RÉPUBLIQUE TUNISIENNE

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université de Gabès

École Nationale d'Ingénieurs de Gabès

RAPPORT DE PROJET DE FIN D'ÉTUDES

Département de Génie des Communications et Réseaux

Étude et mise en place d'une solution basée sur la blockchain pour la délivrance des attestations de stage

Réalisé par : Ibtissem Aloui

Encadrant académique : [Nom de l'encadrant]
Encadrant professionnel : [Nom de l'encadrant]

Année universitaire : 2024–2025

Dédicace

[Contenu de la dédicace]

Remerciements

[Contenu des remerciements]

Table des matières

Table des figures

Liste des tableaux

Liste des Acronymes

- IPFS : InterPlanetary File System
- API : Application Programming Interface
- REST : Representational State Transfer
- CSV : Comma-Separated Values

Introduction générale

Avec la montée en puissance des technologies numériques, les méthodes traditionnelles de gestion des documents officiels sont devenues de moins en moins adaptées aux exigences actuelles d'efficacité, de sécurité et de vérifiabilité. C'est notamment le cas des attestations de stage, souvent traitées manuellement ou de manière centralisée, avec des risques de falsification, de perte ou de délais dans leur validation.

La technologie blockchain se présente comme une solution innovante à ces problématiques, offrant des garanties d'authenticité, de traçabilité et d'intégrité grâce à son caractère décentralisé et immuable. Elle permet d'automatiser et de sécuriser l'ensemble du cycle de vie d'un document sans dépendre d'un tiers de confiance.

C'est dans ce contexte que s'inscrit ce projet de fin d'études. Réalisé au sein de la division informatique du Groupe Chimique Tunisien (GCT) à Gabès, il a pour objectif de concevoir et développer une plateforme web pour la gestion des attestations de stage basée sur la blockchain. Cette solution permet aux entreprises de générer des attestations de manière sécurisée, aux établissements universitaires de les valider, et aux tiers (recruteurs, institutions) de les vérifier à l'aide d'un identifiant unique ou d'un code QR imprimé sur l'attestation. Ce code redirige vers une preuve en ligne hébergée sur le réseau IPFS et liée à une transaction blockchain.

La plateforme repose sur une architecture distribuée utilisant des technologies telles que Node.js, React.js, Solidity, Hardhat, IPFS et MySQL. Elle assure l'automatisation complète du cycle de génération, validation et consultation des attestations. Bien qu'initiée dans un cadre pédagogique, la solution est pensée pour évoluer vers une utilisation concrète en environnement professionnel, répondant aux exigences d'intégrité et de pérennité.

Ce rapport est structuré en trois chapitres :

- Le premier chapitre présente l'organisme d'accueil, analyse l'existant et introduit la solution retenue.
- Le deuxième chapitre traite de l'analyse des besoins et de la conception du système.
- Le troisième chapitre couvre les phases de développement, test, sécurisation et déploiement de la solution.

Nous conclurons par une synthèse des résultats obtenus et des perspectives d'évolution du projet.

Chapitre 1

Étude préliminaire

1.1 Introduction

Ce chapitre présente le cadre général du projet ainsi que l'environnement professionnel dans lequel il a été réalisé. Il expose la problématique, les solutions existantes et la technologie retenue.

1.2 Présentation de l'organisme d'accueil

Le Groupe Chimique Tunisien (GCT) 1 est une entreprise publique tunisienne fondée en 1953, spécialisée dans la transformation du phosphate en produits à forte valeur ajoutée tels que l'acide phosphorique et les engrais. Le GCT contribue significativement à l'économie nationale avec environ 4200 employés.

La Tunisie est ainsi le deuxième pays mondial en matière de valorisation du phosphate naturel, avec 85% de la production transformée.

Le projet a été réalisé au sein de la Direction Régionale des Usines de Gabès, plus précisément dans la division informatique, créée en 1981. Elle assure :

- Maintenance informatique ;
- Administration système et réseau ;
- Développement d'applications internes.

FIGURE 1 – Logo du Groupe Chimique Tunisien (GCT)

La structure de l'entreprise se définit comme l'ensemble des relations hiérarchiques et fonctionnelles entre les divers services et le personnel. Le GCT est structuré comme le montre l'organigramme de la figure 2

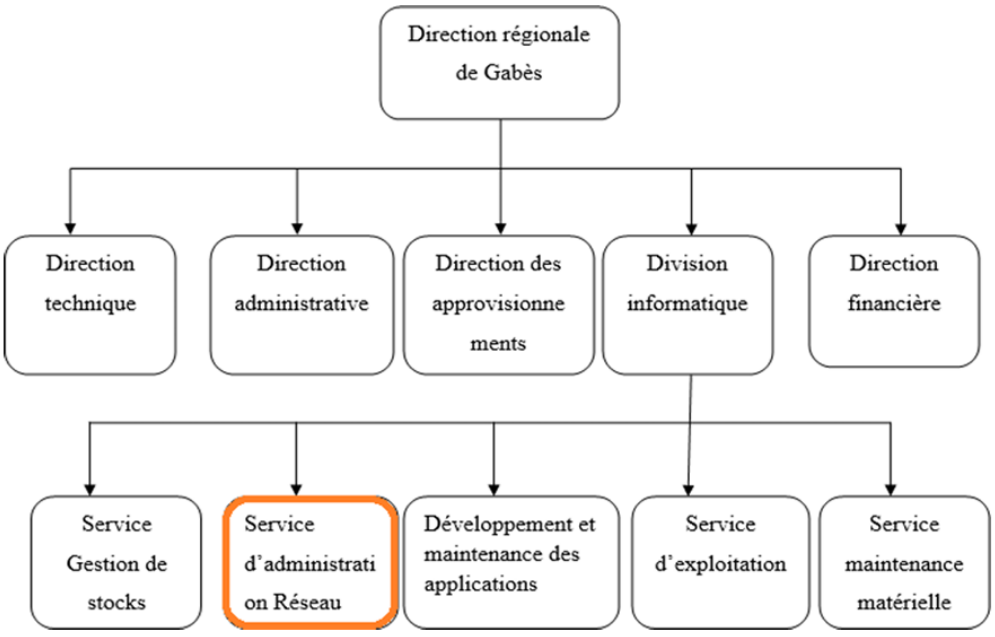

FIGURE 2 – Organigramme du GCT

La division informatique comprend trois services :

- Service bureautique.
- Service administration réseau.
- Centre informatique.

1.3 Problématique

Les attestations de stage sont des documents importants dans le parcours académique et professionnel des étudiants. Remises en format papier ou PDF à la fin du stage, elles peuvent cependant être facilement perdues, modifiées ou falsifiées. Leur vérification par les établissements ou les recruteurs est souvent lente, demande du temps et peut être sujette à des erreurs.

L'absence d'un système centralisé et automatisé rend la gestion de ces documents difficile : manque de traçabilité, lourde charge administrative et risque élevé de fraude. Cela affecte aussi bien les universités que les entreprises.

Face à ces limites, une question se pose : Comment garantir l'authenticité, le suivi et la vérification des attestations de stage tout en réduisant les efforts manuels ?

Dans ce cadre, la technologie blockchain, grâce à sa structure décentralisée, sécurisée et transparente, apparaît comme une solution pertinente pour automatiser la génération, la validation et la consultation des attestations de manière fiable.

1.4 Étude de l'existant

1.4.1 Solutions existantes

En Tunisie, la gestion des attestations de stage repose essentiellement sur des procédures classiques. Les entreprises d'accueil génèrent des documents au format papier ou numérique à l'issue du stage, qui sont ensuite remis aux étudiants pour transmission à leurs établissements. La validation repose souvent sur des échanges par courrier électronique ou sur des remises physiques, impliquant une intervention manuelle de l'administration universitaire.

Certaines institutions universitaires ont mis en place des systèmes numériques internes pour faciliter le suivi des stages, notamment pour l'affectation des encadrants ou la validation académique. Ces plateformes permettent généralement aux étudiants de déposer leurs rapports de stage et, dans certains cas, de télécharger des attestations validées. Cependant, ces systèmes restent isolés, souvent non interopérables avec des acteurs externes, et ne prévoient pas de mécanisme formel permettant une vérification automatisée de l'authenticité des documents émis.

À l'échelle nationale, aucun cadre standardisé n'existe pour garantir la sécurité, la traçabilité

ou l'accès tiers aux attestations. Cela engendre une hétérogénéité dans les pratiques, une charge administrative importante pour les personnels impliqués, et une difficulté pour les tiers à valider l'authenticité des documents reçus.

1.4.2 Limites identifiées

L'analyse des pratiques actuelles fait ressortir plusieurs limites importantes :

- L'absence de vérification automatisée de l'authenticité des attestations ;
- La dépendance à des échanges manuels ou par courriel, générateurs de délais ;
- Le risque de falsification des documents numériques transmis en dehors de tout système sécurisé ;
- L'inexistence d'un accès en ligne aux attestations pour des acteurs tiers autorisés (recruteurs, institutions partenaires) ;
- Le manque de standardisation entre les établissements ;
- L'impossibilité de tracer l'historique des validations de manière infalsifiable.

Ces constats soulignent la nécessité de mettre en œuvre une solution moderne, interopérable et fiable, capable de répondre aux exigences de sécurité, d'intégrité et de simplicité de vérification.

1.5 Principe et fonctionnement de la technologie blockchain

1.5.1 Principe de la blockchain

La blockchain est une technologie de registre distribué permettant de stocker et transmettre des données de manière sécurisée, transparente et sans dépendance à un tiers de confiance. Chaque bloc contient des données horodatées, validées par un consensus, et liées cryptographiquement au bloc précédent, formant une chaîne inaltérable [1].

Une fois qu'une donnée est enregistrée dans la blockchain, elle devient pratiquement immuable : toute tentative de modification nécessiterait de réécrire tous les blocs suivants, ce qui est techniquement irréalisable dans un système bien sécurisé [2]. Cette immuabilité garantit l'intégrité des données et constitue un pilier central de la fiabilité de cette technologie.

1.5.2 Fonctionnement appliqué à la certification des documents

Dans le contexte de la certification de documents, la blockchain est utilisée pour inscrire une empreinte numérique (appelée hachage) d'un document. Cette empreinte, unique et générée à partir du contenu du fichier, est enregistrée dans la blockchain, accompagnée d'un horodatage et d'une signature cryptographique de l'émetteur [3].

Ce procédé permet, sans révéler le contenu du document, de vérifier son authenticité. Toute personne disposant du fichier peut en recalculer le hachage et le comparer à celui inscrit dans la blockchain. Si les deux correspondent, cela prouve que le document est authentique et n'a pas été modifié depuis son enregistrement [4].

Cette approche offre plusieurs avantages :

- *Intégrité* : toute altération modifie l'empreinte et rend la vérification invalide ;
- *Authenticité* : la signature cryptographique permet d'identifier l'émetteur ;
- *Traçabilité* : chaque action est enregistrée de manière permanente ;
- *Accessibilité* : la vérification peut se faire à distance via une interface web ;
- *Confidentialité* : seuls les hachages sont visibles, le contenu reste privé [5].

Cette méthode s'impose donc comme une réponse fiable aux problèmes de fraude, de lenteur de traitement et de vérification des documents numériques.

1.6 Solution proposée

1.6.1 Description générale de la solution

La solution proposée repose sur une plateforme web permettant de gérer tout le cycle de vie des stages : de l'inscription jusqu'à la génération et vérification des attestations. Elle intègre des fonctionnalités adaptées aux différents profils utilisateurs (étudiants, encadrants, responsables, tiers) avec une logique de validation progressive.

1.6.2 Principe de fonctionnement

Le processus de fonctionnement peut être résumé selon les étapes suivantes :

1. **Inscription et connexion avec MetaMask** : l'utilisateur se connecte via son portefeuille MetaMask. Une signature permet d'authentifier son identité sans mot de passe, selon les rôles enregistrés dans un contrat intelligent.
2. **Proposition de sujet de stage** : les étudiants peuvent soumettre un ou plusieurs sujets via leur tableau de bord. Un e-mail est automatiquement envoyé aux deux encadrants (universitaire et professionnel), qui peuvent ensuite accepter ou refuser la proposition directement depuis leur interface respective.
3. **Création du stage** : si les deux encadrants valident le sujet, le stage est officiellement créé et rattaché à l'étudiant dans le système. Le statut de chaque proposition est mis à jour et suivi en temps réel.
4. **Soumission du rapport** : à la fin du stage, l'étudiant soumet son rapport via l'interface. Chaque encadrant reçoit une notification et peut valider ou rejeter le contenu. Des rappels automatiques sont envoyés si aucun retour n'est effectué dans un délai donné.
5. **Déblocage par un tiers** : en cas de blocage (absence de validation), un utilisateur tiers par entité est informé de retard et peut intervenir et valider le rapport, garantissant ainsi la fluidité du processus sans intervention manuelle complexe.
6. **Génération de l'attestation** : une fois le rapport validé, l'attestation est générée au format PDF. Elle contient un QR code pointant vers une page de vérification en ligne.
7. **Certification sur la blockchain** : une empreinte (hash) de l'attestation est calculée et inscrite dans la blockchain via un contrat intelligent. Cela permet une vérification ultérieure de son intégrité.
8. **Vérification publique** : tout tiers (recruteur, institution) peut vérifier l'attestation depuis une page dédiée, en important le PDF ou en scannant le QR code. La correspondance entre l'empreinte et celle enregistrée sur la blockchain permet de valider l'authenticité du document.
9. **Gestion de l'historique et des notifications** : toutes les actions (proposition, validation, refus, génération, déblocage) sont historisées. Des e-mails sont envoyés à chaque étape clé, assurant un suivi transparent et automatisé pour tous les utilisateurs.

1.7 Étude comparative des solutions blockchain

1.7.1 Ethereum

Ethereum est une blockchain publique décentralisée qui permet l'exécution de contrats intelligents. Contrairement à Bitcoin, qui est uniquement centré sur les transactions financières, Ethereum permet l'exécution de programmes appelés "smart contracts". Ces programmes s'exécutent automatiquement quand certaines conditions sont remplies [6].

L'un des principaux avantages d'Ethereum réside dans sa décentralisation, permettant à quiconque de participer au réseau de manière sécurisée et transparente. Grâce à cette structure, Ethereum est devenu un standard de l'industrie pour le déploiement de contrats intelligents.

1.7.2 EVM (Ethereum Virtual Machine)

L'Ethereum Virtual Machine (EVM) est l'environnement d'exécution qui permet l'exécution des contrats intelligents sur la blockchain Ethereum. L'EVM est un moteur de calcul décentralisé, ce qui permet de garantir la sécurité et l'intégrité des transactions sans avoir besoin d'une autorité centrale [7].

L'EVM est compatible avec plusieurs autres blockchains telles que Polygon, Binance Smart Chain et Avalanche, ce qui permet l'exécution de contrats intelligents sur plusieurs réseaux tout en utilisant le même code.

1.7.3 Réseaux compatibles avec l'EVM

Le choix d'un réseau compatible avec l'EVM est essentiel pour assurer l'interopérabilité des applications décentralisées (dApps). Ethereum est bien entendu le réseau de référence, mais d'autres réseaux comme Polygon, Binance Smart Chain (BSC) et Avalanche sont également compatibles avec l'EVM. Ces réseaux permettent de déployer des contrats intelligents avec des frais plus faibles et des vitesses de traitement plus rapides [6].

Blockchain	**Type**	**Compatibilité EVM**	**Documentation**
Ethereum	Public	Oui	Très riche [6]
Polygon	Public	Oui	Bonne [7]
BNB Chain	Public	Oui	Moyenne [7]
Solana	Public	Non	Moyenne [8]
Avalanche (C-Chain)	Public	Oui	Bonne [7]

TABLE 1 – Comparaison des blockchains compatibles EVM

1.7.4 Langages de développement des contrats intelligents

Le langage principal pour écrire des contrats intelligents sur Ethereum et ses réseaux compatibles est Solidity. Ce langage est bien documenté et utilisé par une large communauté de développeurs [9]. D'autres blockchains comme Solana utilisent Rust, un langage plus complexe mais performant, tandis qu'Aptos et Sui préfèrent utiliser Move, un langage encore émergent [9][10].

Blockchain	**Langage principal**	**Courbe d'apprentissage**	**Écosystème d'outils**
Ethereum / EVM	Solidity	Moyenne	Très développé [9]
Solana	Rust	Difficile	Modéré [8]
Aptos / Sui	Move	Difficile	Faible [10]

TABLE 2 – Langages pour le développement de smart contracts

1.7.5 Environnements de test : testnets vs simulateurs locaux

Les testnets comme Sepolia ou Fuji offrent des environnements réels pour tester les contrats intelligents. Cependant, ces réseaux présentent des **frais de gaz élevés** et un **accès limité aux tokens de test**. De plus, certains testnets comme Goerli sont désormais dépréciés, rendant leur usage plus difficile [11][12].

Les simulateurs locaux comme **Hardhat Network** permettent de tester des contrats intelligents rapidement, sans frais et sans dépendre d'un réseau externe. Cela est particulièrement utile pendant la phase de développement et de tests des prototypes [14].

Environnement	**Type**	**Frais de gaz**	**Accès aux jetons**	**Stabilité**
Sepolia (Ethereum)	Testnet public	Élevés [11]	Faucets limités	Faible
Mumbai (Polygon)	Testnet public	Variables [12]	Faucets limités	Moyenne
Hardhat Network	Simulateur local	Aucun [14]	Non requis	Très stable

TABLE 3 – Comparaison des environnements de test

1.7.6 Outils de développement

Parmi les outils de développement, **Hardhat** est le plus utilisé pour les projets nécessitant une simulation locale d'Ethereum. Il est flexible, rapide et offre une intégration complète avec Solidity. **Truffle** et **Remix IDE** sont également populaires, mais ils présentent des limitations en termes de flexibilité et de performance par rapport à Hardhat [14][15][16].

Outil	**Type**	**Langage supporté**	**Avantages**	**Limites**
Hardhat	Framework CLI	Solidity	Simulateur local, rapidité, plugins [14]	Courbe d'apprentissage modérée
Truffle + Ganache	Framework CLI	Solidity	Interface graphique, tests automatisés [15]	Moins flexible
Remix IDE	IDE Web	Solidity	Facilité d'utilisation, sans installation [16]	Déploiement limité aux tests simples
Foundry	CLI avancée	Solidity / Rust	Très rapide, tests avancés [17]	Pour utilisateurs experts

TABLE 4 – Outils de développement pour smart contracts

1.7.7 Justification des choix pour le projet

Les choix suivants ont été retenus pour garantir une solution optimale et évolutive :

– **Réseau** : Ethereum, pour sa large adoption, sa stabilité et sa compatibilité avec l'EVM [6] ;
– **Langage** : Solidity, pour sa simplicité, sa documentation et sa compatibilité avec l'EVM [9] ;
– **Environnement de test** : Hardhat Network, pour sa rapidité, son simulateur local et l'absence de frais [14] ;
– **Outil principal** : Hardhat, pour sa flexibilité, son intégration avec les bibliothèques nécessaires et son environnement de développement complet [14].

1.8 Comparaison des frameworks Backend

Le backend d'une application blockchain gère les transactions, les connexions simultanées et les interactions avec la blockchain. Pour effectuer cette comparaison, les critères suivants ont été utilisés :

— Performance : Capacité à traiter de nombreuses connexions simultanées, essentielle pour les applications blockchain.
— Facilité d'intégration avec la blockchain : Compatibilité avec des bibliothèques comme Web3.js et Ethers.js pour interagir avec des contrats intelligents.
— Simplicité et flexibilité : Rapidité de mise en œuvre et souplesse dans le développement d'applications évolutives.
— Communauté et écosystème : Large communauté et support actif, facilitant le développement et la résolution des problèmes.

Framework	Langage	Performance	Facilité d'intégration avec blockchain
Node.js	JavaScript	Très élevée	Excellente [14]
Express.js	JavaScript	Élevée	Excellente [15]
Django	Python	Moyenne	Bonne [16]
Spring Boot	Java	Très élevée	Moyenne [17]

TABLE 5 – Comparaison des frameworks backend pour blockchain

1.8.1 Justification du choix Backend

Le choix de Node.js comme framework principal pour ce projet est justifié par : - Sa performance élevée et sa capacité à gérer efficacement de nombreuses connexions simultanées, cruciales pour les applications blockchain [14]. - L'écosystème riche de modules npm qui permet une intégration fluide avec Web3.js et Ethers.js pour interagir avec les contrats intelligents Ethereum. - La simplicité et flexibilité qui facilitent le développement rapide et l'adaptation aux besoins du projet blockchain.

—

1.9 Comparaison des frameworks Frontend

Le frontend est l'interface qui permet aux utilisateurs d'interagir avec l'application blockchain. Les critères de comparaison pour les frameworks frontend sont les suivants :

— Courbe d'apprentissage : Facilité d'apprentissage et de mise en œuvre du framework.

— Performance : Vitesse et fluidité de l'interface utilisateur, surtout pour les applications décentralisées (dApps).

— Facilité d'intégration avec la blockchain : Support des bibliothèques nécessaires pour interagir avec les contrats intelligents, comme Web3.js et Ethers.js.

— Communauté et écosystème : Large communauté de développeurs et un bon support, facilitant la résolution des problèmes lors du développement.

Framework	Courbe d'apprentissage	Performance	Facilité d'intégration avec blockchain
React.js	Moyenne	Élevée	Excellente [18]
Vue.js	Faible	Moyenne	Bonne [19]
Angular	Élevée	Très élevée	Bonne [19]
Svelte	Faible	Très élevée	Excellente [20]

TABLE 6 – Comparaison des frameworks frontend pour blockchain

1.9.1 Justification du choix Frontend

Le choix de React.js pour le frontend est basé sur les raisons suivantes : - Excellente performance grâce à l'utilisation du DOM virtuel, ce qui permet de créer des applications réactives avec de très bonnes performances [18]. - Facilité d'intégration avec des bibliothèques comme Web3.js et Ethers.js, nécessaires pour interagir directement avec la blockchain Ethereum. - Large communauté et écosystème, ce qui permet un développement rapide et l'accès à une multitude de ressources et de solutions.

1.10 Conclusion

Ce premier chapitre a permis de poser le cadre du projet, d'identifier les solutions blockchain adaptées et d'explorer les choix technologiques. Après avoir comparé différentes blockchains, frameworks backend et frontend, nous avons retenu **Ethereum** avec **Solidity** pour les contrats intelligents et **Hardhat** pour l'environnement de développement et de test. Ces choix sont motivés par la performance, la compatibilité et la richesse de l'écosystème qui assurent la robustesse de la solution proposée.

Le chapitre suivant s'intéressera à l'analyse des besoins fonctionnels et non fonctionnels, et présentera la conception détaillée de l'architecture du système.

Chapitre 2

Analyse et conception

2.1 Introduction

Ce chapitre vise à présenter l'analyse détaillée du système à mettre en place pour la gestion et la délivrance des attestations de stage via la technologie blockchain. Il comprend l'identification des besoins fonctionnels et non fonctionnels, la définition de l'architecture globale du système, ainsi que la modélisation UML des différentes entités et processus métier impliqués. Cette phase constitue une étape essentielle garantissant la cohérence et la fiabilité de l'application.

2.2 Analyse des besoins

2.2.1 Besoins fonctionnels

Afin de répondre aux objectifs du projet, le système doit permettre la réalisation d'un ensemble de fonctionnalités indispensables au bon déroulement du processus de gestion des stages et de délivrance des attestations. Les principales fonctionnalités identifiées sont les suivantes :

— Permettre aux étudiants de proposer un sujet de stage, de déposer leur rapport, et de recevoir une attestation validée.

— Permettre aux encadrants académique et professionnel de valider ou refuser les propositions de sujet, puis d'évaluer les rapports déposés.

— Permettre aux responsables universitaires de consulter les attestations et valider les stages.

— Permettre aux responsables en entreprise de générer l'attestation du stagiaire.

— Permettre aux tiers débloqueurs d'intervenir dans des cas bloqués (absence de validation).

— Offrir un service de vérification publique d'attestation via un QR code sécurisé.

— Notifier par mail chaque acteur des actions requises.

2.2.2 Besoins non fonctionnels

En plus des fonctionnalités attendues, la solution doit respecter un certain nombre de contraintes non fonctionnelles assurant le bon fonctionnement, la fiabilité et la sécurité du système :

— **Sécurité** : les utilisateurs sont authentifiés via leurs adresses Ethereum à travers MetaMask. Les attestations sont enregistrées sur la blockchain locale (Hardhat) sous forme de hachage, assurant leur intégrité et empêchant toute falsification.

— **Confidentialité contrôlée** : les données internes telles que les rapports, commentaires et évaluations sont strictement réservées aux utilisateurs concernés, selon leurs rôles. En revanche, les attestations validées sont volontairement rendues accessibles via un QR code public, afin de permettre une vérification fiable et transparente par tout tiers (recruteur, organisme, etc.).

— **Accessibilité** : la plateforme est accessible via une interface web simple, nécessitant uniquement MetaMask. Le système repose sur des API REST entre le frontend React, le backend Node.js et la base de données MySQL.

— **Traçabilité** : toutes les actions importantes sont enregistrées dans une table d'historique (avec rôle, date et type d'action), facilitant le suivi des opérations et l'intervention en cas de blocage.

— **Gratuité des technologies** : toutes les solutions techniques adoptées (React, Node.js, Hardhat, IPFS Desktop, ngrok, etc.) sont gratuites et open-source, conformément aux exigences.

2.3 Architecture globale du système

Le système repose sur une architecture client-serveur étendue par une couche blockchain locale et un système de stockage distribué (IPFS). Le frontend, développé en React.js, permet l'interaction utilisateur via MetaMask. Le backend, en Node.js, orchestre les échanges avec la base de données MySQL, les contrats intelligents Solidity (déployés en local via Hardhat) et le stockage IPFS. Il gère également l'envoi de notifications email via SMTP.

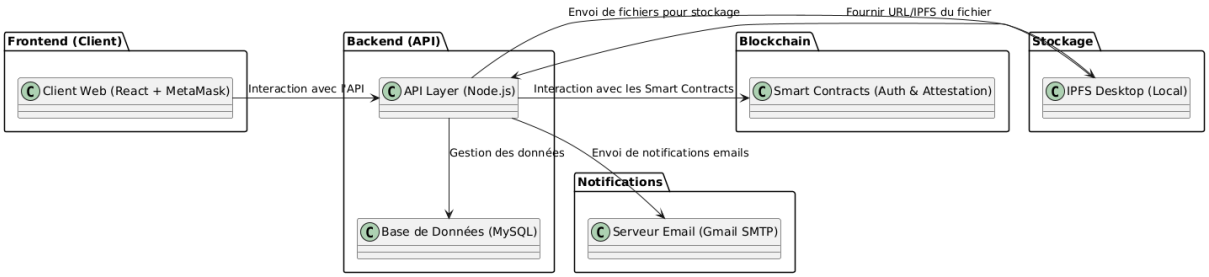

FIGURE 3 – Architecture globale de la plateforme StageChain

2.4 Modélisation UML

2.4.1 Diagramme de classes UML relationnel

Ce diagramme représente toutes les entités relationnelles présentes dans la base ainsi que les relations entre elles.

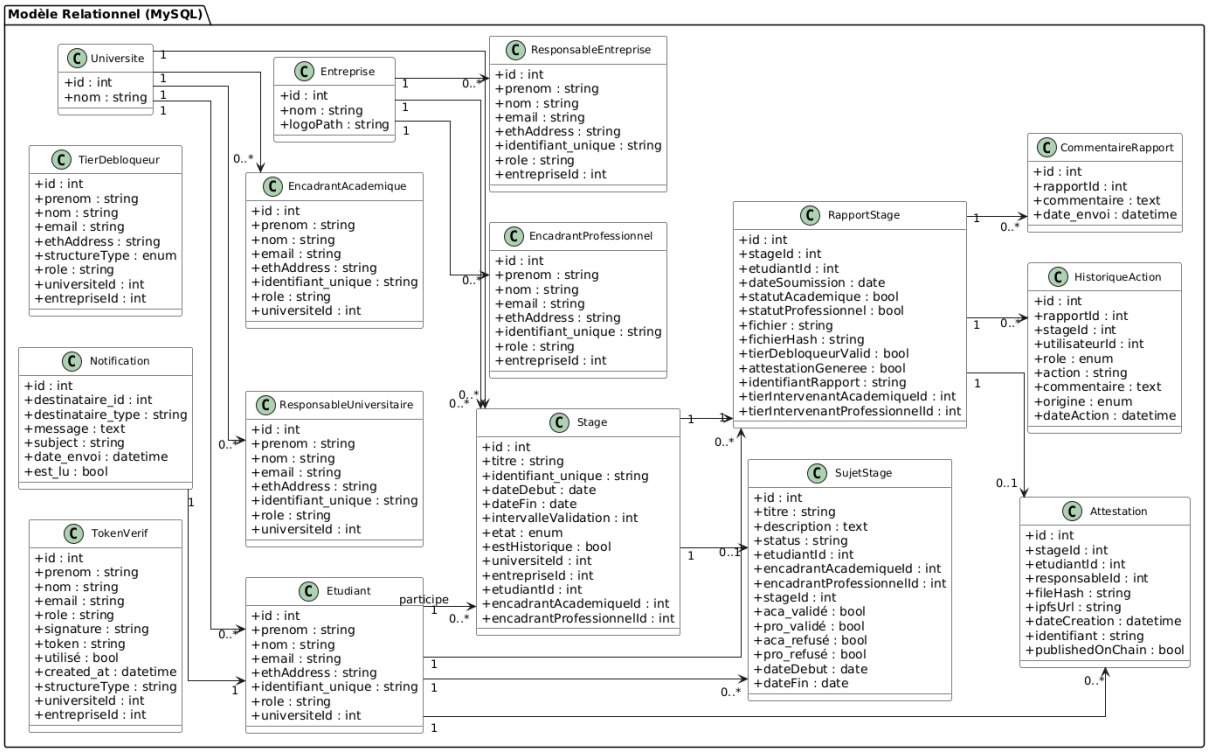

FIGURE 4 – Diagramme de classes de la base de données relationnelle

2.4.2 Diagrammes de cas d'utilisation

Acteur : Étudiant

Le diagramme suivant présente les interactions fonctionnelles d'un étudiant avec la plateforme. L'étudiant peut proposer un sujet de stage, soumettre un rapport, consulter l'état d'avancement, recevoir des notifications et vérifier son attestation générée.

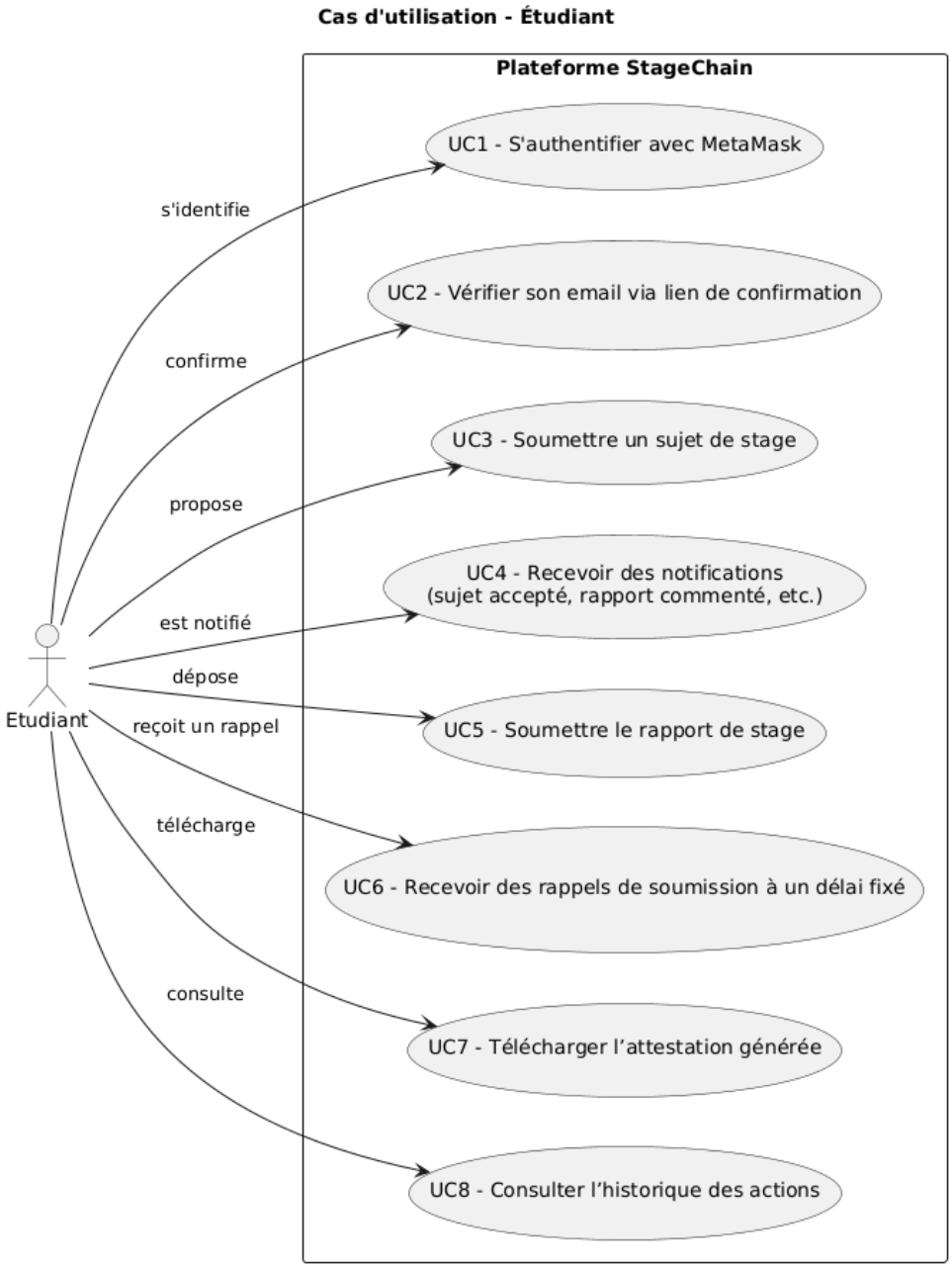

FIGURE 5 – Cas d'utilisation – Étudiant

Acteurs : Encadrants

Ce diagramme modélise les actions communes des encadrants académique et professionnel. Ils peuvent valider ou refuser un sujet proposé, évaluer les rapports de stage, commenter et notifier l'étudiant, contribuant ainsi à l'encadrement et à l'évaluation du stage.

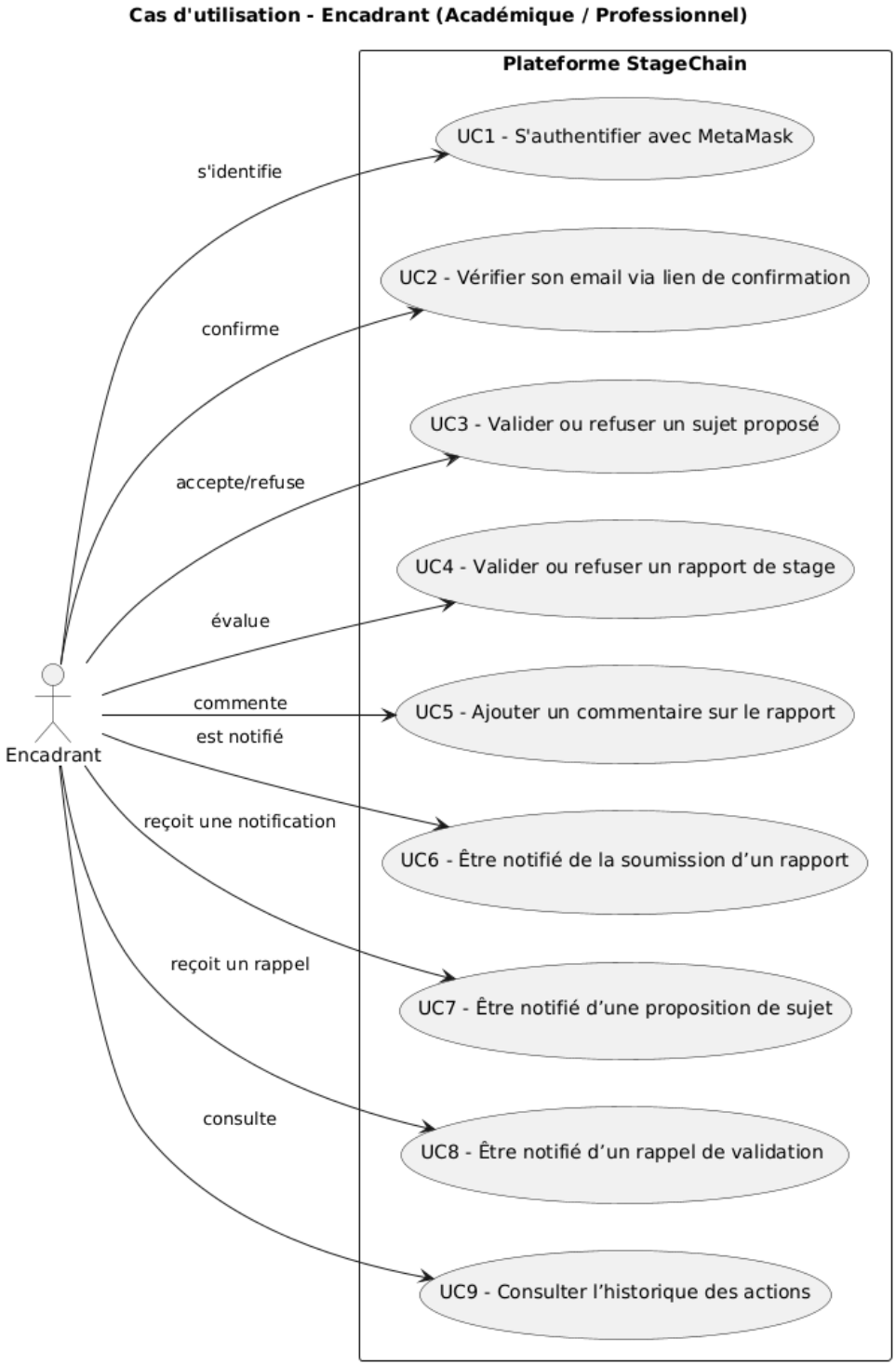

FIGURE 6 – Cas d'utilisation – Encadrants académique et professionnel

Acteur : Responsable Universitaire

Le responsable universitaire intervient principalement dans la validation finale du stage après consultation d'attestation reçue et la consultation des données liées aux étudiants et encadrants de son université. Ce diagramme montre les cas d'utilisation qui lui sont réservés.

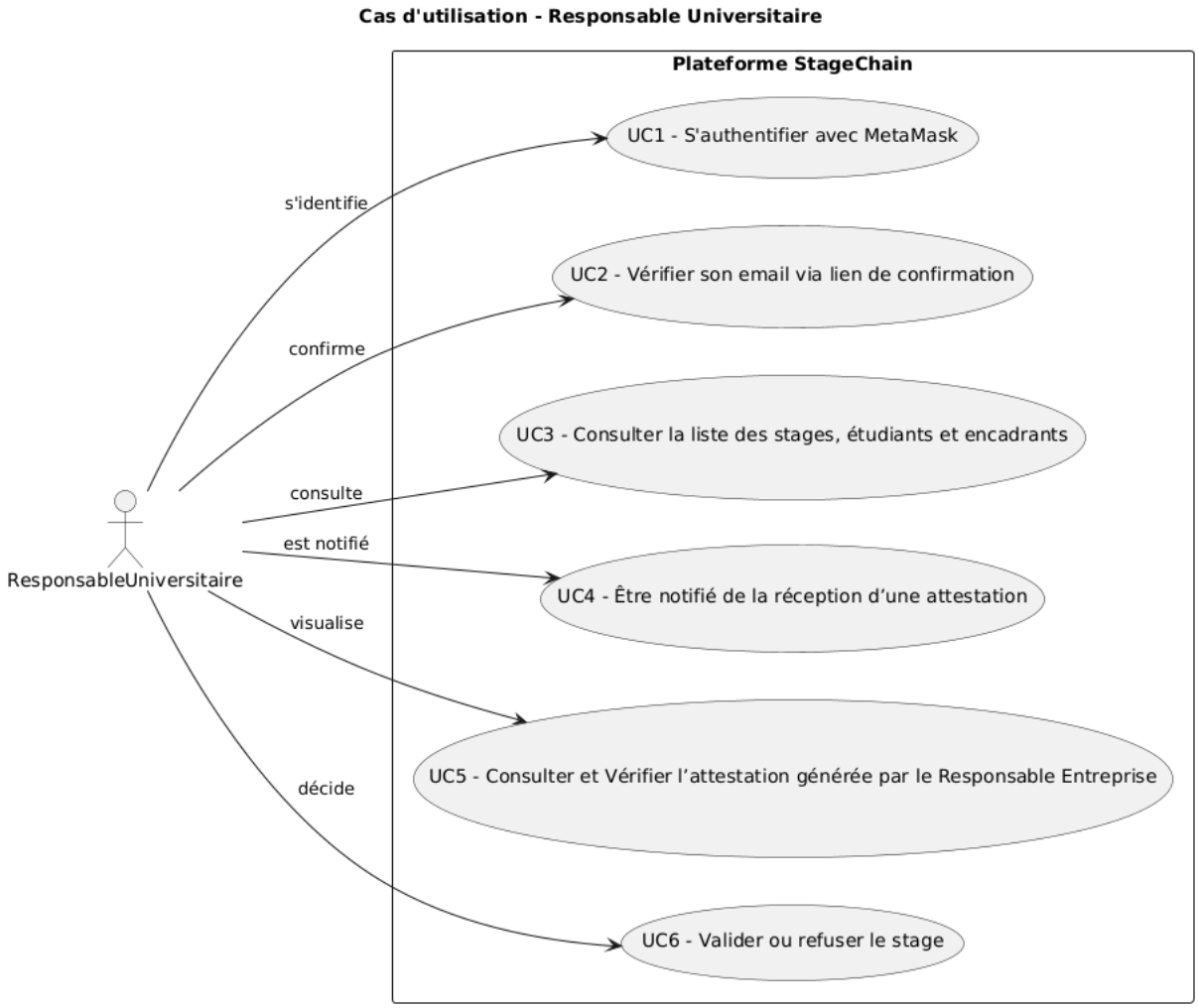

FIGURE 7 – Cas d'utilisation – Responsable universitaire

Acteur : Responsable Entreprise

Ce diagramme présente les interactions du responsable d'entreprise, qui est chargé de générer l'attestation de stage une fois le rapport évalué. Il peut aussi consulter les stagiaires et encadrants de son entreprise.

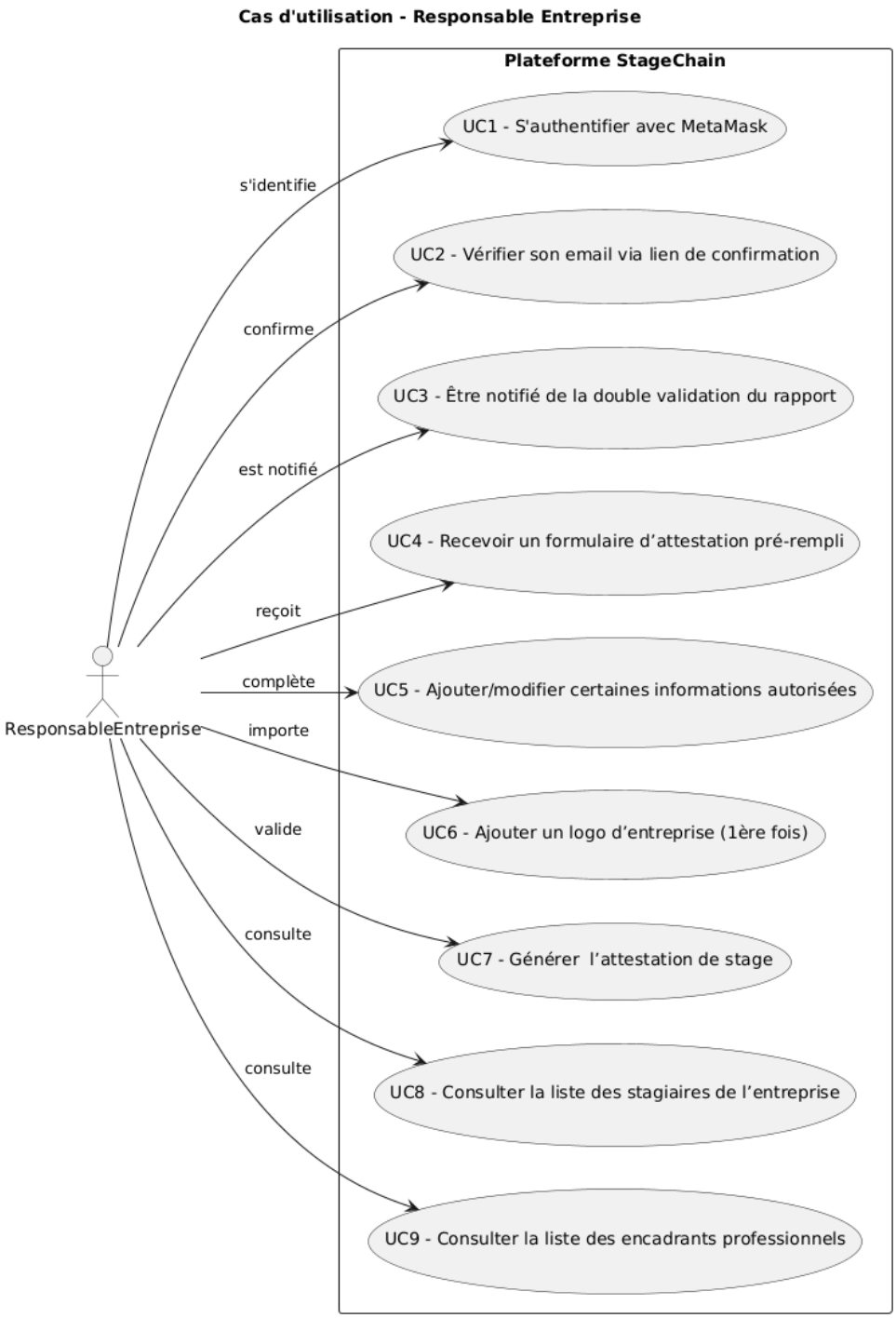

FIGURE 8 – Cas d'utilisation – Responsable entreprise

Acteur : Tiers Débloqueur

En cas de blocage du processus (non validation du rapport dans les délais), un tiers débloqueur désigné peut intervenir . Ce diagramme illustre les scénarios où son rôle est sollicité.

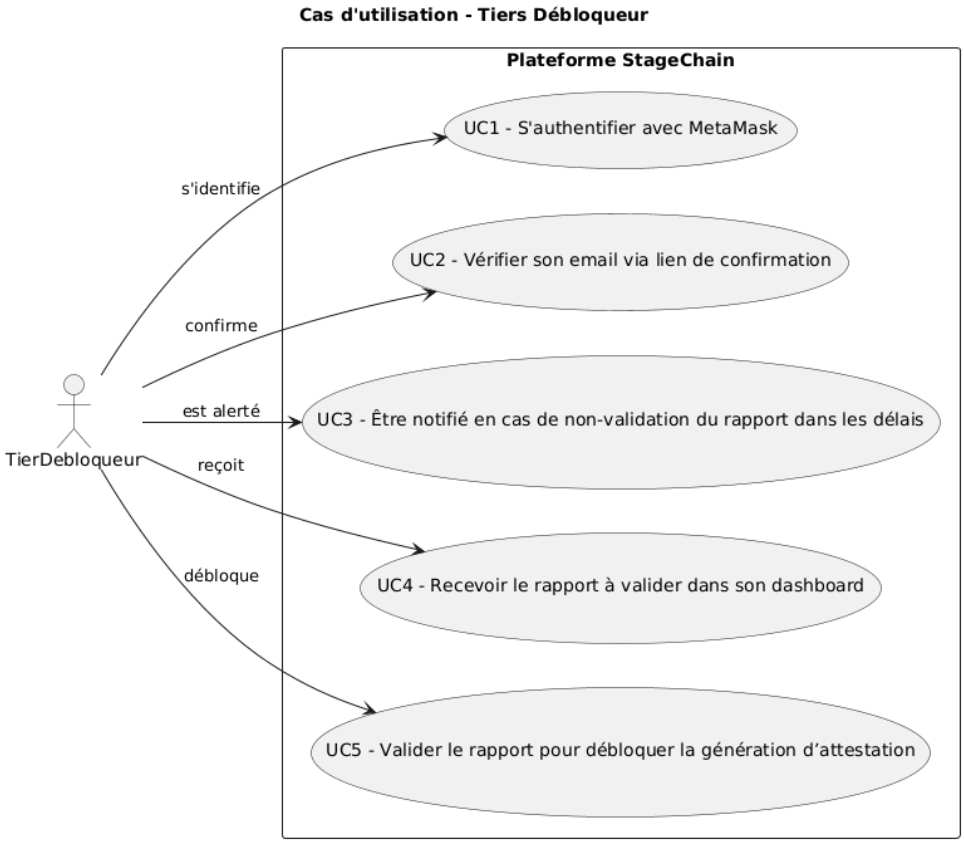

FIGURE 9 – Cas d'utilisation – Tiers débloqueur

Acteur : Vérificateur Public

Ce diagramme décrit l'unique interaction d'un acteur externe (recruteur, établissement, etc.) avec la plateforme : la vérification d'une attestation via un QR code public. Il accède à une page dédiée contenant le fichier PDF et les métadonnées.

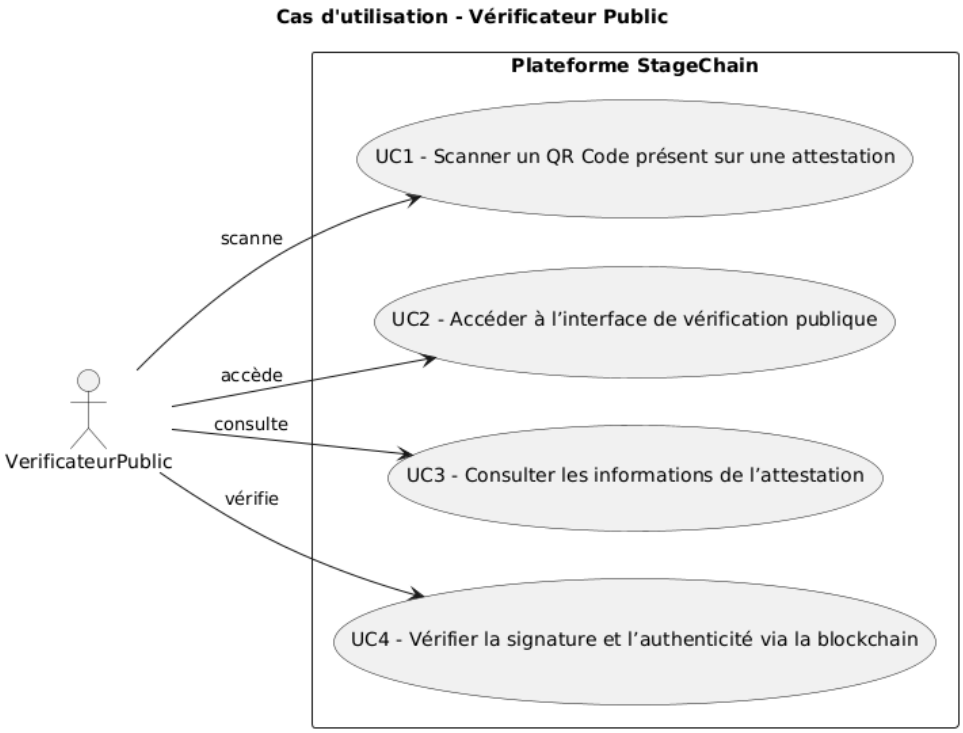

FIGURE 10 – Cas d'utilisation – Vérificateur public via QR code

2.4.3 Diagrammes de séquence

Les diagrammes de séquence suivants illustrent les interactions entre les différents acteurs et composants du système à travers les principales étapes du processus. Ils ont été divisés en trois séquences distinctes : proposition du sujet, dépôt et validation du rapport, génération de l'attestation et validation de stage.

Séquence 1 : Proposition du sujet de stage

Ce diagramme illustre la séquence d'interactions lors de la proposition d'un sujet de stage par l'étudiant. Il montre comment les encadrants interviennent pour valider ou refuser cette proposition, et comment cela déclenche la création du stage.

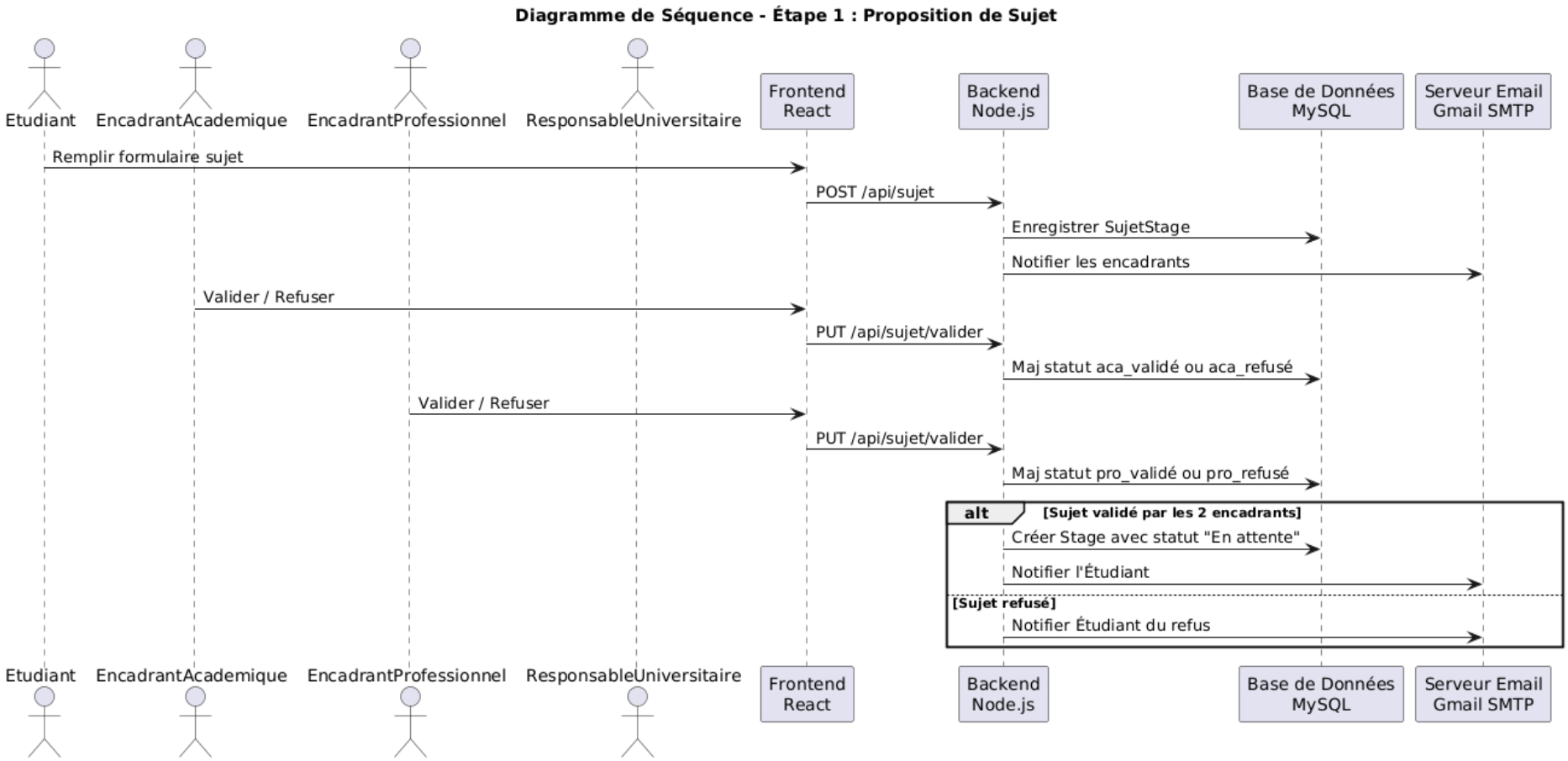

FIGURE 11 – Diagramme de séquence – Proposition du sujet de stage

Séquence 2 : Dépôt et validation du rapport

Ce diagramme décrit le processus de dépôt du rapport de stage par l'étudiant. Il détaille ensuite les validations successives par les encadrants, accompagnées éventuellement de commentaires ou de demandes de correction et l'intervention des tiers (un tier par entité selon l'encadrant qui a fait le retard) en cas de dépassement des délais de validation.

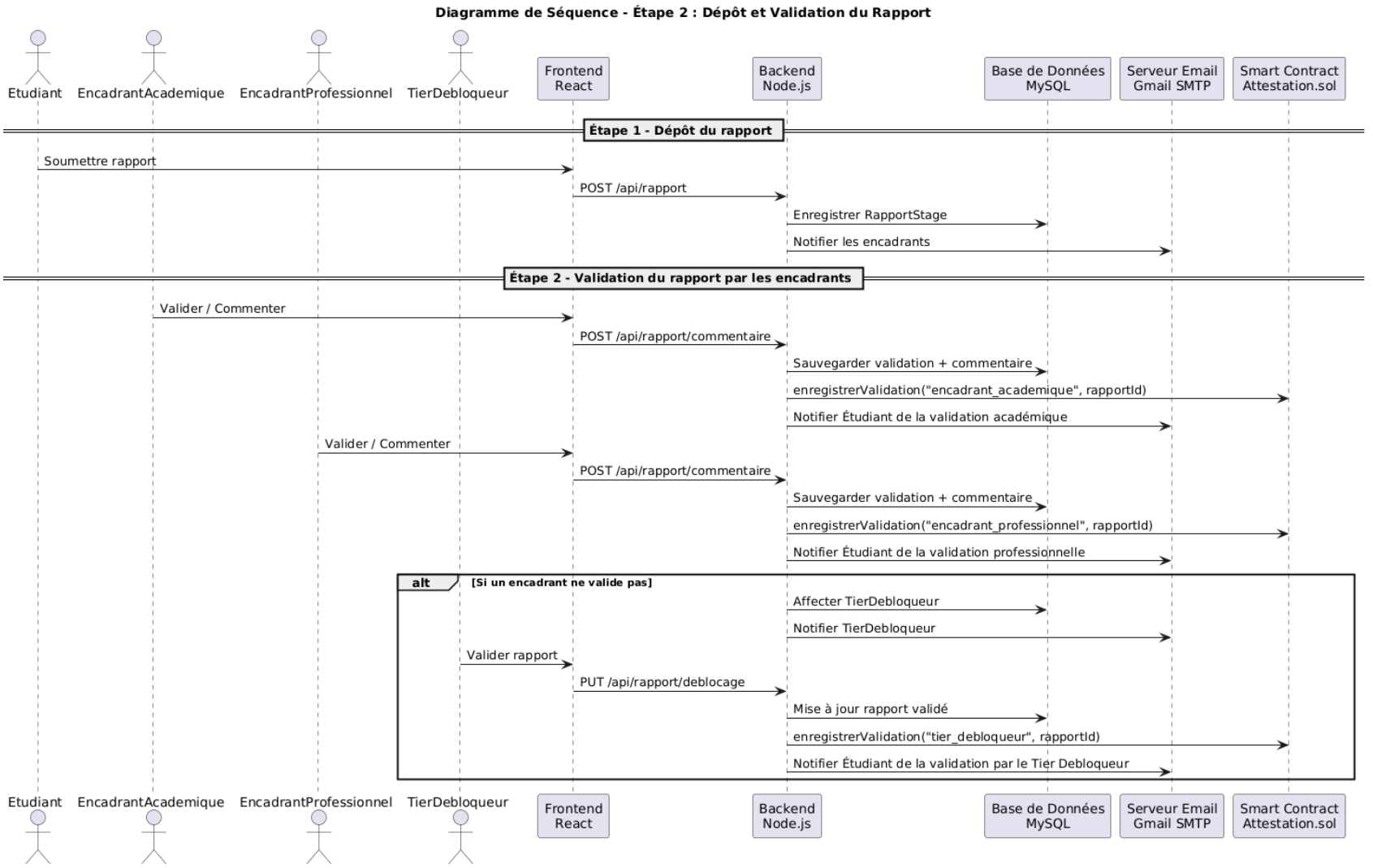

FIGURE 12 – Diagramme de séquence – Dépôt et validation du rapport

Séquence 3 : Génération de l'attestatione et validation de stage

Ce diagramme modélise la phase finale du processus : après double validation du rapport, le responsable entreprise génère une attestation. Celle-ci est publiée sur la blockchain et rendue vérifiable publiquement via un QR code et envoyée à l'étudiant et au responsable universitaire.

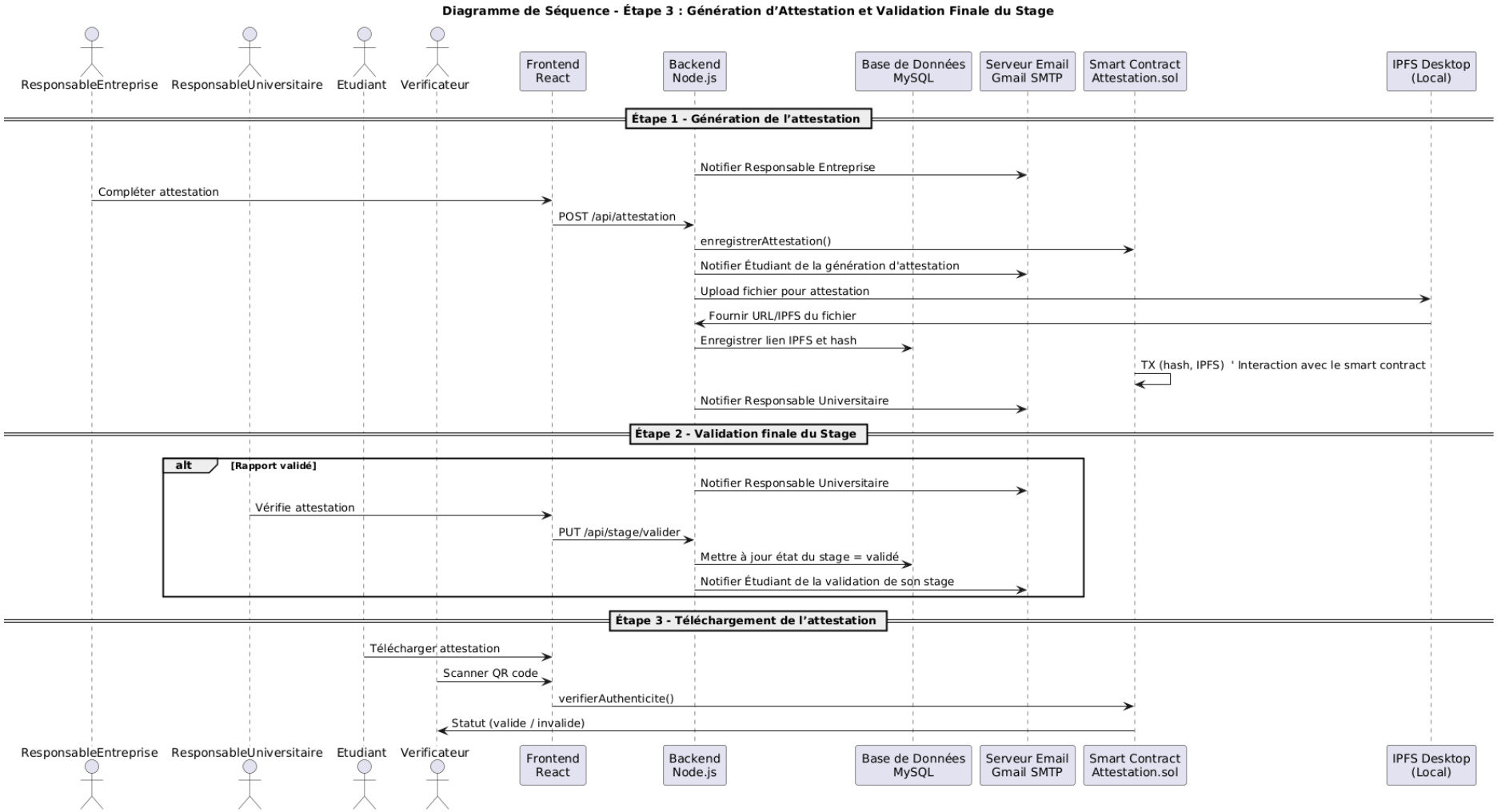

FIGURE 13 – Diagramme de séquence – Validation du stage et génération de l'attestation

2.4.4 Diagramme de déploiement

Ce diagramme présente l'organisation des composants et leur interaction.

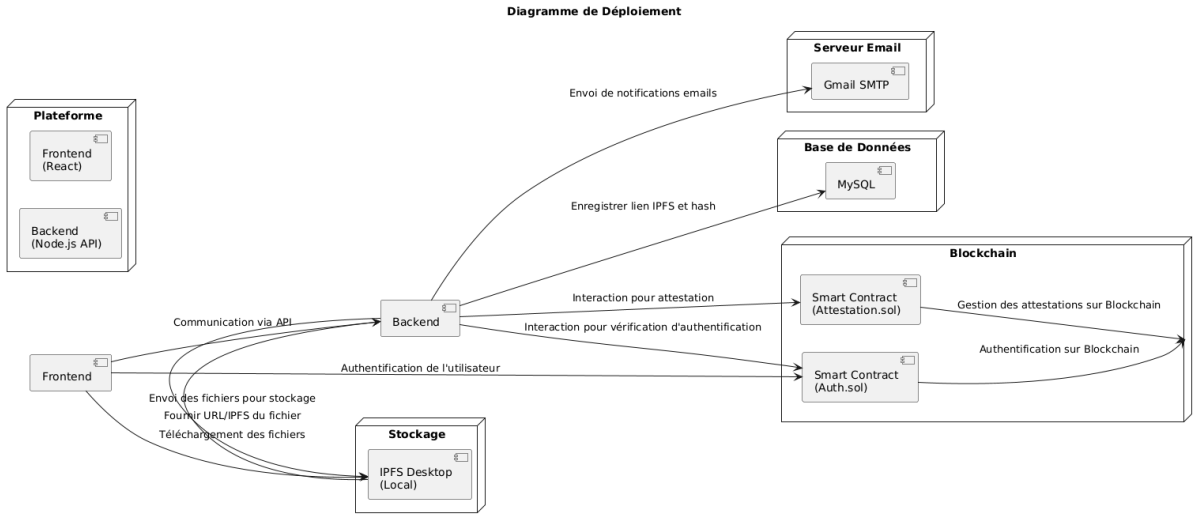

FIGURE 14 – Diagramme de déploiement de la solution StageChain

2.5 Conclusion

Ce chapitre a présenté l'analyse des besoins du système, son architecture globale, ainsi que la modélisation UML à travers des diagrammes représentatifs. L'ensemble de ces éléments constitue la base conceptuelle nécessaire à la phase de réalisation technique qui suit.

Chapitre 3

Réalisation

3.1 Introduction

Ce chapitre présente la mise en œuvre pratique de la solution StageChain. Il détaille les technologies utilisées, le développement des différentes fonctionnalités, ainsi que l'intégration des smart contracts, du stockage IPFS et des services web.

3.2 Mise en place de l'environnement de développement

3.2.1 Environnement matériel

Le développement de la plateforme a été réalisé sur un poste de travail dont les caractéristiques sont les suivant :

Nom du périphérique	DESKTOP-80LN4LL
Processeur	Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz 2.90 GHz
Mémoire RAM installée	8,00 Go (7,85 Go utilisable)
Carte graphique	Intel(R) HD Graphics 620 (128 MB)
Stockage	233 GB SSD Samsung SSD 860 EVO M.2 250GB
ID de périphérique	F604FBEC-0229-49E9-BD7B-4F1FBA047E72
ID de produit	00330-50015-90972-AAOEM
Type du système	Système d'exploitation 64 bits, processeur x64
Stylet et fonction tactile	La fonctionnalité d'entrée tactile ou avec un stylet n'est pas disponible sur cet écran

FIGURE 15 – caractéristiques du poste de travail

3.2.2 Environnement logiciel

Node.js

Environnement d'exécution JavaScript open source et multiplateforme qui fournit un ensemble de primitives d'E/S asynchrones dans sa bibliothèque standard [21].

FIGURE 16 – Logo de Node.js

Hardhat

Environnement de développement flexible et extensible pour les logiciels Ethereum. Il vous permet d'écrire, de tester, de déboguer et de déployer facilement les contrats intelligents.[22]

FIGURE 17 – Logo de l'environnement Hardhat

React.js

Framework JavaScript pour la création d'interfaces utilisateur interactives.[23]

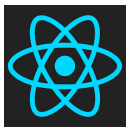

FIGURE 18 – Logo de l'environnement Frontend React

MySQL

Système de gestion de base de données relationnelle open source utilisé pour le stockage des données métier.[24]

FIGURE 19 – Logo de mysql

IPFS Desktop pour le stockage décentralisé

Application de bureau permettant d'exécuter un nœud IPFS local pour le stockage décentralisé des attestations .[25]

FIGURE 20 – Logo de IPFS

ngrok pour l'exposition du backend

Outil de tunneling sécurisé permettant d'exposer le serveur local à Internet pour les tests d'intégration.[26]

FIGURE 21 – Logo de Ngrok

3.2.3 Outils et extensions utilisés

MetaMask

extension de navigateur populaire et établie qui fonctionne comme un portefeuille de crypto-monnaies.[27]

FIGURE 22 – Logo de Metamask Wallet

Postman

Postman est une plateforme API tout-en-un pour tester les API.[28]

FIGURE 23 – Logo de Postman

Git et GitHub

GitHub est une plateforme basée sur le cloud pour stocker, partager et travailler avec d'autres pour écrire du code.[29]

Git est un système de contrôle de version qui effectue intelligemment le suivi des modifications apportées aux fichiers.[29]

FIGURE 24 – Logos des Git et Github

Visual Studio Code

Editeur de code source et un environnement de développement intégré (IDE).[30]

FIGURE 25 – Logo Vscode

3.3 Développement du backend

3.3.1 API REST pour la gestion des utilisateurs

3.3.2 Intégration avec MetaMask

3.3.3 Parsing CSV pour l'importation des universités et entreprises

3.4 Développement du frontend

3.4.1 Interfaces graphiques selon les rôles

Page d'accueil

Formulaire d'inscription avec MetaMask

Page de connexion avec MetaMask

Dashboard étudiant

Dashboard encadrant académique

Dashboard encadrant professionnel

Dashboard responsable universitaire

Dashboard responsable entreprise

Dashboard tiers débloqueur universitaire

Dashboard tiers débloqueur entreprise

3.5 Développement des contrats intelligents

3.5.1 Contrat Auth pour la gestion des rôles

3.5.2 Contrat Attestation pour la génération et validation

3.5.3 Événements et logique de validation

Conclusion générale et perspectives

[Texte de conclusion générale ici...]

Webographie

[1] https://www.ibm.com/think/topics/blockchain-for-business

(consulté le 3 février 2025)

[2] https://www.flur.ee/fluree-blog/why-does-blockchain-immutability-matte

(consulté le 5 février 2025)

[3] https://help.verisart.com/en/articles/5050610-how-do-blockchain-certif

(consulté le 7 mars 2025)

[4] https://www.alibabacloud.com/blog/how-to-use-blockchain-to-ensure-the-599875

(consulté le 10 mars 2025)

[5] https://en.wikipedia.org/wiki/Blockchain

(consulté le 15 mars 2025)

[6] https://ethereum.org/en/

(consulté le 20 mars 2025)

[7] https://chainlist.org/

(consulté le 22 mars 2025)

[8] https://solana.com/developers

(consulté le 25 mars 2025)

[9] https://soliditylang.org/

(consulté le 28 mars 2025)

[10] https://aptos.dev/guides/move/move-on-aptos

(consulté le 30 mars 2025)

[11] https://faucet.quicknode.com/ethereum/sepolia

(consulté le 5 avril 2025)

[12] https://ethereum.stackexchange.com/questions/162173/why-are-sepolia-ga

(consulté le 10 avril 2025)

[13] https://blog.infura.io/ropsten-rinkeby-kovan-deprecation/

(consulté le 15 avril 2025)

[14] https://hardhat.org/hardhat-network/

(consulté le 20 avril 2025)

[15] https://trufflesuite.com/

(consulté le 25 avril 2025)

[16] https://remix.ethereum.org/

(consulté le 1er mai 2025)

[17] https://book.getfoundry.sh/

(consulté le 5 mai 2025)

[18] https://reactjs.org/

(consulté le 10 mai 2025)

[19] https://vuejs.org/

(consulté le 15 mai 2025)

[20] https://svelte.dev/

(consulté le 20 mai 2025)

[21] https://nodejs.org/en/learn/getting-started/introduction-to-nodejs

(consulté le 23 mai 2025)

[22] https://hardhat.org/hardhat3-alpha

(consulté le 23 mai 2025)

[23] https://react.dev/

(consulté le 23 mai 2025)

[24] https://dev.mysql.com/doc/

(consulté le 23 mai 2025)

[25] https://docs.ipfs.tech/install/ipfs-desktop/

(Guide d'installation IPFS Desktop, consulté le 28 mai 2025)

[26] https://ngrok.com/docs

(Documentation technique ngrok, consultée le 30 mai 2025)

[27] https://www.realite-virtuelle.com/metamask-qu-est-ce-que-c-est/

(C quoi Metamask consulté le 10 avril 2025)

[28] https://www.postman.com/product/what-is-postman/

(Postman consulté le 6 mars 2025)

[29] https://docs.github.com/fr/get-started/start-your-journey/about-github

(Git et Github consulté le 3 mars 2025)

[30] https://bility.fr/definition-visual-studio-code/

(Vscode consulté le 3 février 2025)

Annexes

- Annexe A : Extraits de code
- Annexe B : Captures d'écran
- Annexe C : Résultats de tests

Résumé

Résumé en français

[À rédiger]

Abstract in English

[À rédiger]

Résumé en arabe

[À rédiger]