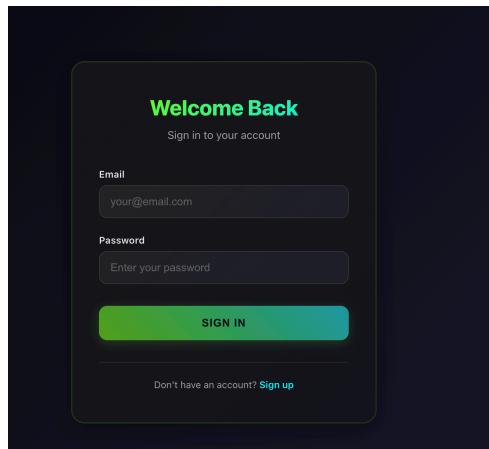


FIGURE 4.2 – Interface de connexion sécurisée

Pour garantir la sécurité des actifs, l'accès au portefeuille est protégé par authentification biométrique (FaceID/TouchID) ou un code PIN chiffré.

Tableau de bord principal (Dashboard) :

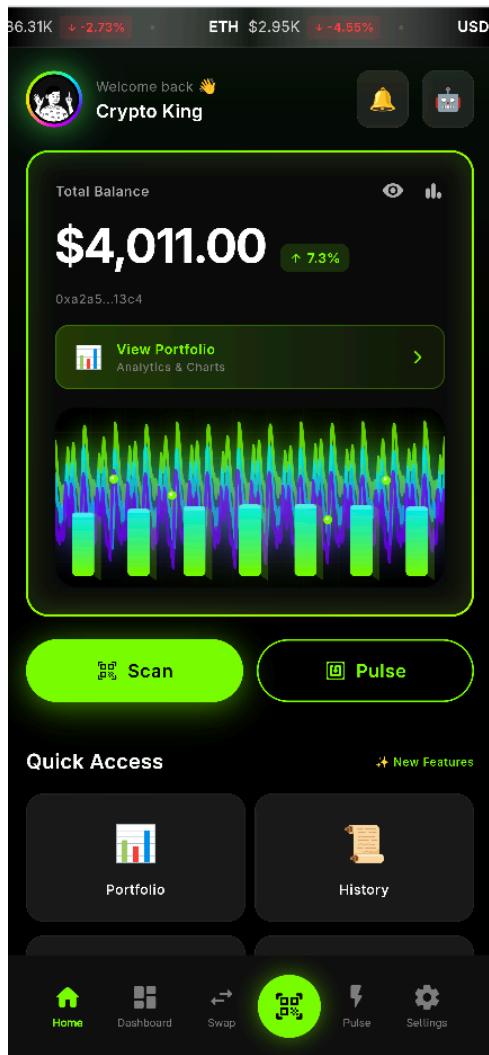


FIGURE 4.3 – Tableau de bord principal

Le tableau de bord centralise les informations essentielles pour l'utilisateur. Il affiche le solde total consolidé en USD, un graphique interactif de l'évolution du portefeuille, et des boutons d'action rapide.

Suivi du Marché en Temps Réel :

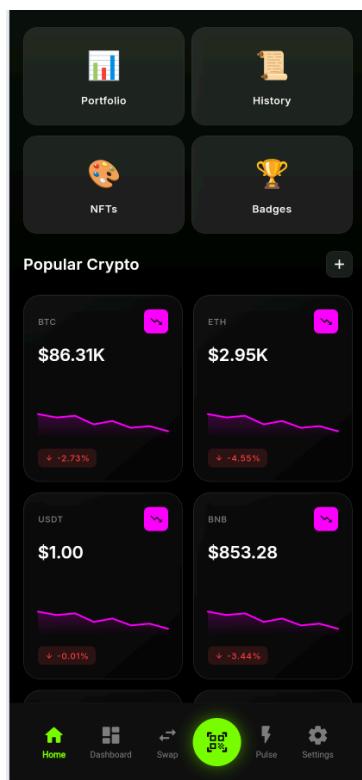


FIGURE 4.4 – Liste des crypto-monnaies

Cette vue liste les principales crypto-monnaies avec leurs prix mis à jour en temps réel via une connexion WebSocket. Chaque ligne présente le logo du token, son nom, son prix actuel et sa variation en pourcentage.

Détail d'un Token :



FIGURE 4.5 – Détail et graphique d'un token

En sélectionnant une crypto-monnaie, l'utilisateur accède à une vue détaillée affichant un graphique en chandeliers japonais ou linéaire pour analyser les tendances.

Envoi de fonds (Send Transaction) :

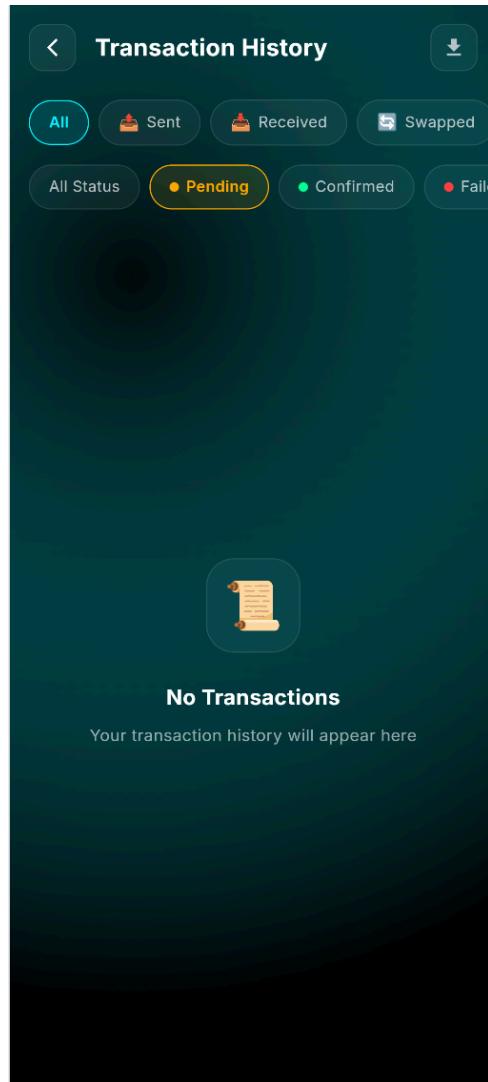


FIGURE 4.6 – Formulaire d'envoi de transaction

Le formulaire d'envoi permet de saisir l'adresse du destinataire et le montant à transférer. Une fonctionnalité clé est l'estimation automatique des frais de gas.

Scanner QR Holographique :

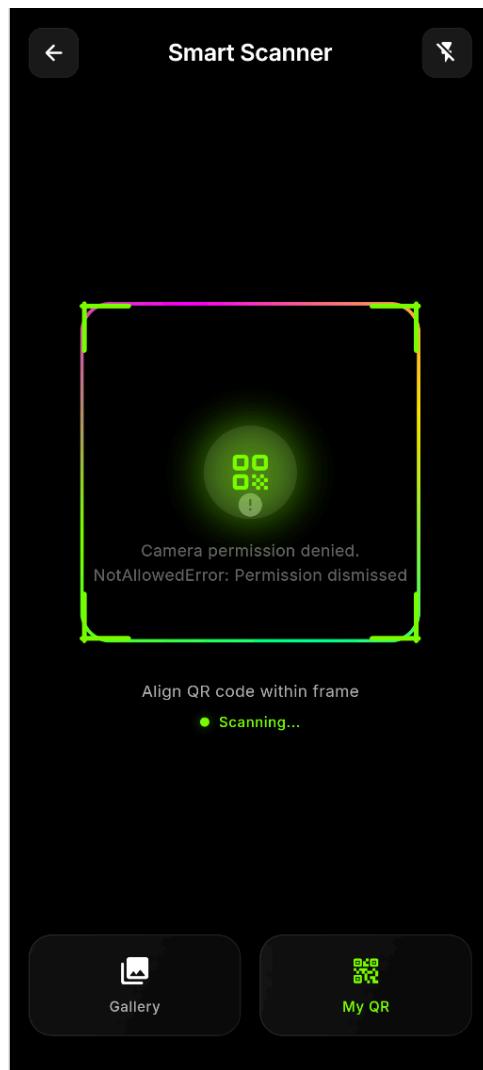


FIGURE 4.7 – Scanner QR Code intelligent

Pour faciliter la saisie des adresses, un scanner de QR Code est intégré. Il utilise une interface holographique pour guider l'utilisateur.

Historique des Transactions :

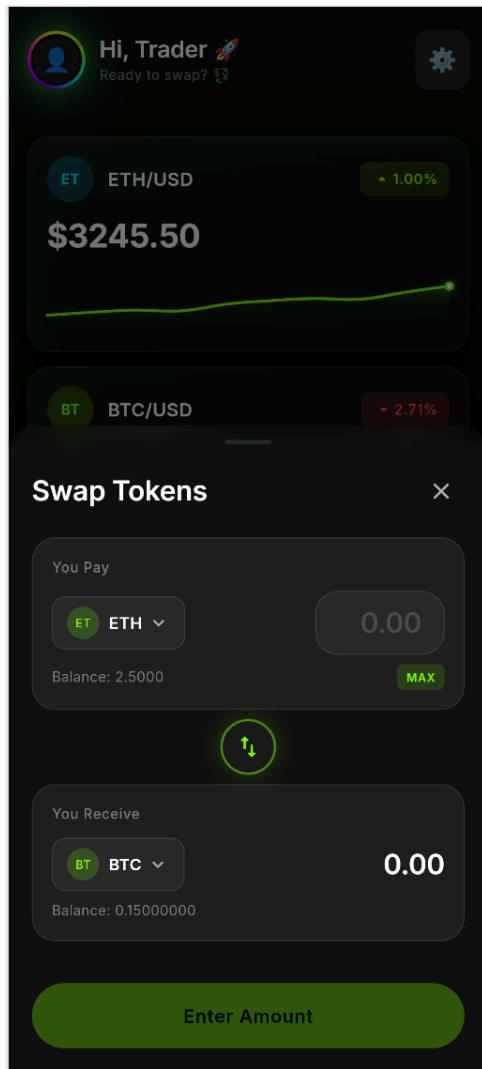


FIGURE 4.8 – Historique des transactions

Cette vue liste l'ensemble des transactions entrantes et sortantes avec leur statut (**Confirmé**, **En attente**, ou **Échoué**).

4.2.2 Tableau de Bord Admin (Angular)

Dashboard Administrateur :

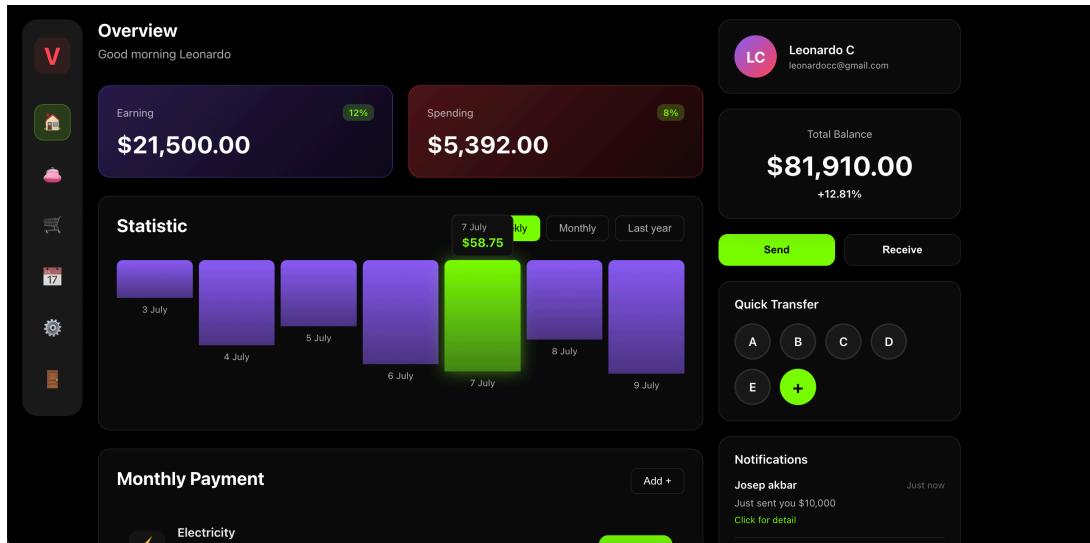


FIGURE 4.9 – Vue d'ensemble du Dashboard Admin

Le tableau de bord d'administration offre une vue globale sur la santé du système Aether Wallet, incluant des indicateurs clés de performance (KPI).

Gestion des Utilisateurs :

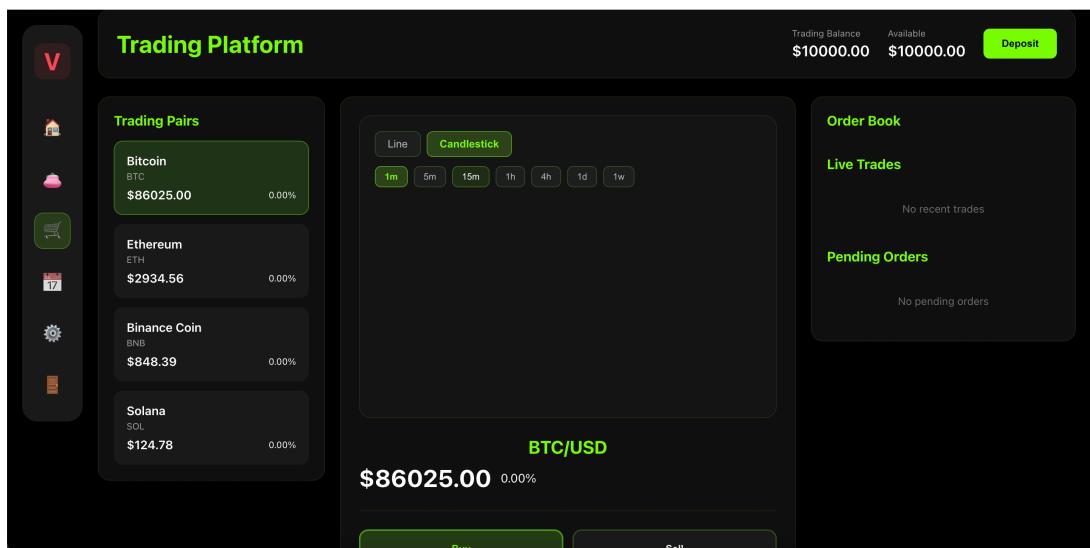


FIGURE 4.10 – Gestion des utilisateurs et rôles

Cette interface permet aux administrateurs de gérer les comptes utilisateurs, de visualiser les dates de dernière connexion et de gérer les statuts des comptes.

4.3 Conclusion

Ce chapitre a présenté l'implémentation complète de l'application Aether Wallet. Les interfaces développées offrent une expérience utilisateur fluide et immersive, respectant les standards du Deep Glassmorphism. L'intégration des fonctionnalités blockchain (transferts, historique, solde) a été réussie, validant les choix techniques effectués lors des phases précédentes.

Perspective

Dans la continuité du travail réalisé, une première perspective majeure consiste à étendre l'interopérabilité de l'application en intégrant le support de nouvelles blockchains telles que Bitcoin, Polygon et la Binance Smart Chain. Cela permettrait de transformer Aether Wallet en une véritable solution multi-chaînes. De plus, l'intégration du protocole WalletConnect est envisagée pour permettre aux utilisateurs de connecter leur portefeuille mobile à n'importe quelle application décentralisée (dApp) du marché, élargissant ainsi considérablement les cas d'usage possibles.

Une autre piste d'amélioration stratégique réside dans l'enrichissement des fonctionnalités financières et de l'assistance intelligente. L'intégration d'un module de Swap décentralisé (DEX) permettrait aux utilisateurs d'échanger des tokens directement dans l'application sans passer par des plateformes externes. Parallèlement, l'amélioration de l'Assistant Chat IA via un modèle de langage (LLM) plus performant offrirait une aide contextuelle plus précise, capable d'analyser les tendances du marché et de guider les utilisateurs débutants avec une plus grande pertinence.

Enfin, une attention particulière sera portée au renforcement de la sécurité et à l'ancrage physique du portefeuille. À long terme, l'objectif est d'implémenter le support des portefeuilles matériels (Hardware Wallets) tels que Ledger ou Trezor, offrant ainsi un niveau de sécurité "Cold Storage" pour les montants importants. Cette évolution, couplée à des fonctionnalités de gouvernance décentralisée et de staking, positionnerait Aether Wallet comme une solution complète, sécurisée et adaptée aussi bien aux particuliers qu'aux investisseurs institutionnels.

Conclusion générale

Ce Projet de Fin d'Études a abouti à la conception et à la réalisation d'**Aether Wallet**, une application de portefeuille de crypto-monnaies de niveau production. Le projet a permis de répondre aux défis complexes de sécurité et d'ergonomie inhérents aux solutions actuelles, en proposant une interface utilisateur immersive basée sur le style Deep Glassmorphism et une architecture robuste.

Pour la réalisation de ce projet, un écosystème technologique moderne a été mobilisé, incluant Flutter pour le développement cross-platform, Node.js et TypeScript pour le backend, ainsi que PostgreSQL et Redis pour la gestion performante des données. L'intégration blockchain a été assurée via Web3dart, permettant une interaction directe et sécurisée avec le Mainnet Ethereum pour la gestion des transactions et des soldes.

L'approche méthodologique Agile Scrum a facilité le pilotage itératif du développement, tandis que l'adoption de la Clean Architecture et du pattern BLoC a garanti la maintenabilité du code. Cette expérience a permis de renforcer considérablement les compétences en ingénierie logicielle, notamment en cryptographie appliquée, en développement mobile avancé et en intégration Web3.

Ce travail a offert une vision complète du cycle de vie d'une application financière critique, de la conception architecturale à la validation technique. Il a permis de surmonter des défis concrets liés à la sécurité des clés privées (BIP39), à l'optimisation des performances graphiques (60 FPS) et à la gestion des données de marché en temps réel.

Perspectives : Les évolutions futures envisagées incluent le support de blockchains supplémentaires comme Bitcoin et Polygon, l'intégration du protocole WalletConnect pour interagir avec les applications décentralisées (dApps), ainsi que l'ajout de fonctionnalités de trading (Swap) et de support pour les portefeuilles matériels (Hardware Wallets).

