

ECOLE MAROCAINE D'INGENIEURIE DE RABAT



Filière : *Génie Informatique*

Option : SI & Cyber Sécurité

Mémoire de fin d'études

TITRE :

**Développement d'un système de stockage distribué de
fichiers utilisant la BLOCKCHAIN**

Présenté par :

Mr. BAKLOUL MOHAMED

Soutenu le 11/09/2024, devant le jury composé de :

- | | |
|----------------------|----------------------|
| - Pr. SAIDI ABDELALI | (Président) |
| - Pr. HADDAR IMANE | (Rapporteuse) |
| - Pr. ANISSE KHALD | (Encadrant de l'EMG) |
| - Pr. SAFAE CHERDAL | (Examinatrice) |

*Année universitaire : **2023 / 2024***

Numéro de PFE : EMG_GI_2024/10

Table des Matières

Table des Matières	2
Liste des Figures	4
Dédicace	5
Remerciement	6
Résumé	7
Abstract	8
Liste des abréviations	9
Introduction Générale	10
Chapitre1 : Description théorique	14
1 Introduction	14
2 Objectifs	14
2.1 Les objectifs personnels :	16
3 Fonctionnement de la D'App UpShare :	17
4 Conduite et pilotage de projets	21
4.1 Préambule :	21
4.2 Méthode en cascade :	22
4.2.1 Les principes de cette méthode :	23
5 Recueil des besoins et les objectifs fixés :	24
5.1 Besoins fonctionnels et non fonctionnels :	24
5.1.1 Besoins Fonctionnels :	24
5.1.2 Besoins Non Fonctionnels	25
6 Analyse et conception :	26
6.1 Outil utilisé pour la conception :	28
6.2 Les Diagrammes UML :	29
Chapitre2 : Présentation de la réalisation	39
1 Présentation du projet UpShare :	39
1.1 Motivation et problématique du projet :	39
1.2 Solution	42
2 Mise en œuvre du projet :	43
2.1 Architecture logicielle et technique :	43
2.2 Outils utilisés :	45

3	Réalisation :	54
3.1	Introduction :	54
3.2	Présentation des interfaces de l'application :	55
3.3	Résumé du processus d'Ajout d'un Fichier :	68
Conclusion Générale		69
Référence		71

Liste des Figures

Figure 1 : Fonctionnement de la D'App UpShare-----	20
Figure 2 : Méthode en Cascade -----	22
Figure 3 : Diagramme de Cas d'utilisation-----	31
Figure 4 : Diagramme de séquence Téléchargement de fichiers -----	33
Figure 5 : Diagramme de séquence Affichage des fichiers-----	34
Figure 6 : Diagramme de séquence Suppression des fichiers-----	35
Figure 7 : Diagramme de séquence Partage des Fichiers -----	36
Figure 8 : Diagramme de séquence Accès Fichiers partagé-----	37
Figure 9 : Diagramme de Classe -----	38
Figure 10 : Architecture centralisée -----	40
Figure 11 : Architecture Décentralisée -----	44
Figure 12 : Code d'interaction avec Ganache et Pinata (API) -----	56
Figure 13 : Compte MetaMask et transaction appliqué-----	57
Figure 14 : Interface Ganache et Adresse Public Utilisé -----	58
Figure 15 : Pinata Interface et fichier Télécharger-----	59
Figure 16 : Home Page UpShare -----	61
Figure 17 : Interface Home Page (Avantage du Stockage)-----	62
Figure 18 : Bas de page du Site web UpShare -----	63
Figure 19 : Interface de Téléchargement -----	64
Figure 20 : Interface Parcourir un fichier-----	65
Figure 21 : Interface Partage des Fichiers -----	65
Figure 22 : Interface de service du UpShare-----	66
Figure 23 : Interface de consultation des fichiers partager par accès-----	66
Figure 24 : Interface Liste d'accès -----	67

Dédicace

Du profond de mon cœur, je dédie ce travail accompagné d'un amour profond et sincère, À mes chers parents, pour leur soutien et tous les sacrifices consentis et leurs prières tout au long de mes études.

À celle qui m'a entouré d'amour et d'affection, m'a arrosé d'espoirs, m'a comblé avec sa tendresse, à la source d'amour incessible qui ma bénie par ces prières « Meryem ANTARI » ma mère. Tant de phrases et d'expressions, peu importe à quel point elles sont expressives, elles ne peuvent pas exprimer le degré d'affection et d'amour que j'éprouve pour toi.

Q'ALLAH te bénisse et t'accorde une longue vie afin que je puisse te combler à mon tour

À mon support dans la vie, à mon très cher père « Abderrahim BAKLOUL », qui m'a aidé à devenir ce que je suis aujourd'hui, ceci est ma profonde gratitude pour ton éternel amour.

Ta patience sans fin et ton encouragement sont pour moi le soutien irremplaçable que tu as toujours su m'apporter. Tu as su m'inculquer le sens de la responsabilité, de l'optimisme et de la confiance en soi face aux obstacles de la vie. Je te dois ce que je suis aujourd'hui et ce que je serai demain et je ferai toujours de mon mieux pour rester ta fierté et ne jamais te décevoir. Aucune dédicace ne saurait exprimer l'amour et le respect que j'ai envers toi.

J'implore le tout puissant pour qu'il t'accorde une bonne santé, te préserve, et te protège de tout mal.

En ce jour mémorable, pour moi ainsi que pour vous, recevez ce travail en signe de ma vive reconnaissance et ma profonde estime.

Remerciement

Je désire aussi adresser mes vifs remerciements au corps professoral et administratif de l'école marocaine d'ingénierie qui m'ont fourni toutes les connaissances nécessaires durant ces années.

Mes remerciements s'adressent également à :

- Pr. Anisse Khald, l'encadrent pédagogique qui m'a aimablement dirigé et encadré au long de ce travail et pour tout le temps qu'il m'a consacré.

Les monsieur, Hicham Labani, Rafael, Mohamed labbit, Anjar Abderrahim, pour toute l'aide et la disponibilité qu'ils m'ont apportés.

Tous mes camarades de promotion pour leur support moral et intellectuel m'ont apporté, pour l'entente et la complicité qu'ils ont créé au sein de l'établissement durant ces années passées ensemble.

En résumé, mes remerciements s'orientent vers tous ceux qui de près ou de loin ont contribué à la réalisation de ce travail.

Résumé

Le présent rapport est le fruit de quatre mois de travail dans le cadre de mon projet de fin d'études.

Mon objectif était de concevoir et de développer une application blockchain permettant de stocker des fichiers de manière décentralisée, en utilisant **PINATA IPFS** pour le stockage sur **IPFS**, **Solidity** pour les contrats intelligents, et **ReactJS** pour l'interface utilisateur.

L'application permet aux utilisateurs de télécharger, stocker et partager des fichiers sur la blockchain de manière sécurisée et décentralisée. Grâce à Pinata, les fichiers sont intégrés à IPFS, assurant ainsi leur accessibilité et immuabilité.

Solidity est utilisé pour créer des contrats intelligents permettant de gérer les interactions entre les utilisateurs et les fichiers. Enfin, l'interface de l'application, développée avec ReactJS, offre une expérience utilisateur fluide.

Le projet a suivi la méthode en cascade, permettant de structurer chaque étape de manière séquentielle, avec une validation à chaque phase avant de passer à la suivante. Des diagrammes UML, incluant des diagrammes de séquence et de classe, ont été utilisés pour modéliser les interactions entre les différentes entités de l'application, garantissant ainsi une conception solide et bien structurée.

En termes de technologies, le projet a utilisé un ensemble cohérent d'outils modernes pour répondre aux exigences de la blockchain, garantissant sécurité, transparence et efficacité dans la gestion des fichiers.

Mots clés : Blockchain, IPFS, Sol, ReactJs

Abstract

This report is the result of four months of work as part of my final year project.

My objective was to design and develop a blockchain application that allows decentralized file storage, using PINATA IPFS for IPFS-based storage, Solidity for smart contracts, and ReactJS for the user interface.

The application enables users to upload, store, and share files on the blockchain in a secure and decentralized manner. With Pinata, the files are integrated into IPFS, ensuring their accessibility and immutability.

Solidity is used to create smart contracts that manage interactions between users and files. Finally, the application's interface, developed with ReactJS, provides a smooth user experience.

The project followed the waterfall method, allowing each phase to be structured sequentially, with validation at each stage before proceeding to the next. UML diagrams, including sequence and class diagrams, were used to model the interactions between the various entities of the application, ensuring a solid and well-structured design.

In terms of technologies, the project utilized a consistent set of modern tools to meet the blockchain requirements, ensuring security, transparency, and efficiency in file management.

Keywords : Blockchain, IPFS, Sol, ReactJs

Liste des abréviations

IPFS	Inter Planetary File System
JS	JavaScript
UML	Unified Modeling Language
JSX	JavaScript XML
CSS	Cascading Style Sheets
HTML	HyperText Markup Language
DAPP	Decentralized Application

Introduction Générale

De nos jours, la majorité des applications que nous utilisons quotidiennement sont basées sur un modèle centralisé. Cela signifie que les données des utilisateurs ainsi que les opérations effectuées dans ces applications sont gérées par une entité unique, généralement une entreprise ou un serveur central. Prenons l'exemple des réseaux sociaux. Lorsque nous téléchargeons des photos, envoyons des messages ou interagissons avec d'autres utilisateurs en laissant des "likes", toutes ces informations sont stockées sur les serveurs de l'entreprise qui possède l'application. Ce modèle centralisé est souvent pratique pour les entreprises, car il leur permet de contrôler et de gérer toutes les informations des utilisateurs depuis un seul endroit.

Cependant, cette centralisation présente certaines limites. En effet, les utilisateurs n'ont souvent que très peu de contrôle sur leurs propres données. Les informations sont stockées et exploitées par l'entreprise qui possède l'application, et les utilisateurs n'ont généralement pas leur mot à dire sur la manière dont ces données sont utilisées. Par ailleurs, ces systèmes centralisés sont vulnérables aux cyberattaques. Si le serveur central est piraté, toutes les données des utilisateurs peuvent être compromises. En outre, si ce serveur rencontre un problème technique ou tombe en panne, cela peut rendre l'application complètement inutilisable pour tous les utilisateurs. Face à ces défis, de nombreuses entreprises et développeurs se tournent aujourd'hui vers des solutions décentralisées, qui promettent plus de sécurité, de transparence et de contrôle pour les utilisateurs.

Les systèmes distribués, par opposition aux systèmes centralisés, reposent sur une architecture où les données ne sont pas stockées en un seul point, mais distribuées sur plusieurs nœuds (ordinateurs ou serveurs) répartis à travers le monde. Cela signifie qu'aucune entité unique ne contrôle l'intégralité du système, rendant les systèmes distribués beaucoup plus résistants aux pannes et aux attaques. Par exemple, dans une application distribuée, les informations des utilisateurs ne sont pas stockées sur un seul serveur, mais sur plusieurs, de sorte que si un serveur tombe en panne ou est attaqué, les autres serveurs du réseau peuvent continuer à fonctionner normalement.

L'un des principaux avantages des systèmes distribués est la sécurité. En répartissant les données sur plusieurs nœuds, il devient beaucoup plus difficile pour un pirate informatique d'accéder à toutes les informations. Pour compromettre un système distribué, il faudrait attaquer simultanément plusieurs nœuds, ce qui est bien plus complexe que de pirater un seul serveur centralisé. De plus, dans un système distribué, les données des utilisateurs ne sont pas stockées en un seul endroit, mais sont fragmentées et répliquées à travers le réseau, rendant l'accès non autorisé beaucoup plus difficile.

Un autre avantage majeur des systèmes distribués est le contrôle accru des utilisateurs sur leurs propres données. Dans une application centralisée, les utilisateurs doivent faire confiance à l'entreprise qui gère l'application pour protéger leurs informations et respecter leur vie privée. Cependant, dans une application distribuée, les utilisateurs conservent un contrôle total sur leurs données. Ils peuvent choisir quelles informations partager, avec qui et dans quelles conditions. Cette transparence accrue permet aux utilisateurs de mieux gérer leur vie privée et de protéger leurs informations sensibles.

L'une des technologies les plus prometteuses dans le domaine des systèmes distribués est la blockchain. La blockchain est une technologie de registre distribué qui permet de stocker et de sécuriser des informations de manière transparente et immuable. Contrairement aux bases de données centralisées, où les informations sont contrôlées par une seule entité, la blockchain permet à tous les participants du réseau de partager une version commune et sécurisée des données.

Le fonctionnement de la blockchain repose sur des blocs de données qui sont liés entre eux de manière cryptographique. Chaque bloc contient un ensemble de transactions ou d'informations, et une fois qu'un bloc est ajouté à la chaîne, il ne peut plus être modifié. Cela garantit l'intégrité des données et empêche toute altération ou manipulation frauduleuse. De plus, chaque participant du réseau possède une copie complète de la blockchain, ce qui garantit que le système reste opérationnel même en cas de défaillance d'un ou plusieurs nœuds du réseau.

La blockchain permet également l'exécution de "contrats intelligents" (ou smart contracts), qui sont des programmes informatiques autonomes qui s'exécutent automatiquement lorsque certaines conditions sont remplies. Ces contrats intelligents permettent de sécuriser et d'automatiser les transactions entre utilisateurs sans avoir besoin d'un tiers de confiance. Par exemple, dans une application de stockage distribué, un contrat intelligent pourrait être utilisé pour garantir que seuls les utilisateurs ayant les droits nécessaires puissent accéder à un fichier.

Mon projet : une application de stockage décentralisée

Dans le cadre de mon projet de fin d'études, j'ai développé une application de stockage décentralisée qui permet aux utilisateurs de stocker leurs fichiers de manière sécurisée et distribuée. Cette application utilise trois technologies principales : IPFS, Solidity et ReactJS.

- IPFS est un système de stockage distribué qui permet de répartir les fichiers sur plusieurs nœuds à travers le monde. Contrairement aux systèmes de stockage centralisés, où les fichiers sont stockés sur un seul serveur, IPFS divise les fichiers en plusieurs fragments, qui sont ensuite distribués sur le réseau. Cela garantit que les fichiers restent accessibles même en cas de panne d'un ou plusieurs nœuds du réseau. De plus, les fichiers stockés sur IPFS sont immuables, ce qui signifie qu'une fois qu'un fichier est téléchargé, il ne peut plus être modifié ni supprimé. Cette immuabilité renforce la sécurité des fichiers et garantit leur intégrité à long terme.

- Solidity est un langage de programmation utilisé pour écrire des contrats intelligents sur la blockchain Ethereum. Dans mon application, les contrats intelligents sont utilisés pour gérer les interactions entre les utilisateurs et l'application, en garantissant que seuls les utilisateurs autorisés peuvent accéder à leurs fichiers. Par exemple, un contrat intelligent pourrait vérifier que l'utilisateur a payé les frais de stockage avant de lui permettre de télécharger un fichier. Cela permet de sécuriser les transactions entre les utilisateurs et l'application sans avoir besoin d'un intermédiaire.
- ReactJS est une bibliothèque JavaScript qui permet de créer des interfaces utilisateurs dynamiques et interactives. Grâce à ReactJS, j'ai pu concevoir une interface simple et intuitive, où les utilisateurs peuvent facilement télécharger, partager et gérer leurs fichiers. L'interface permet également aux utilisateurs de visualiser l'état de leurs fichiers et de suivre les transactions effectuées sur la blockchain en temps réel.

L'application que j'ai développée présente plusieurs avantages par rapport aux solutions de stockage centralisées traditionnelles. Tout d'abord, la sécurité est grandement renforcée grâce à l'utilisation d'IPFS et de la blockchain. Les fichiers sont distribués sur plusieurs nœuds, ce qui les rend beaucoup plus résistants aux attaques et aux pannes. De plus, les contrats intelligents garantissent que seuls les utilisateurs autorisés peuvent accéder à leurs fichiers, renforçant ainsi la confidentialité et la protection des données.

Chapitre1 : Description théorique

1 Introduction

Dans ce premier chapitre, je présente les fondements théoriques de mon projet. Cette section détaille les objectifs principaux du site web que j'ai développé, en mettant en lumière les raisons de son existence et les besoins auxquels il répond. J'y décris également la conception des différents diagrammes, qui jouent un rôle clé dans la structuration et la planification du développement de l'application. Ces diagrammes permettent de visualiser les interactions entre les différentes composantes du système. L'objectif de cette partie est de poser des bases solides pour mieux comprendre les choix stratégiques et méthodologiques adoptés lors du processus de développement.

2 Objectifs

Le stockage de fichiers distribué est une nouvelle façon de gérer les données, bien plus efficace que les méthodes traditionnelles centralisées. Au lieu de garder toutes les informations sur un seul serveur, comme c'est le cas dans les systèmes centralisés, les fichiers sont répartis sur plusieurs ordinateurs à travers le monde. Cela rend le stockage plus sûr et plus résistant face aux problèmes comme les cyberattaques ou les pannes. La technologie de la blockchain joue un rôle clé dans ce type de stockage, en assurant une gestion sécurisée et transparente des fichiers.

Une des forces du stockage distribué est sa capacité à garantir que les données restent accessibles même en cas de problème sur une partie du réseau. Grâce à la redondance, les fichiers sont copiés sur plusieurs nœuds du réseau. Cela signifie que même si une partie des ordinateurs n'est plus disponible, les données restent toujours accessibles à partir d'autres points du réseau. Cette méthode n'est pas seulement plus sûre, elle est aussi plus économique, car elle optimise l'utilisation des ressources et permet de réduire les coûts à long terme.

Avec des outils comme IPFS, cette approche décentralisée devient encore plus intéressante. IPFS permet de distribuer les fichiers à travers plusieurs ordinateurs, ce qui rend le système plus résistant et plus fiable. De plus, la confidentialité des informations est renforcée, car il devient plus difficile pour des pirates d'accéder aux données stockées sur plusieurs machines plutôt que sur un seul serveur centralisé.

Dans ce cadre, je propose une solution moderne et sécurisée pour le stockage et la gestion des fichiers, documents et images. En utilisant la blockchain et IPFS, cette solution permet aux entreprises et aux particuliers de mieux contrôler leurs données. Elle est particulièrement adaptée aux besoins actuels, où la sécurité et la maîtrise des informations sont des priorités. De plus, cette méthode réduit les coûts en utilisant des ressources de manière plus intelligente et en évitant les frais élevés associés aux serveurs centralisés.

Ce système facilite aussi les échanges avec les clients, car il simplifie la gestion des fichiers au quotidien. Par exemple, il est possible de partager des documents ou des images de manière rapide et sécurisée, sans passer par des serveurs centralisés qui peuvent être lents ou vulnérables. Dans un environnement numérique qui évolue rapidement, cette solution offre une alternative moderne et fiable aux méthodes traditionnelles.

2.1 Les objectifs personnels :

Dans le cadre de ce projet, plusieurs objectifs personnels ont été fixés pour profiter au maximum de cette expérience :

- Utilisation des connaissances techniques : Ce projet m'a donné l'occasion de mettre en pratique les compétences que j'ai acquises en génie logiciel. J'ai utilisé des technologies comme IPFS, Solidity et ReactJS pour résoudre des problèmes concrets, tout en explorant de nouvelles solutions décentralisées basées sur la blockchain.
- Rédaction des spécifications fonctionnelles : Une des étapes essentielles de ce projet a été de bien définir les besoins techniques et fonctionnels. Cela m'a permis de mieux maîtriser la rédaction de documents techniques et de m'assurer que les spécifications étaient en accord avec les objectifs du projet.
- Développement d'un projet réel : Ce projet m'a permis de passer de la théorie à la pratique en développant une application décentralisée. J'ai ainsi pu faire face aux contraintes techniques du monde réel, comprendre les défis liés à la gestion d'un projet et trouver des solutions adaptées.
- Renforcement de la responsabilité : Travailler sur un projet complexe m'a appris à mieux gérer mon temps et les ressources à disposition. J'ai aussi développé un plus grand sens de l'autonomie et de la prise d'initiative, des qualités importantes dans un environnement professionnel.
- Apprentissage auprès des autres : Ce projet m'a offert l'opportunité d'apprendre non seulement à travers mes propres recherches, mais aussi grâce aux conseils de mes formateurs et aux échanges avec mes collègues, enrichissant ainsi mes connaissances et mon expérience.

3 Fonctionnement de la D'App UpShare :

L'application "UpShare" repose sur les principes de la décentralisation et utilise des technologies comme IPFS et la blockchain Ethereum. Son objectif principal est d'offrir aux utilisateurs une solution sécurisée pour stocker et gérer leurs documents, sans dépendre d'un serveur central.

"UpShare" utilise IPFS, un système de fichiers distribué. Contrairement aux méthodes traditionnelles où les données sont stockées sur un seul serveur, IPFS permet de les répartir sur plusieurs ordinateurs dans le monde. Cela garantit plusieurs avantages majeurs :

- Sécurité renforcée : Les fichiers ne sont pas accessibles par un seul point d'entrée, rendant les attaques informatiques plus difficiles.
- Immuabilité : Une fois un fichier enregistré sur IPFS, il ne peut ni être modifié, ni supprimé, ce qui assure que les documents restent intacts.
- Accessibilité mondiale : Les fichiers sont disponibles même si certains ordinateurs du réseau sont hors ligne.

Lorsqu'un utilisateur télécharge un document sur "UpShare", celui-ci est stocké sur IPFS et une adresse unique est créée. Cette adresse, appelée adresse IPFS, permet de retrouver le fichier de manière fiable et sécurisée.

En plus d'IPFS, l'application utilise la blockchain Ethereum pour gérer les échanges entre les utilisateurs et les documents, grâce à un contrat intelligent. Ce contrat est un programme qui exécute des actions automatiquement dès que certaines conditions sont remplies. Dans le cas de "UpShare", il assure que les utilisateurs peuvent accéder à leurs documents de manière sécurisée et que toutes les transactions sont transparentes.

Le processus commence par l'authentification de l'utilisateur. Pour se connecter, celui-ci utilise MetaMask, une extension de navigateur permettant d'interagir avec la blockchain Ethereum. MetaMask assure une connexion sécurisée, en exigeant que l'utilisateur se connecte à son portefeuille Ethereum. Chaque utilisateur dispose ainsi d'une adresse unique sur la blockchain, ce qui lui permet d'être identifié de façon certaine.

Une fois connecté, l'utilisateur peut interagir avec le contrat intelligent de plusieurs manières :

- Création d'un bloc utilisateur : Chaque utilisateur peut créer un bloc qui lui est propre sur la blockchain. Ce bloc contient son adresse Ethereum ainsi que les informations relatives à ses documents.
- Téléchargement de fichiers : Lorsque l'utilisateur télécharge un fichier, l'adresse IPFS de ce fichier est ajoutée à son bloc sur la blockchain Ethereum. Cela permet à l'utilisateur de retrouver tous ses documents à partir de son bloc sur la blockchain.
- Récupération de fichiers : Pour accéder à un document, l'utilisateur interagit avec le contrat intelligent pour récupérer l'adresse IPFS de ses fichiers. Il peut ensuite utiliser cette adresse pour consulter ses documents directement sur IPFS.

Ce système garantit que les documents sont non seulement sécurisés, mais également accessibles de manière simple et transparente.

La gestion des fichiers sur "UpShare" est conçue pour être à la fois simple et sécurisée pour tous les utilisateurs, qu'ils soient administrateurs ou clients. Voici comment le processus fonctionne :

Connexion via MetaMask : L'utilisateur commence par se connecter à MetaMask, ce qui permet de s'authentifier en toute sécurité. Chaque utilisateur est identifié par son adresse Ethereum.

Création de son bloc utilisateur : Grâce au contrat intelligent, l'utilisateur peut créer un bloc spécifique sur la blockchain. Ce bloc contient son adresse Ethereum ainsi que toutes les informations nécessaires à la gestion de ses fichiers.

Téléchargement des fichiers sur IPFS : Une fois connecté, l'utilisateur peut télécharger des documents sur IPFS. Chaque fichier téléchargé reçoit une adresse IPFS unique qui est ensuite liée au bloc utilisateur sur la blockchain Ethereum. Les fichiers sont donc stockés de manière décentralisée et associés directement à l'utilisateur grâce à la blockchain.

Accès aux fichiers : Lorsqu'un utilisateur souhaite consulter un document, il récupère l'adresse IPFS associée à son fichier depuis son bloc utilisateur sur la blockchain.

Avec cette adresse, il peut accéder au document sur IPFS, sans avoir besoin de passer par un serveur central.

Ce système garantit que les fichiers sont non seulement bien protégés, mais aussi facilement accessibles à tout moment.

L'application "UpShare" offre plusieurs bénéfices majeurs aux utilisateurs, notamment en termes de sécurité et de facilité d'utilisation :

- **Sécurité renforcée** : Les documents sont stockés de manière décentralisée, ce qui les protège contre les attaques qui viseraient un serveur unique. De plus, l'utilisation de la blockchain garantit que seules les personnes autorisées peuvent accéder aux documents.
- **Transparence** : Toutes les interactions avec les documents sont enregistrées sur la blockchain. Cela signifie que chaque action peut être suivie et vérifiée, assurant ainsi une traçabilité complète.
- **Contrôle total pour les utilisateurs** : Chaque utilisateur possède un contrôle total sur ses fichiers via son propre bloc sur la blockchain. Il n'y a pas d'intermédiaire ou d'entité centrale pour gérer les données.
- **Fiabilité** : Grâce à la décentralisation d'IPFS, les documents sont disponibles en permanence, même si certaines parties du réseau sont hors ligne. Ce système garantit que les documents restent accessibles et protégés.

Voici une illustration décrivant le fonctionnement de l'application :

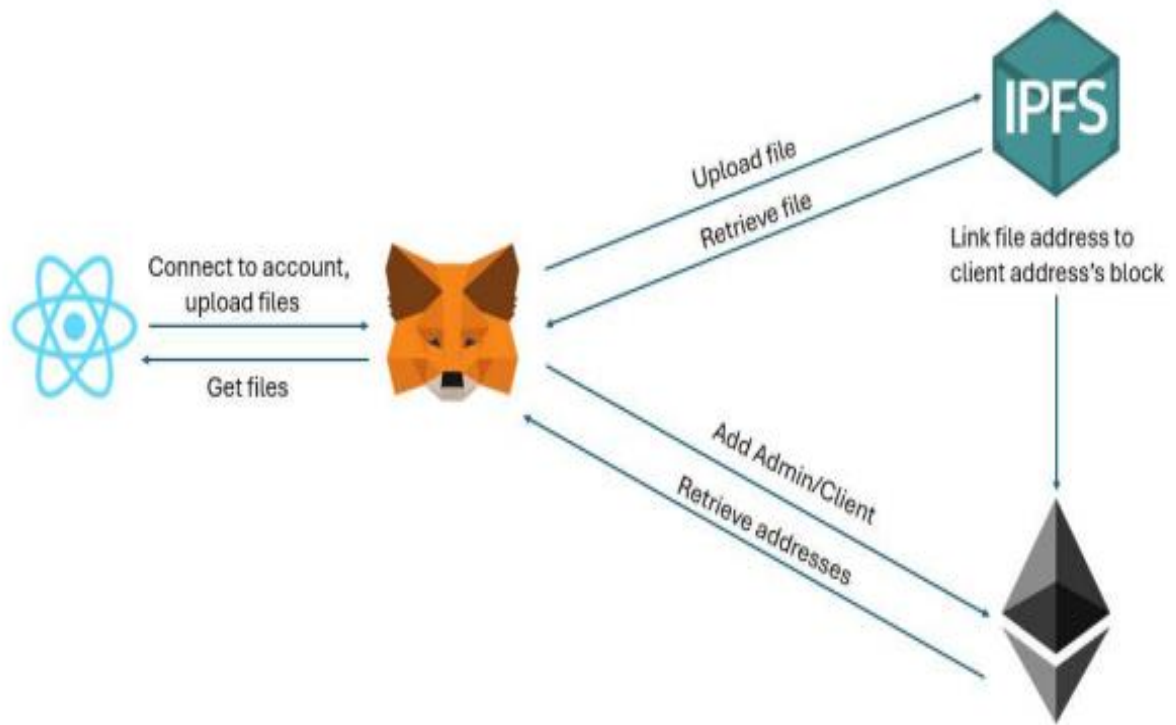


Figure 1 : Fonctionnement de la D'App UpShare

4 Conduite et pilotage de projets

4.1 Préambule :

Le cycle de vie d'un projet comprend plusieurs étapes importantes qui aident à bien organiser et suivre le travail. La première étape est l'initiation, où l'on définit l'objectif du projet. On identifie les besoins ou les problèmes à résoudre. C'est aussi à ce moment que tout le monde se met d'accord sur ce que le projet doit accomplir et les grandes lignes à suivre.

La planification est ensuite une étape clé. Elle permet de décider des tâches à réaliser, des délais à respecter, des ressources nécessaires et des responsabilités de chacun. Cette planification peut changer au fur et à mesure que le projet avance, pour s'adapter aux nouveaux éléments qui apparaissent.

Pendant la phase d'exécution, l'équipe se met au travail pour réaliser les tâches définies. La réussite de cette étape dépend de la qualité de la préparation et de la coordination entre les membres de l'équipe.

Le contrôle est une étape où l'on vérifie si le projet avance comme prévu. On mesure les progrès et on fait des ajustements si nécessaire pour rester sur la bonne voie. Cela permet d'atteindre les objectifs fixés tout en respectant le temps et les ressources.

4.2 Méthode en cascade :

Le cycle de vie en cascade est particulièrement adapté aux projets d'une durée inférieure à une année ou comportant une composante réglementaire solide. Ce modèle est basé sur un principe strict : le projet est divisé en plusieurs phases distinctes, avec un passage linéaire d'une phase à l'autre, sans retour en arrière. Cela signifie que chaque phase doit être complètement terminée avant de passer à la suivante. Une fois qu'une phase est achevée, son résultat sert de point de départ pour la phase suivante. Cette méthode offre une structure claire et permet une bonne maîtrise du processus, notamment dans les projets où les exigences sont bien définies dès le début.

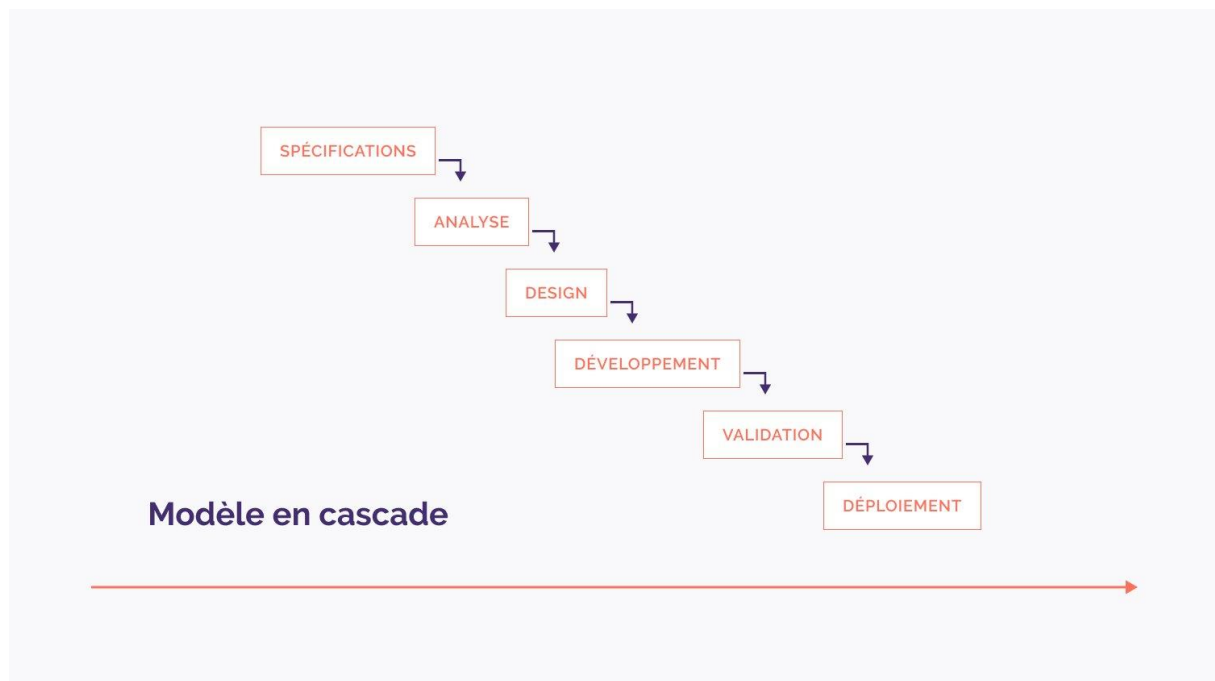


Figure 2 : Méthode en Cascade

4.2.1 Les principes de cette méthode :

La méthode en cascade est une approche classique de gestion de projet qui se déroule en plusieurs étapes distinctes. Chaque étape doit être complétée avant de passer à la suivante, assurant ainsi une progression structurée et organisée du projet. Voici les principales phases de cette méthode :

Spécifications : Cette première étape consiste à définir précisément les besoins et les exigences du projet. Il s'agit de bien comprendre ce que le client ou les parties prenantes attendent du produit final. Cette phase est cruciale, car elle sert de base pour toutes les étapes suivantes.

Analyse : Une fois les spécifications définies, l'étape d'analyse permet d'étudier ces besoins en profondeur. L'objectif est de mieux comprendre les attentes et de déterminer les solutions les plus adaptées pour les réaliser. Cette phase prépare le terrain pour la conception.

Conception : Durant cette étape, un plan détaillé du système à créer est élaboré. Cela inclut l'architecture, les composants, et les fonctionnalités du projet. La conception fournit une vision claire de la manière dont le projet sera développé.

Développement : Les développeurs commencent à construire le projet en suivant le plan établi lors de la conception. C'est la phase où le code est écrit et où les différentes fonctionnalités sont mises en place.

Validation : Une fois le développement terminé, le projet est testé pour s'assurer qu'il fonctionne correctement et qu'il répond aux spécifications définies initialement. Cette étape permet de détecter et de corriger les éventuels problèmes avant la livraison du produit.

Déploiement : La dernière étape consiste à mettre le projet à disposition des utilisateurs. Le produit final est livré, et les utilisateurs peuvent commencer à l'utiliser dans le cadre prévu.

5 Recueil des besoins et les objectifs fixés :

5.1 Besoins fonctionnels et non fonctionnels :

5.1.1 Besoins Fonctionnels :

Sécurité : La sécurité est très importante pour notre application décentralisée. Nous devons protéger les données des utilisateurs contre les accès non autorisés et les fuites d'informations. Pour cela, il est essentiel de mettre en place des mesures de sécurité efficaces.

Résilience : L'application doit être disponible à tout moment, même si certains nœuds d'IPFS rencontrent des problèmes. Il est donc crucial que les fichiers restent accessibles grâce à des méthodes de "pinning". Cela permet de s'assurer que les données ne sont pas perdues, même si certains nœuds se déconnectent.

Performance : La performance de l'application est également essentielle. Elle doit offrir des temps de réponse rapides pour les opérations d'upload, de téléchargement et de gestion des fichiers. Les utilisateurs ne doivent pas attendre longtemps pour effectuer ces actions, car cela pourrait nuire à leur expérience.

Scalabilité : La solution doit être conçue pour supporter un nombre croissant d'utilisateurs et de fichiers sans affecter la performance. Lorsque le nombre d'utilisateurs augmente, il est important que l'application puisse s'adapter et continuer à fonctionner efficacement.

Compatibilité : Il est aussi important que l'application soit compatible avec les principaux navigateurs web et des extensions comme MetaMask. Une large compatibilité permet de toucher un plus grand nombre d'utilisateurs et garantit que chacun, quel que soit son environnement de travail, peut accéder à l'application sans problème.

Usabilité : L'interface utilisateur doit être facile à utiliser, même pour ceux qui ne sont pas très familiarisés avec les technologies décentralisées.

Maintenabilité : Le code de l'application doit être bien structuré et documenté pour faciliter les mises à jour et les modifications futures.

Conformité Légale : L'application doit respecter les lois sur la protection des données et la confidentialité, notamment concernant le stockage et la gestion des informations personnelles. Cela inclut le respect des lois sur la protection des données.

5.1.2 Besoins Non Fonctionnels

Disponibilité : Les fichiers doivent toujours être accessibles, même si certains serveurs tombent en panne.

Performance : Les actions comme télécharger ou envoyer des fichiers doivent être rapides.

Scalabilité : L'application doit pouvoir gérer plus d'utilisateurs et de fichiers sans ralentir.

Compatibilité : Elle doit fonctionner avec plusieurs navigateurs et extensions comme MetaMask pour être accessible au plus grand nombre.

Facilité d'utilisation : L'interface doit être simple à comprendre et utiliser, même pour les débutants.

Maintenabilité : Le code doit être facile à modifier et à mettre à jour pour assurer l'évolution de l'application.

Conformité légale : L'application doit respecter les lois sur la protection des données pour garantir la confidentialité des utilisateurs.

6 Analyse et conception :

Dans le cadre de la modélisation du projet, le langage UML a été adopté en raison de sa simplicité et de sa large utilisation dans le domaine de l'informatique. UML, ou « Unified Modeling Language », est l'un des standards les plus couramment utilisés pour la conception des logiciels. Il fournit une approche visuelle efficace pour la modélisation et la documentation des systèmes informatiques. En effet, UML se base sur des représentations schématiques des composants logiciels, ce qui permet de visualiser à la fois la structure et le comportement d'un système de manière claire et organisée.

L'un des avantages majeurs d'UML est qu'il permet de modéliser différents aspects d'une application, notamment les relations entre les objets, les interactions entre les composants et les flux de données. En choisissant UML, il devient plus simple de créer une représentation visuelle des processus complexes et des interactions dans le système, facilitant ainsi la communication entre les développeurs et les autres parties prenantes du projet.

Comme mon projet a débuté dès les premières étapes de développement, j'ai commencé par réaliser des diagrammes de classes. Ces diagrammes sont essentiels pour structurer et visualiser la manière dont les différents éléments du système interagissent. Les diagrammes de classes représentent les relations entre les objets, leurs attributs et leurs méthodes, ce qui permet de mieux comprendre l'architecture du système et de clarifier les interactions entre les différentes entités logicielles.

Les diagrammes de classes sont particulièrement utiles pour illustrer les relations d'héritage, d'association et de composition entre les objets, ce qui facilite la structuration du code lors de la phase de développement. De plus, ils fournissent une base solide pour identifier les fonctionnalités à implémenter et les interactions à gérer, ce qui garantit une meilleure planification du développement logiciel.

Justification du Choix d'UML

Le choix d'UML pour la modélisation de ce projet repose sur plusieurs raisons, qui se traduisent par des avantages concrets lors de la conception et du développement du système.

- **Simplicité d'utilisation** : UML offre des diagrammes clairs et concis, ce qui permet aux programmeurs de différents niveaux de compétence de comprendre facilement les concepts modélisés. Même les développeurs qui n'ont pas participé directement à la phase de conception peuvent se familiariser rapidement avec les schémas UML, facilitant ainsi la collaboration et la maintenance du projet.
- **Outils de planification** : UML aide à planifier les différentes fonctionnalités du programme avant même le début de la programmation. Grâce aux diagrammes UML, il devient plus facile d'identifier les éléments clés du système et de structurer les différentes étapes du développement. Cela permet également de prévoir les potentielles difficultés techniques et de mieux organiser le travail des équipes de développement.

- **Flexibilité et standardisation** : UML est une notation standard, utilisée par de nombreux professionnels du logiciel à travers le monde. Cela permet non seulement de garantir une compréhension commune des modèles et des schémas, mais aussi d'assurer une interopérabilité avec d'autres outils et méthodes de développement. En utilisant UML, on s'assure que le projet peut facilement être repris par d'autres équipes ou intégré dans des systèmes existants, sans nécessiter de modifications majeures dans la structure de la documentation.
- **Structuration des fonctionnalités et des données** : Les diagrammes UML permettent de structurer efficacement les fonctionnalités et les données d'une application. Ils fournissent une vue d'ensemble de la manière dont les différentes parties du système interagissent, facilitant ainsi la conception d'une architecture logique et bien organisée. Cela garantit que les fonctionnalités sont implémentées de manière cohérente et que les relations entre les données sont bien définies dès le départ.

6.1 Outil utilisé pour la conception :

Visual Paradigm Online est un outil en ligne spécialisé dans la conception de diagrammes, particulièrement adapté à la programmation. Il prend en charge une grande variété de diagrammes commerciaux et techniques, tels que UML (Unified Modeling Language), BPMN (Business Process Model and Notation), URD (User Requirements Document), DFD (Data Flow Diagram), et SysML (Systems Modeling Language). Ce large éventail de fonctionnalités en fait un outil polyvalent pour la modélisation de systèmes complexes.

La plateforme se distingue par son interface graphique intuitive, qui simplifie la création et la manipulation des diagrammes, permettant aux utilisateurs de naviguer facilement entre les différentes options et outils proposés. Cette accessibilité rend l'outil approprié aussi bien pour les débutants que pour les professionnels chevronnés.

De plus, Visual Paradigm Online est compatible avec une grande diversité d'applications, ce qui le rend flexible et pratique pour intégrer des processus de conception dans divers environnements de travail et de développement. Son approche visuelle et sa compatibilité en font un atout essentiel pour la planification et la documentation des projets.

6.2 Les Diagrammes UML :

a. Les acteurs principaux :

Individus : Ceux qui souhaitent stocker, partager ou gérer des fichiers personnels de manière sécurisée et décentralisée.

Professionnels : Des utilisateurs comme les freelances ou les travailleurs à distance qui ont besoin d'un espace de stockage fiable et sécurisé, entreprises et Organisations.

Startups et PME : Les petites et moyennes entreprises.

Organisations de Recherche : Les institutions académiques ou de recherche.

b. Tableau des exigences :

Le système à concevoir doit permettre à l'utilisateur d'effectuer les opérations suivantes :

Fonction	Acteurs
Offrir la possibilité de s'authentifier	Professionnels, Individus, utilisateurs
Offrir la possibilité de consulter les différents modules de la page home.	Professionnels, Individus, utilisateurs

Offrir la possibilité de l'ajout/suppression de fichiers.	Professionnels, Individus, utilisateurs
Offrir la possibilité de voir la liste des fichiers sur Pinata.	Professionnels, Individus, utilisateurs
Offrir la possibilité de se déconnecter.	Professionnels, Individus, utilisateurs
Offrir la possibilité de changer le compte MetaMask	Professionnels, Individus, utilisateurs

c. Diagramme de cas d'utilisation :

Le diagramme de cas d'utilisation illustre les interactions entre les utilisateurs (acteurs) et le système, en mettant en évidence les différentes fonctionnalités offertes par l'application. Il permet de visualiser les scénarios d'utilisation, facilitant ainsi la compréhension des besoins des utilisateurs et des exigences fonctionnelles du système.

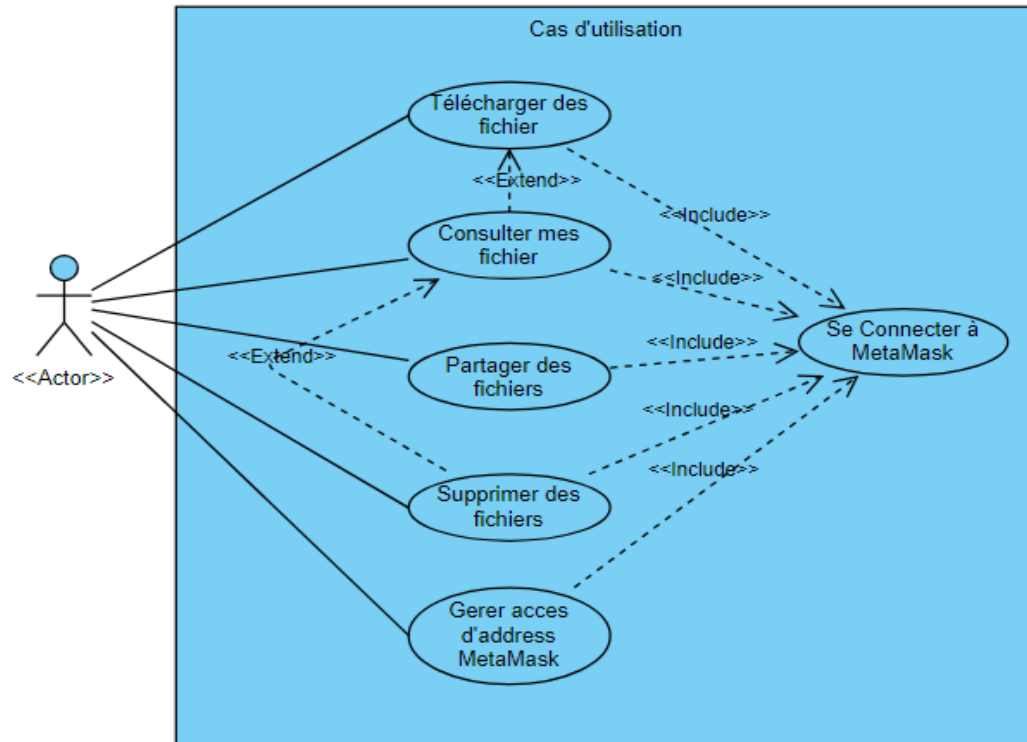


Figure 3 : Diagramme de Cas d'utilisation

d. Description textuelle :

Pour rendre notre diagramme plus lisible et afin de décrire le comportement d'un système, les concepteurs d'UML proposent l'utilisation d'une technique nommée la description textuelle.

Description textuelle :

Objectif					
Accéder à l'espace personnalisé					
Acteur principaux	Professionnels, Individus, utilisateurs				
Rôle de cas d'utilisation	Ce scénario permet à l' utilisateurs , d'accéder au site web et son espace.				
Précondition :	Existence des données de l'utilisateur dans Pinata IPFS				
Post-condition :	Affichage de de la page de téléchargement				
Scénario nominal :	<table> <tr> <th>Actions des acteurs</th><th>Action du système</th></tr> <tr> <td> 1. Visiter la page Home 2. Cliquer sur téléchargement 3. affichage de la page de téléchargement </td><td> 1. Affichage du connections MetaMask avec le site web 2. Vérification de l'existence de l'utilisateur 3. si utilisateur existe ou n'existe pas alors Afficher la page de téléchargement (tant que l'utilisateur a un compte MetaMask, il peut télécharger) </td></tr> </table>	Actions des acteurs	Action du système	1. Visiter la page Home 2. Cliquer sur téléchargement 3. affichage de la page de téléchargement	1. Affichage du connections MetaMask avec le site web 2. Vérification de l'existence de l'utilisateur 3. si utilisateur existe ou n'existe pas alors Afficher la page de téléchargement (tant que l'utilisateur a un compte MetaMask, il peut télécharger)
Actions des acteurs	Action du système				
1. Visiter la page Home 2. Cliquer sur téléchargement 3. affichage de la page de téléchargement	1. Affichage du connections MetaMask avec le site web 2. Vérification de l'existence de l'utilisateur 3. si utilisateur existe ou n'existe pas alors Afficher la page de téléchargement (tant que l'utilisateur a un compte MetaMask, il peut télécharger)				

e. Diagramme de séquence :

Un diagramme de séquence est un outil qui montre comment les objets d'un système interagissent entre eux au fil du temps. Il illustre l'ordre des messages échangés et les interactions dans un scénario particulier.

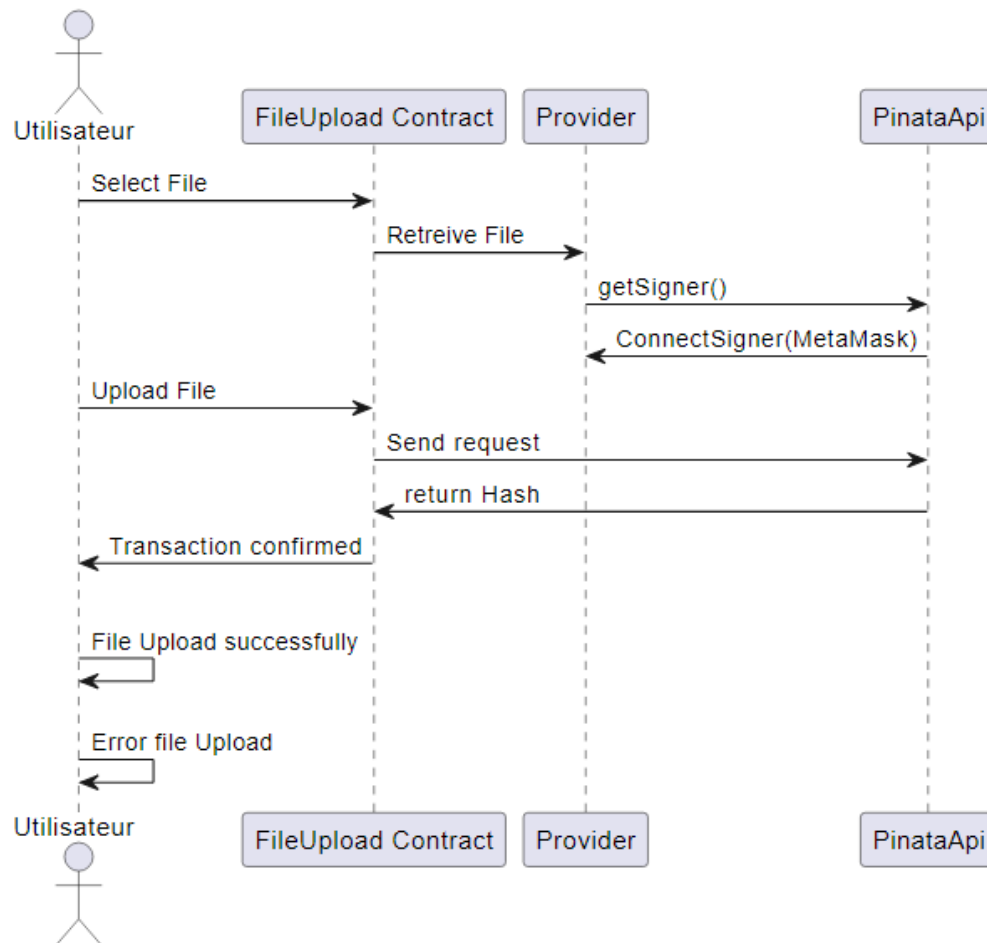


Figure 4 : Diagramme de séquence Téléchargement de fichiers

Explication par étape :

1. L'utilisateur sélectionne un fichier.
2. Le fichier est signé via MetaMask.
3. Le fichier est envoyé via le contrat FileUpload.
4. PinataApi renvoie le hash du fichier.
5. En cas de succès, la transaction est confirmée.

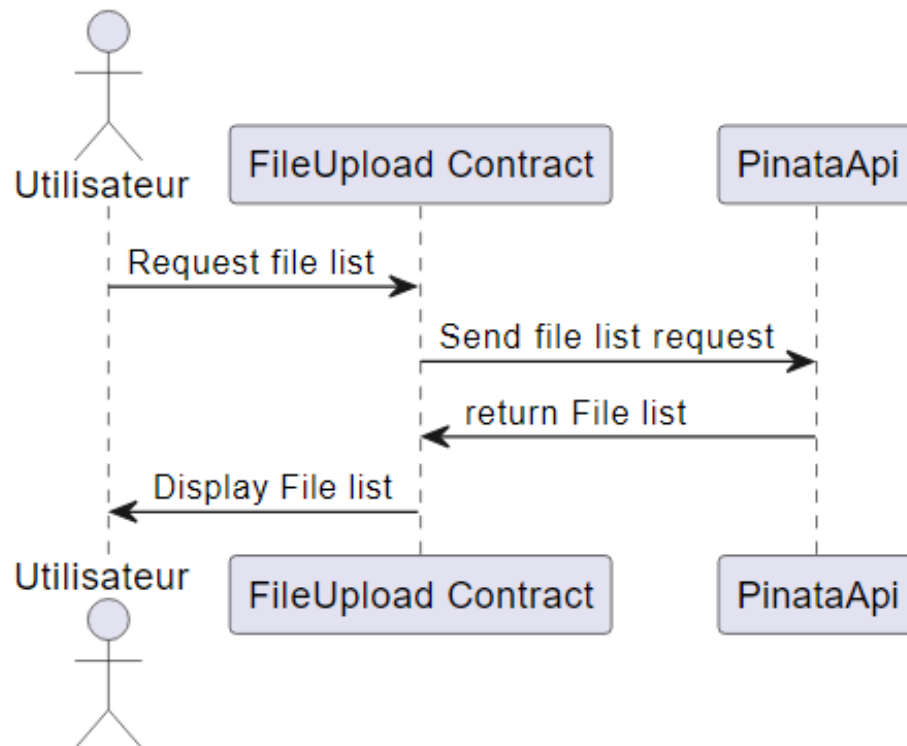


Figure 5 : Diagramme de séquence Affichage des fichiers

Explication par étape :

1. L'utilisateur demande la liste des fichiers.
2. Le contrat FileUpload fait une requête pour obtenir la liste des fichiers.
3. La liste est retournée à l'utilisateur.

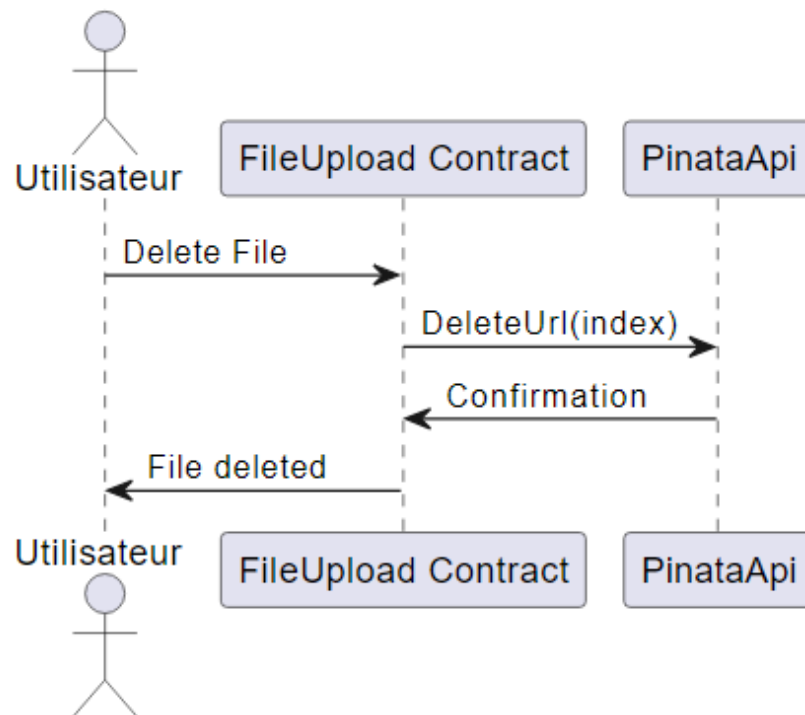


Figure 6 : Diagramme de séquence Suppression des fichiers

Explication par étape :

1. L'utilisateur demande la suppression d'un fichier.
2. Le contrat FileUpload envoie l'index à supprimer.
3. Confirmation de la suppression du fichier.

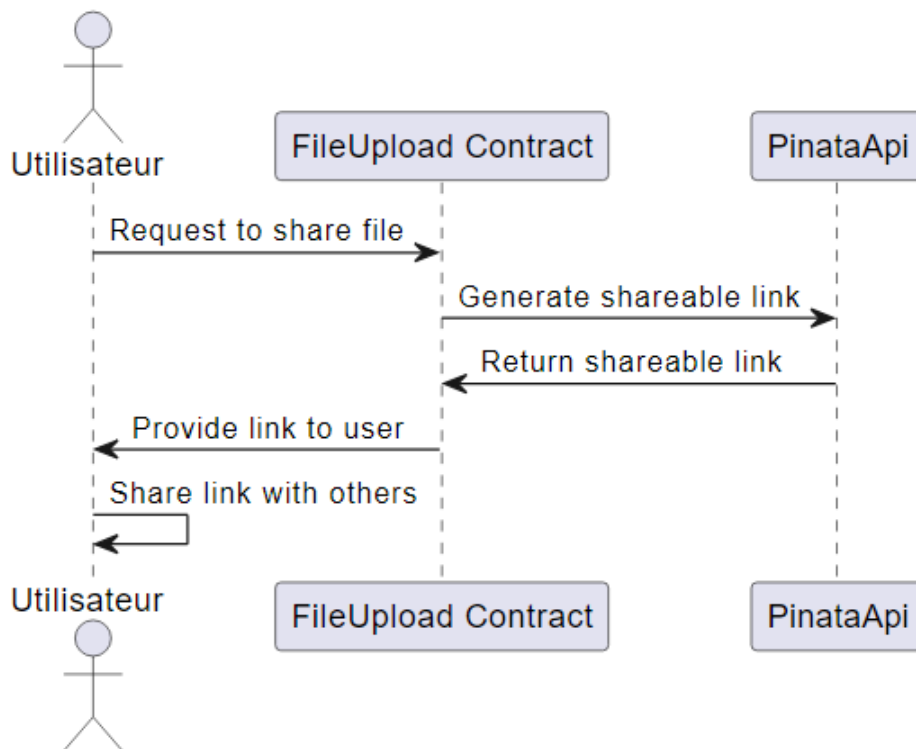


Figure 7 : Diagramme de séquence Partage des Fichiers

Explication par étape :

1. L'utilisateur souhaite partager un fichier.
2. Le contrat FileUpload génère un lien de partage en interrogeant PinataApi.
3. Le lien est généré et renvoyé à l'utilisateur.
4. L'utilisateur partage le lien avec d'autres personnes.

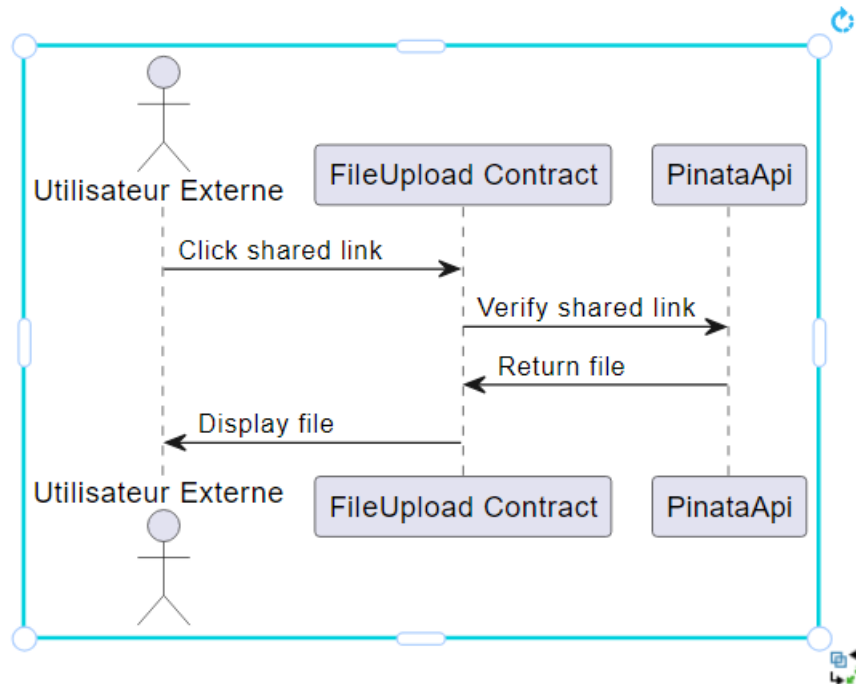


Figure 8 : Diagramme de séquence Accès Fichiers partagé

Explication par étape :

1. Un utilisateur externe clique sur le lien partagé.
2. Le lien est vérifié par le contrat FileUpload.
3. PinataApi renvoie le fichier correspondant au lien partagé.
4. Le fichier est affiché à l'utilisateur externe.

f. Diagramme de classe

Le diagramme de classes est un schéma utilisé en génie logiciel pour présenter les classes et les interfaces des systèmes ainsi que leurs relations.

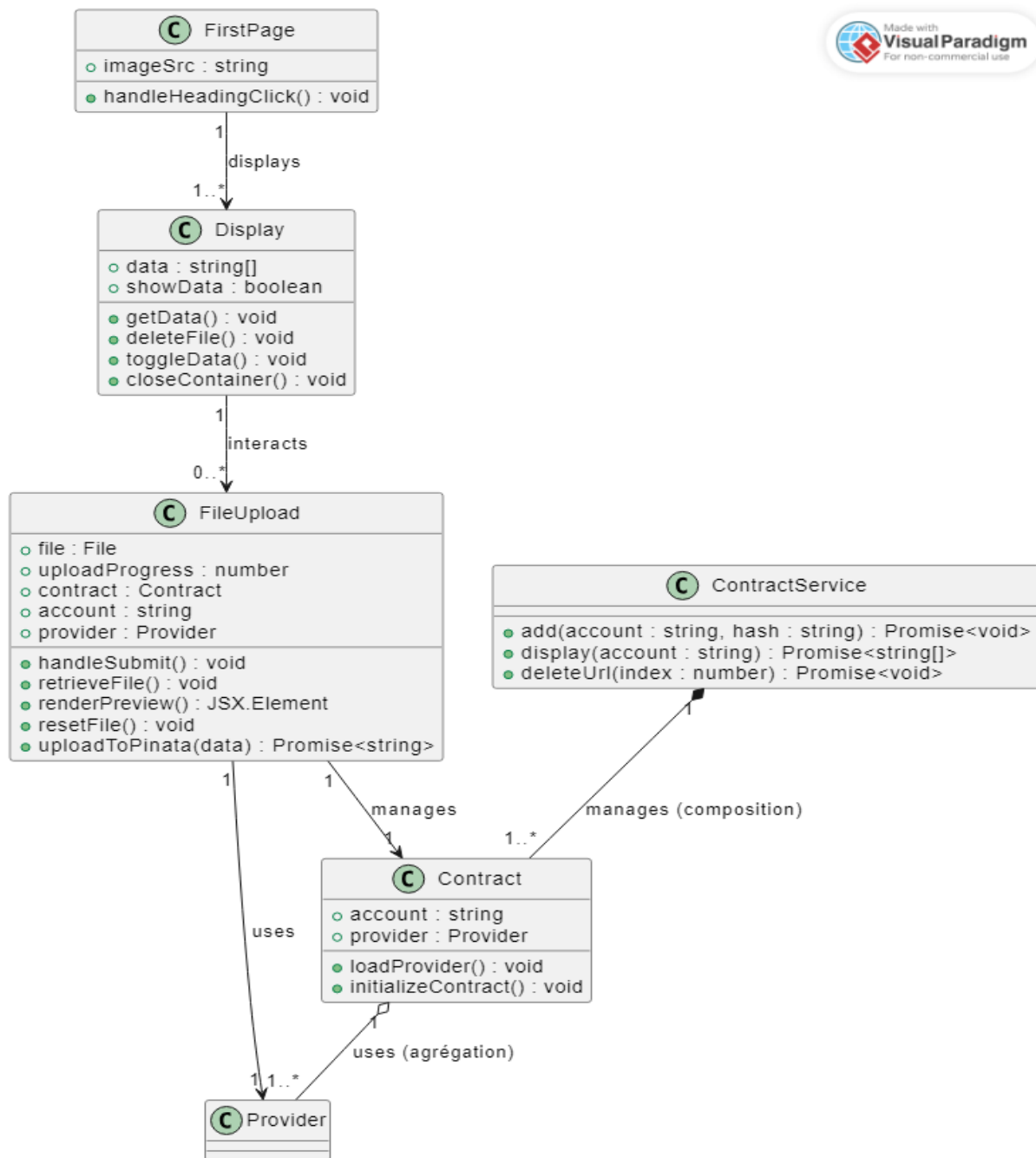


Figure 9 : Diagramme de Classe

Chapitre2 : Présentation de la réalisation

1 Présentation du projet UpShare :

1.1 Motivation et problématique du projet :

Les applications centralisées sont souvent utilisées pour gérer des données, mais elles présentent de gros risques, surtout en matière de sécurité. En mettant toutes les informations dans un seul système, le risque d'attaques informatiques augmente beaucoup. Si un pirate réussit à entrer dans le système, il peut accéder à des données sensibles. Cela peut entraîner la perte d'informations importantes, ce qui est dangereux pour l'organisation.

Un autre problème avec les applications centralisées est qu'elles dépendent d'une bonne connexion internet et de serveurs qui doivent toujours être opérationnels. Si un problème technique ou une attaque se produit, cela peut sérieusement perturber le fonctionnement quotidien de l'organisation.

La centralisation des données signifie que l'organisation dépend d'un fournisseur ou d'une plateforme. Cela peut poser des problèmes lorsque les besoins changent ou que de nouvelles technologies apparaissent. Dans ce cas, il devient difficile d'adapter le système.

Pour éviter ces risques, il est important de mettre en place de bonnes stratégies de gestion des risques et de sauvegarde des données. Sinon, les organisations qui utilisent des systèmes centralisés peuvent rencontrer des problèmes graves qui pourraient nuire à leur fonctionnement. C'est pourquoi des solutions comme la blockchain, qui permet de stocker les données de manière distribuée, sont de plus en plus considérées pour sécuriser les informations et les rendre plus accessibles.

Architecture centralisée

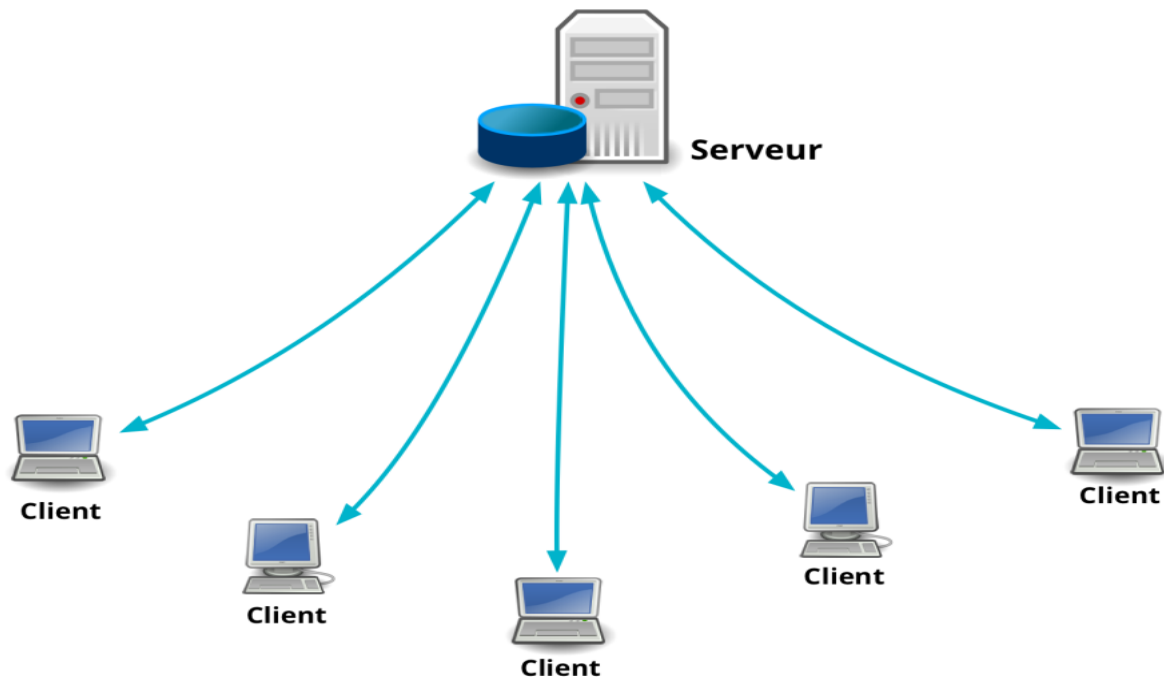


Figure 10 : Architecture centralisée

Les risques liés aux applications centralisées :

Les applications centralisées, bien qu'elles offrent des avantages en termes de gestion et d'organisation des données, présentent également de nombreux risques qui peuvent compromettre leur efficacité et la sécurité des informations. Voici les principaux risques associés à ce modèle :

- **Piratage** : Les systèmes centralisés sont des cibles privilégiées pour les cyberattaques. Une attaque réussie peut exposer un grand nombre de données sensibles, entraînant des conséquences graves pour l'organisation.
- **Perte de données** : Les erreurs humaines, les failles de sécurité ou des défaillances techniques peuvent entraîner la disparition totale des données stockées. La perte d'informations critiques peut nuire à la continuité des opérations.

- **Problèmes de réseau** : Les applications centralisées dépendent d'une connexion internet stable. En cas de panne de réseau ou d'attaque informatique, l'accès aux données peut être bloqué, perturbant ainsi les activités quotidiennes.
- **Défaillances techniques** : Si le serveur rencontre une panne, cela peut immobiliser tout le système. Les utilisateurs peuvent alors se retrouver dans l'incapacité d'accéder aux informations nécessaires, ce qui peut provoquer des retards et des frustrations.
- **Dépendance à un fournisseur** : L'utilisation d'une seule plateforme centralisée crée une dépendance. Cela limite les options disponibles pour l'organisation et peut restreindre la flexibilité en cas de changement des besoins.
- **Problèmes juridiques** : Centraliser les données peut engendrer des problèmes de conformité aux lois sur la protection des données, notamment dans le contexte de réglementations strictes. Cela expose l'organisation à des sanctions potentielles.
- **Accès non autorisé** : Si les identifiants de connexion sont volés ou mal gérés, des individus non autorisés peuvent accéder aux données sensibles. Cela constitue un risque sérieux pour la sécurité.
- **Manque de résilience** : En cas de catastrophe, le système centralisé peut ne pas être en mesure de récupérer rapidement, ce qui entraîne des interruptions prolongées.
- **Coûts élevés** : La sécurisation et la maintenance d'un système centralisé nécessitent des ressources financières et humaines considérables. Les coûts associés peuvent peser lourdement sur le budget de l'organisation.
- **Saturation** : Lorsque trop d'utilisateurs tentent d'accéder simultanément au système, cela peut entraîner une saturation, ralentissant ainsi les performances et affectant l'expérience utilisateur.

1.2 Solution

Pour résoudre les problèmes liés aux systèmes de stockage centralisés, j'ai conçu une solution décentralisée qui repose sur des technologies modernes pour assurer la sécurité et la fiabilité du stockage de fichiers. Ma solution utilise principalement la technologie IPFS, qui permet de sauvegarder et partager des fichiers directement entre utilisateurs, sans passer par un serveur centralisé.

Dans les systèmes centralisés, toutes les données sont stockées sur un seul serveur, ce qui présente des risques élevés de perte de données ou de piratage. Avec IPFS, les fichiers sont répartis sur plusieurs ordinateurs, appelés nœuds, réduisant ainsi le risque de perte. Même si un serveur tombe en panne, les fichiers restent accessibles via d'autres nœuds. Cela garantit la disponibilité continue des données.

J'ai également intégré Pinata à ma solution, un service qui "pinn" les fichiers sur IPFS pour qu'ils restent disponibles en ligne, même si le nœud principal est déconnecté. Pinata renforce donc la fiabilité et la sécurité des fichiers.

L'interface utilisateur de mon application est développée avec ReactJS, une technologie qui rend l'utilisation de l'application simple et fluide. Les utilisateurs peuvent ainsi télécharger et partager des fichiers facilement et en temps réel.

Pour sécuriser et gérer les transactions, j'utilise la blockchain. Ganache, un environnement blockchain local, pour tester les contrats intelligents avant leur déploiement sur la blockchain publique. MetaMask, une extension de navigateur, est utilisée pour faciliter les transactions sécurisées, permettant aux utilisateurs de gérer leurs clés privées et d'interagir avec la blockchain en toute sécurité.

Pourquoi utiliser la blockchain ?

La blockchain est utilisée dans cette solution pour son efficacité en matière de sécurité et de gestion des données décentralisées. Contrairement aux systèmes centralisés où toutes les informations sont stockées sur un seul serveur, la blockchain répartit les données sur un réseau de plusieurs nœuds. Cela réduit considérablement les risques de piratage ou de perte de données.

La blockchain permet de valider les transactions de manière sécurisée. Cela assure que toutes les actions sont enregistrées de manière immuable, ce qui renforce la confiance des utilisateurs dans le système. Grâce à cette technologie, les données sont non seulement mieux protégées contre les attaques, mais elles restent également accessibles en permanence, même en cas de problème technique avec un serveur.

2 Mise en œuvre du projet :

2.1 Architecture logicielle et technique :

L'architecture décentralisée est une façon d'organiser un système où les données ou les services ne sont pas contrôlés par un seul endroit central, mais sont plutôt répartis entre plusieurs points ou nœuds.

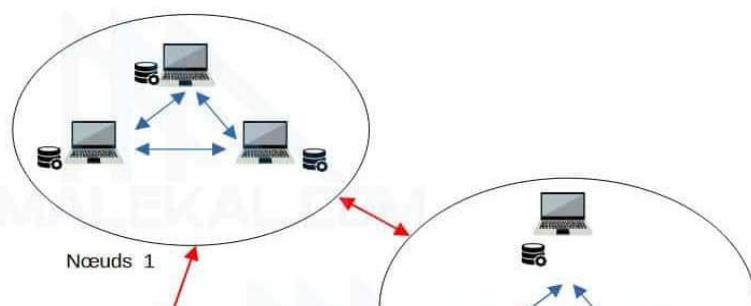


Figure 11 : Architecture Décentralisée

Principe Clés de l'Architecture Décentralisée :

Répartition des Données :

Au lieu de stocker toutes les données sur un seul serveur, les informations sont réparties entre plusieurs ordinateurs ou nœuds. Cela réduit le risque de perte totale de données si un seul point de défaillance survient.

Autonomie des Nœuds :

Chaque nœud ou ordinateur dans le réseau fonctionne de manière autonome. Ils coopèrent pour stocker et gérer les données, mais aucun d'eux n'a le contrôle total sur le système.

Redondance et Résilience :

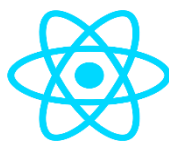
Les données sont souvent copiées et stockées sur plusieurs nœuds. Ainsi, même si certains nœuds échouent ou se déconnectent, les données restent accessibles via d'autres nœuds.

Sécurité Renforcée :

La décentralisation rend les systèmes plus résistants aux attaques. Il est plus difficile pour un attaquant de compromettre l'ensemble du réseau, car il devrait attaquer plusieurs nœuds simultanément.

2.2 Outils utilisés :

Les langages et les outils de programmation utilisés pour la mise en place de l'application sont présentés ci-après :



ReactJS :

ReactJS est une bibliothèque JavaScript développée par Facebook qui facilite la création d'interfaces utilisateurs dynamiques et réactives. L'un des principes fondamentaux de React est la modularité à travers la création de composants réutilisables, qui agissent comme des blocs de construction pour les applications web. Chaque composant représente une partie de l'interface utilisateur, ce qui permet de concevoir des interfaces plus complexes tout en conservant une structure claire et maintenable.

Ce qui distingue React des autres bibliothèques est son utilisation d'un DOM virtuel. Plutôt que de recharger toute la page à chaque modification, React met à jour uniquement les éléments qui ont changé. Cela se traduit par une amélioration significative de la rapidité et de la performance des applications web, en minimisant les accès au DOM réel, souvent coûteux en termes de performance.

Un autre avantage de React est son flux de données unidirectionnel. Cela signifie que les données circulent dans une seule direction, du parent vers les enfants, rendant ainsi le comportement de l'application plus prévisible et simplifiant le processus de débogage.



Hardhat :

Hardhat est un cadre de développement innovant conçu spécifiquement pour la création d'applications décentralisées sur la blockchain Ethereum. Il se distingue par son approche centrée sur l'expérience développeur, offrant une multitude d'outils qui simplifient le processus de développement, de déploiement et de test des contrats intelligents.

Un des points forts de Hardhat est sa flexibilité. Il permet aux développeurs d'intégrer facilement des plugins, ce qui leur permet d'ajuster l'environnement de développement selon leurs besoins. Cela inclut des outils pour l'analyse des performances des contrats, la vérification de code, et la connexion à des réseaux de test. Cette modularité assure que chaque projet peut être personnalisé et optimisé pour une expérience de développement fluide.

La console de Hardhat offre une interface interactive, permettant d'interagir directement avec la blockchain. Cela permet aux développeurs d'exécuter des scripts, de gérer les contrats et de tester les transactions dans un environnement contrôlé. Ce niveau de contrôle facilite le débogage et la mise au point des contrats intelligents avant leur déploiement sur le réseau principal.

Hardhat inclut également un simulateur de blockchain appelé Hardhat Network. Ce dernier permet de créer un environnement local où les développeurs peuvent tester leurs applications sans frais de transaction. Avec des fonctionnalités comme la gestion des comptes et le débogage avancé, Hardhat Network simplifie le processus de test, rendant le développement à la fois rapide et efficace.



Ganache :

Ganache est un simulateur de blockchain Ethereum qui joue un rôle crucial dans le développement d'applications décentralisées. Cet outil permet aux développeurs de tester leurs applications et contrats intelligents dans un environnement contrôlé et local, ce qui est

essentiel pour s'assurer que tout fonctionne correctement avant de déployer les contrats sur le réseau principal d'Ethereum.

L'un des principaux avantages de Ganache est sa capacité à créer une blockchain locale. Cela signifie que les développeurs peuvent configurer un réseau simulé avec des comptes et des tokens fictifs. Grâce à cette fonctionnalité, il est possible de simuler des transactions, d'observer les comportements des contrats intelligents et de tester les différents aspects de l'application sans encourir de coûts réels.

Ganache fournit également une interface conviviale qui permet de visualiser l'état de la blockchain, d'analyser les transactions et de vérifier l'exécution des contrats. Cette visibilité aide à identifier et résoudre les problèmes potentiels avant que le code ne soit mis en production.



MetaMask :

MetaMask est un portefeuille numérique sous forme d'extension de navigateur, conçu pour interagir avec la blockchain Ethereum. Il sert d'interface utilisateur permettant de gérer des actifs numériques, tels que les tokens et les NFT, ainsi que d'interagir avec des applications

décentralisées (d'Apps). Grâce à sa simplicité d'utilisation et à ses fonctionnalités robustes, MetaMask est devenu un outil essentiel pour quiconque s'intéresse à l'écosystème Ethereum.

Une des principales fonctions de MetaMask est la gestion des clés privées. En sécurisant ces clés, MetaMask facilite les transactions, permettant aux utilisateurs de se connecter à diverses applications Ethereum sans avoir besoin d'un nœud complet. Cela simplifie considérablement l'expérience de l'utilisateur, car il peut effectuer des transactions et accéder à des services en ligne sans avoir à se soucier des détails techniques complexes.

MetaMask est particulièrement populaire pour son intégration fluide avec des plateformes d'échange décentralisées, des jeux blockchain, et d'autres services DeFi (finance décentralisée). Cette intégration permet aux utilisateurs d'accéder facilement à une variété d'applications tout en maintenant un contrôle total sur leurs actifs numériques.



Solidity :

Solidity est le langage de programmation utilisé pour écrire des contrats intelligents sur la blockchain Ethereum. Inspiré de langages comme JavaScript et C++, il permet aux développeurs de créer des programmes auto-exécutables qui régissent les transactions et les

interactions sur la blockchain. Grâce à sa syntaxe accessible, Solidity est devenu le langage privilégié pour le développement de solutions décentralisées.

Les contrats intelligents sont des accords automatisés qui s'exécutent sans intermédiaire. Cela signifie que les conditions d'un contrat sont codées directement dans le programme, permettant ainsi d'exécuter des transactions en toute transparence et sans risque d'erreur humaine. Les applications de Solidity sont vastes et incluent des domaines tels que les services financiers, les systèmes de vote, les jeux, et bien plus encore.

L'un des atouts majeurs de Solidity est sa capacité à gérer des types complexes et des structures d'interaction sophistiquées. Les développeurs peuvent définir des fonctions, des événements et des structures de données personnalisées, ce qui offre une flexibilité considérable lors de la création de contrats intelligents. Cette puissance permet également de concevoir des applications plus robustes et adaptées aux besoins spécifiques des utilisateurs.



JSX est une extension syntaxique qui permet de combiner le code JavaScript et le HTML dans les fichiers React. Cette approche innovante facilite la création d'interfaces utilisateurs en offrant aux développeurs la possibilité d'écrire des composants avec une syntaxe proche du HTML. En intégrant la simplicité du HTML avec la puissance de JavaScript, JSX permet de gérer efficacement les données et la logique des applications.

L'un des principaux avantages de JSX est qu'il rend le développement d'interfaces utilisateurs plus intuitif. En permettant aux développeurs de structurer à la fois l'interface et la logique au même endroit, JSX réduit la complexité et améliore la lisibilité du code. Cela permet de mieux comprendre comment les composants interagissent et se rendent plus accessibles, même pour ceux qui ne sont pas familiers avec le langage.

Sous le capot, le code JSX est transformé en appels JavaScript qui optimisent les performances des applications. Ce processus de transformation est effectué par des outils comme Babel, qui convertissent le code JSX en JavaScript classique, garantissant ainsi que les navigateurs peuvent interpréter et exécuter le code sans problème. Cela permet de tirer parti des dernières fonctionnalités de JavaScript tout en continuant à utiliser une syntaxe familière.



Pinata :

Pinata est un service essentiel qui permet de stocker et de gérer des fichiers sur le réseau IPFS (InterPlanetary File System). Il facilite l'accès et l'utilisation d'IPFS pour les développeurs et les créateurs, en offrant une interface utilisateur conviviale et des outils pratiques pour "pinner" des fichiers. Ce processus de "pinning" garantit la disponibilité continue des fichiers sur le réseau décentralisé, évitant ainsi tout risque de perte d'accès.

L'une des caractéristiques marquantes de Pinata est sa simplicité d'utilisation. Grâce à une interface intuitive, les utilisateurs peuvent facilement télécharger, gérer et partager leurs fichiers sur IPFS, sans avoir à se soucier des complexités techniques sous-jacentes. Cela en fait un outil particulièrement attrayant pour ceux qui souhaitent tirer parti des avantages de la décentralisation sans se plonger dans les détails techniques.

Pinata est particulièrement populaire dans les projets liés aux Nfts (tokens non fongibles) et aux contenus numériques. En offrant un moyen simple de stocker et de distribuer des actifs numériques en toute sécurité, Pinata permet aux créateurs de s'assurer que leurs œuvres restent accessibles et protégées, sans dépendre de serveurs centralisés qui peuvent être vulnérables aux pannes ou aux attaques.



IPFS :

IPFS, ou « Inter Planetary File System », est un protocole innovant de stockage et de partage de fichiers qui fonctionne de manière décentralisée. Contrairement aux systèmes traditionnels qui hébergent des données sur des serveurs centralisés, IPFS distribue les fichiers sur un réseau pair-à-pair. Cette approche permet de rendre les données plus accessibles et résistantes à la censure.

Chaque fichier sur IPFS est identifié par un hash unique, ce qui garantit l'intégrité et l'immutabilité du contenu. Grâce à cette caractéristique, il est impossible de modifier un fichier sans changer son hash, assurant ainsi que les utilisateurs accèdent toujours à la version originale des données. Ce mécanisme est essentiel pour maintenir la confiance dans les systèmes décentralisés, où la vérification des données est primordiale.

IPFS est particulièrement utile pour les projets qui nécessitent une forte disponibilité et une résistance aux pannes. Les applications basées sur la blockchain, par exemple, peuvent bénéficier de l'architecture décentralisée d'IPFS, car cela réduit le risque de perte de données en cas de défaillance d'un serveur ou d'attaque. De plus, les sites web statiques peuvent être hébergés de manière sécurisée et efficace sur IPFS, garantissant un accès rapide et fiable aux utilisateurs.



VS Code :

VS Code, ou Visual Studio Code, est un éditeur de code source gratuit et open-source, développé par Microsoft. Il est largement apprécié dans le monde du développement pour sa légèreté, sa rapidité et sa grande flexibilité. Ces caractéristiques en font un outil privilégié pour les développeurs de tous niveaux, qu'ils soient débutants ou professionnels expérimentés.

Un des atouts majeurs de VS Code est sa prise en charge d'une multitude de langages de programmation, tels que JavaScript, Python, C++, et bien d'autres. Cette polyvalence est renforcée par la possibilité d'installer facilement des extensions depuis sa marketplace intégrée, permettant aux utilisateurs d'adapter l'environnement de développement à leurs besoins spécifiques.

VS Code propose également des outils de développement puissants, comme un débogueur intégré et le contrôle de version Git, facilitant ainsi la gestion des projets et le suivi des modifications. Des fonctionnalités avancées, telles que l'autocomplétion, l'analyse de code en temps réel et la gestion des projets, contribuent à améliorer la productivité des développeurs en simplifiant les tâches courantes.

L'interface de VS Code est hautement personnalisable, offrant aux utilisateurs la possibilité de configurer l'éditeur selon leurs préférences personnelles. Cette capacité à s'adapter aux besoins individuels, associée à ses nombreuses extensions, fait de VS Code un outil incontournable pour les développeurs, quel que soit leur niveau d'expertise.



Visual Paradigm :

Visual Paradigm est un logiciel de modélisation visuelle complet qui permet aux équipes de concevoir et de planifier des systèmes logiciels de manière structurée. Il est utilisé principalement pour créer des diagrammes UML (Unified Modeling Language) comme les diagrammes de classes, de séquence, et de cas d'utilisation, qui facilitent la conception et la documentation des architectures logicielles complexes. En plus d'UML, Visual Paradigm supporte d'autres méthodologies de modélisation, telles que BPMN (Business Process Model and Notation) et ERD (Entity Relationship Diagram), ce qui le rend adapté à une large gamme de projets, allant du développement logiciel à la gestion des processus métier. Il est particulièrement utile pour les équipes travaillant sur des projets collaboratifs, car il permet de visualiser, planifier et aligner la conception et le développement à chaque étape du processus.

3 Réalisation :

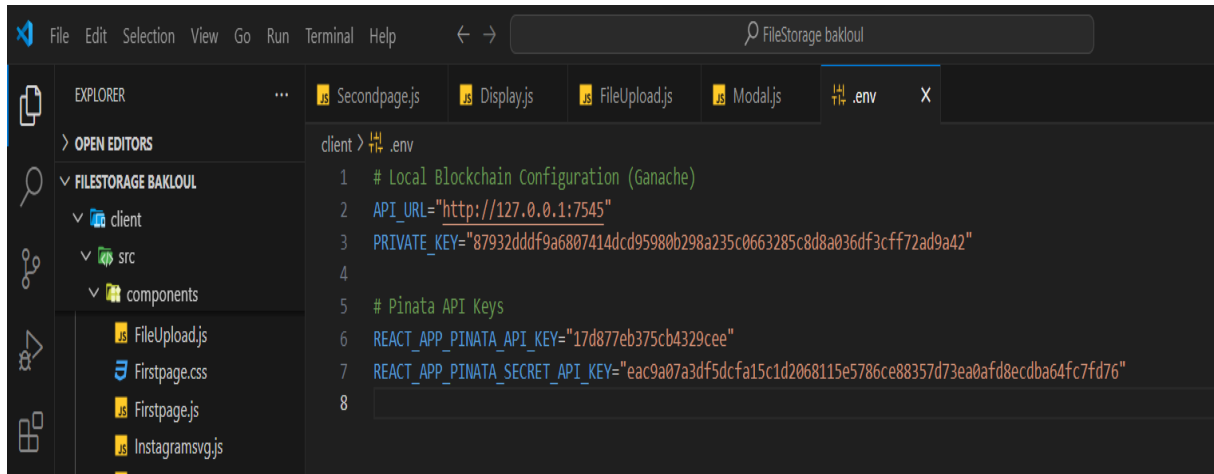
3.1 Introduction :

Dans ce chapitre, je présente la partie pratique de mon projet, en mettant l'accent sur les interfaces de l'application Web. Cette section montre comment j'ai concrétisé mes idées, mettant en avant l'atteinte de mes objectifs. En explorant les interfaces utilisateur, je souligne l'importance de l'expérience utilisateur et de l'ergonomie, qui sont essentielles pour garantir une application à la fois fonctionnelle et professionnelle. Cette partie pratique donne un aperçu détaillé de la conception et du développement, en mettant en lumière les choix techniques et esthétiques qui ont guidé chaque aspect de mon projet.

Sur ce chapitre, je vais expliquer le travail réalisé, en utilisant plusieurs figures pour mieux le décrire.

3.2 Présentation des interfaces de l'application :

J'ai développé une interface avec ReactJS pour permettre aux utilisateurs d'interagir avec l'application, tandis que Pinata et IPFS gèrent le stockage décentralisé des fichiers. Ganache et MetaMask sont intégrés pour gérer les transactions et les interactions avec la blockchain. Cette intégration implique la configuration des composants nécessaires pour assurer que l'application puisse stocker et accéder aux fichiers sur IPFS tout en interagissant de manière fluide avec la blockchain via Ganache et MetaMask.



```
client > .env
1 # Local Blockchain Configuration (Ganache)
2 API_URL="http://127.0.0.1:7545"
3 PRIVATE_KEY="87932dddf9a6807414dcd95980b298a235c0663285c8d8a036df3c7f72ad9a42"
4
5 # Pinata API Keys
6 REACT_APP_PINATA_API_KEY="17d877eb375cb4329cee"
7 REACT_APP_PINATA_SECRET_API_KEY="eac9a07a3df5dcfa15c1d2068115e5786ce88357d73ea0afd8ecd8a64fc7fd76"
8
```

Figure 12 : Code d'interaction avec Ganache et Pinata (API)

Cette figure montre le compte MetaMask et les transactions confirmées avec succès.

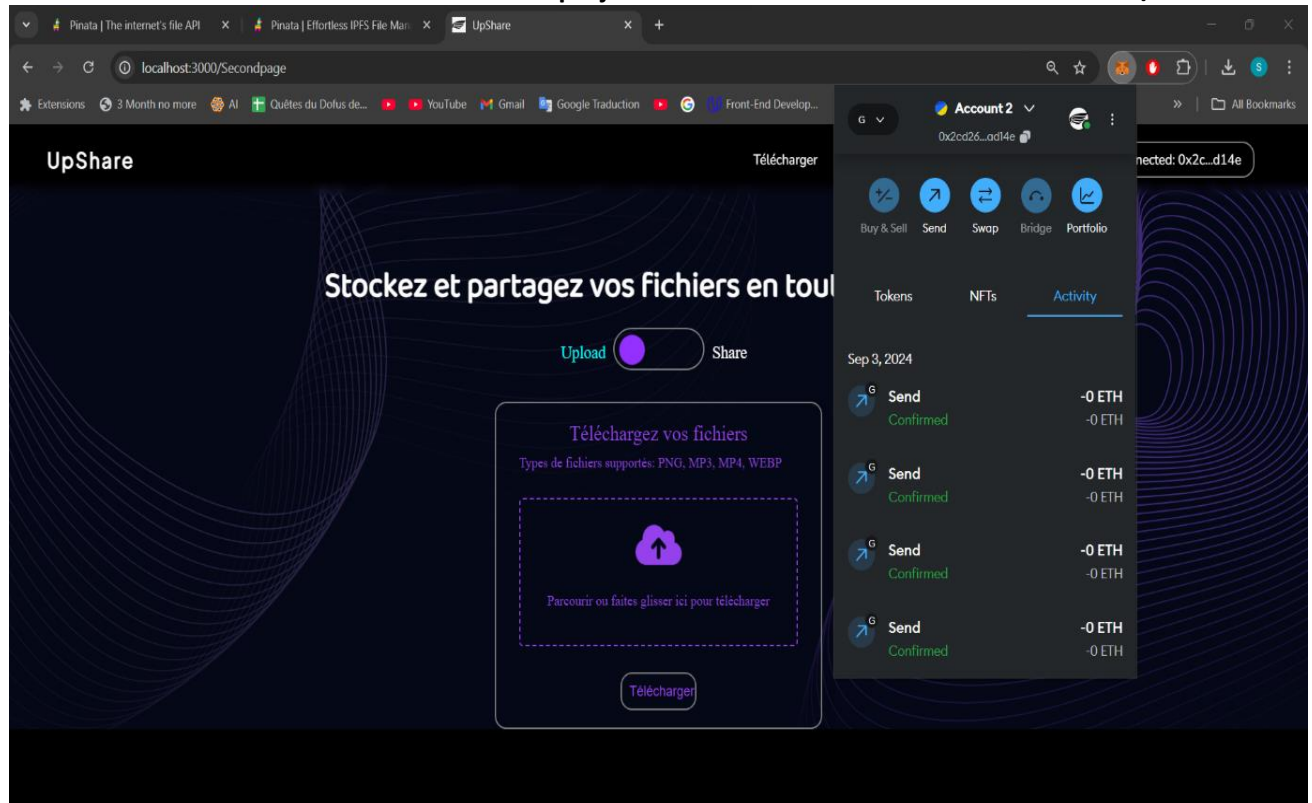


Figure 13 : Compte MetaMask et transaction appliqué

Vous voyez une liste de comptes Ethereum avec leurs adresses et soldes affichés. La section des transactions récentes montre les détails des transactions, ainsi que leur statut, comme réussi ou en attente. Vous avez aussi la possibilité de déployer des contrats intelligents et de voir ceux qui sont déjà actifs. La console en bas de l'écran affiche les journaux d'activité et signale les erreurs éventuelles, ce qui vous aide à suivre ce qui se passe et à résoudre les problèmes. En plus, vous pouvez ajuster les paramètres pour personnaliser la blockchain locale afin qu'elle réponde aux besoins spécifiques de votre développement.

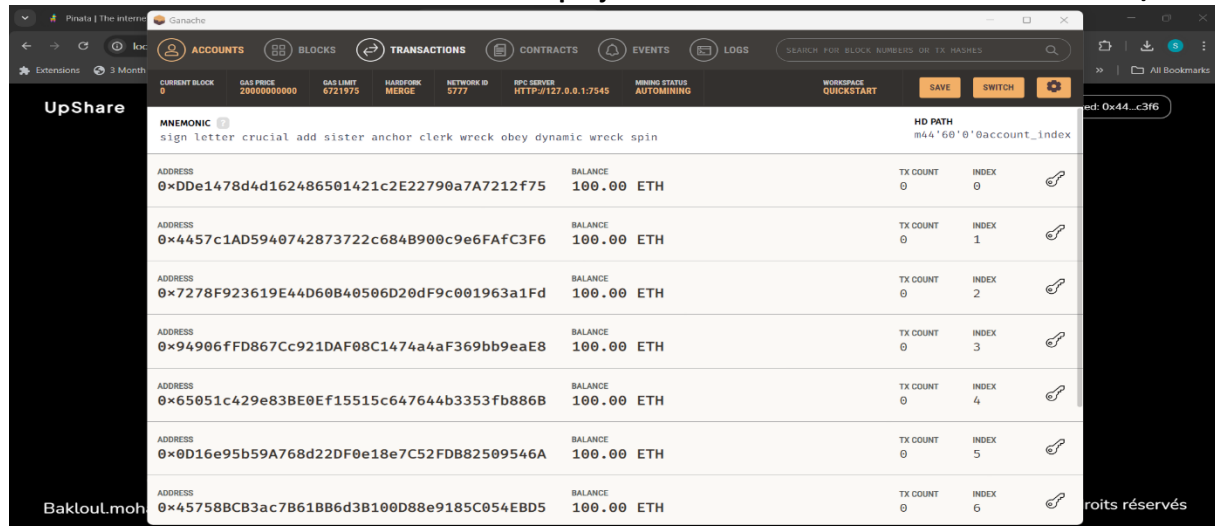


Figure 14 : Interface Ganache et Adresse Public Utilisé

Ganache est un simulateur de blockchain qui offre une interface conviviale pour le développement et le test d'applications décentralisées. Voici une description des sections principales de Ganache :

Blocs : La section des blocs affiche la liste des blocs générés dans la blockchain locale. Chaque bloc contient un ensemble de transactions validées, un horodatage et un hash unique qui garantit l'intégrité des données.

Transactions : Cette section montre les détails de chaque transaction effectuée sur la blockchain. Pour chaque transaction, des informations telles que l'adresse de l'expéditeur, l'adresse du destinataire, le montant transféré et les frais de transaction sont visibles.

Contrats : Dans cette section, les contrats intelligents déployés sur la blockchain sont listés. Elle fournit des informations détaillées sur chaque contrat, y compris l'adresse du contrat, le créateur et le bytecode du contrat.

Événements : Les événements émis par les contrats intelligents sont enregistrés dans cette section. Les événements servent de notifications que les contrats peuvent émettre pour signaler des actions spécifiques.

Journaux (Logs) : La section des journaux affiche les logs détaillés des opérations effectuées sur la blockchain. Ces logs incluent les appels de fonction, les résultats des transactions et les erreurs éventuelles.

Le site Web de Pinata pour IPFS offre une interface simple et fonctionnelle. En haut de la page, vous trouvez un menu pour naviguer entre les différentes sections. Sur la page d'accueil, j'ai un aperçu des fichiers que j'ai ajoutés. Je peux facilement télécharger de nouveaux fichiers, consulter ceux déjà stockés, et gérer les options de « pinning ». L'interface est conçue pour être claire et facile à utiliser, ce qui simplifie la gestion de mes fichiers sur IPFS.

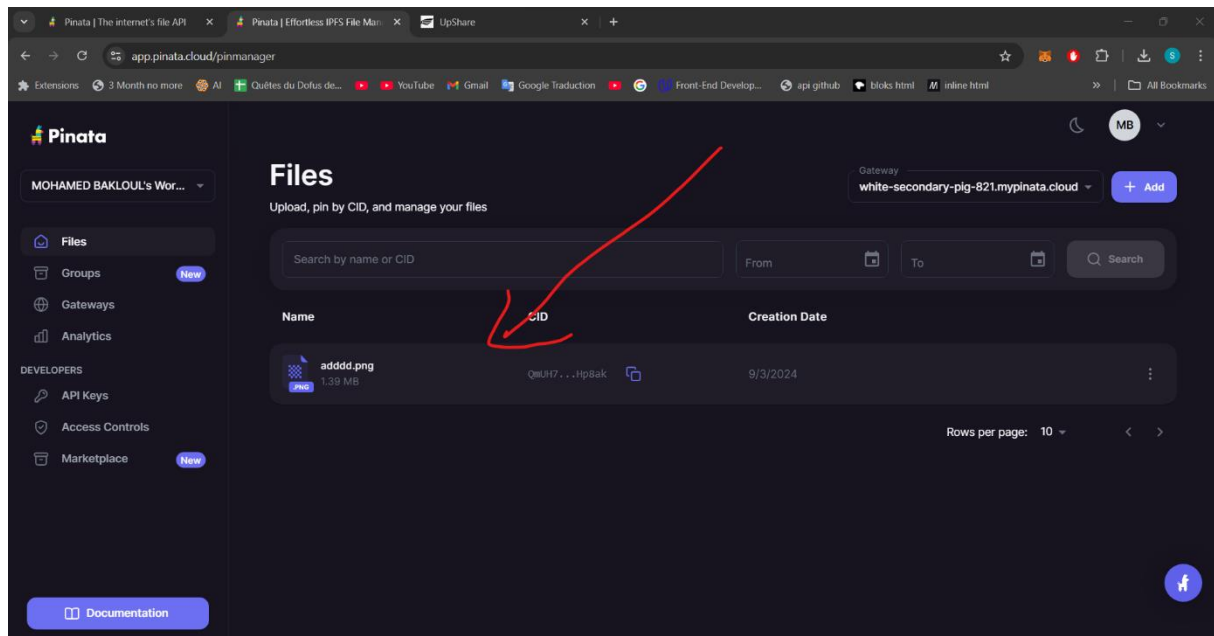


Figure 15 : Pinata Interface et fichier Télécharger

Aperçu des Fonctionnalités Clés de l'Application Pinata

Pinata est un service essentiel pour le stockage et la gestion de fichiers sur le réseau IPFS. Voici un aperçu de ses fonctionnalités clés :

- **Files** : Cette section permet de gérer les fichiers que vous avez téléchargés sur IPFS via Pinata. Vous pouvez visualiser les détails de chaque fichier, y compris son empreinte de hachage (hash), sa taille et d'autres métadonnées. Cela facilite le suivi et l'organisation de vos fichiers stockés de manière décentralisée.

- **Gateways** : Les passerelles sont des points d'accès pour accéder aux données hébergées sur IPFS. Pinata vous permet de gérer les paramètres de vos passerelles, ce qui peut influencer la disponibilité et la vitesse de récupération de vos fichiers IPFS.
- **Frames Analytics** : Cette section fournit des analyses détaillées sur la manière dont vos fichiers IPFS sont utilisés et consultés à travers les cadres (frames) web. Vous pouvez obtenir des informations sur les accès, les téléchargements et d'autres métriques pertinentes, vous aidant à comprendre l'utilisation de vos fichiers.
- **Analytics** : En plus des analyses spécifiques aux cadres, Pinata offre des analyses plus générales sur l'utilisation de votre compte, y compris le nombre de fichiers téléchargés, la bande passante utilisée, etc.
- **API Keys** : Cette section permet de gérer les clés d'API utilisées pour accéder aux fonctionnalités de Pinata via des applications tierces ou des scripts. Vous pouvez générer de nouvelles clés, révoquer les anciennes et gérer les autorisations associées. Cela facilite l'intégration de Pinata dans vos propres projets et applications.

La page d'accueil affiche un bouton "Cliquez ici pour télécharger" pour initier le téléchargement des fichiers et une icône indiquant la connexion active avec MetaMask.

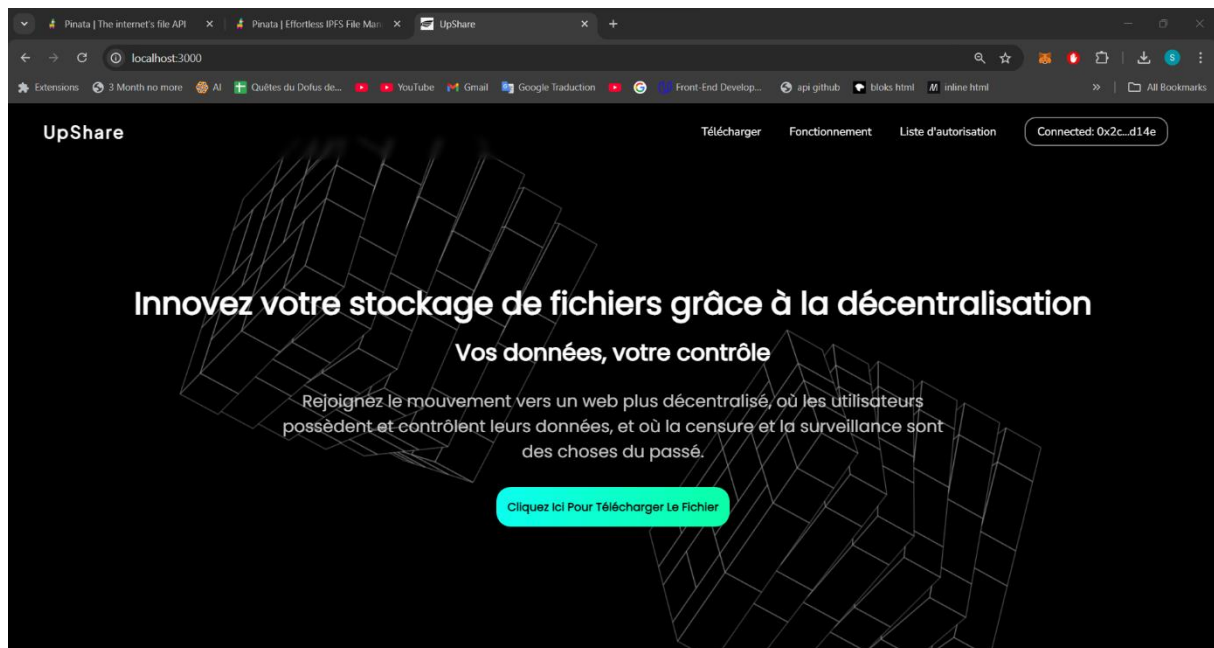


Figure 16 : Home Page UpShare

La page d'accueil de l'application décentralisée (D'App) UpShare est conçue pour s'adapter dynamiquement à l'état de connexion du portefeuille MetaMask de l'utilisateur, garantissant ainsi une expérience fluide et intuitive.

Lorsqu'un visiteur arrive sur la page sans être connecté à son compte MetaMask, un message clair s'affiche : « Connect to MetaMask ». Cette invitation guide les utilisateurs vers la connexion de leur portefeuille, assurant ainsi un accès sécurisé à toutes les fonctionnalités de l'application.

Une fois que l'utilisateur se connecte avec succès à MetaMask, la page d'accueil se transforme pour afficher un bouton qui dirige vers sa page de téléchargement. Cette fonctionnalité dynamique est essentielle pour orienter efficacement les utilisateurs vers la section appropriée de l'application, améliorant ainsi la navigation et l'accessibilité.

En résumé, la conception adaptative de la page d'accueil d'UpShare permet non seulement d'assurer la sécurité de l'accès utilisateur, mais aussi d'optimiser l'expérience globale en facilitant la transition entre les états de connexion.

En bas de la page, on trouve également une description des avantages du stockage des fichiers sur la blockchain.

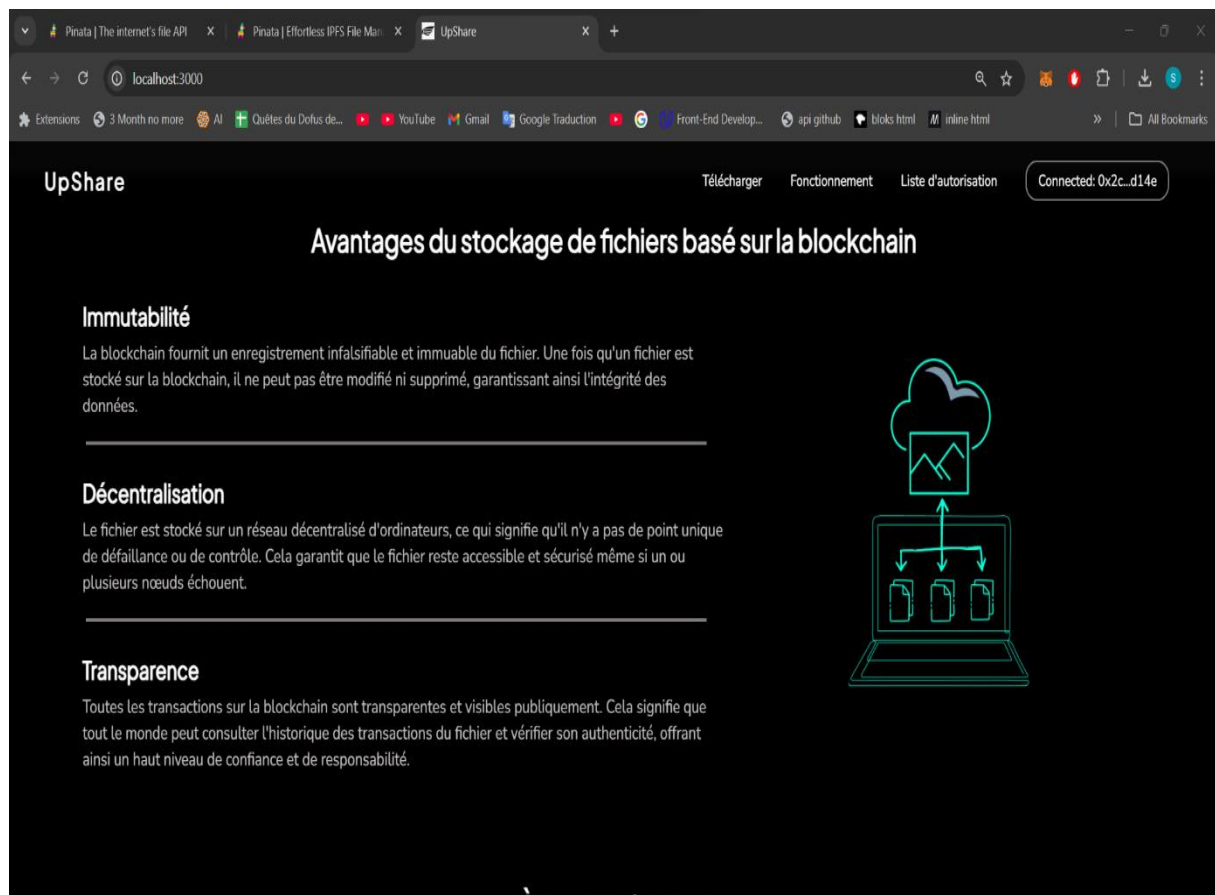


Figure 17 : Interface Home Page (Avantage du Stockage)

Ainsi que des sections "À propos de nous", "Contacts", et des liens vers les réseaux sociaux.

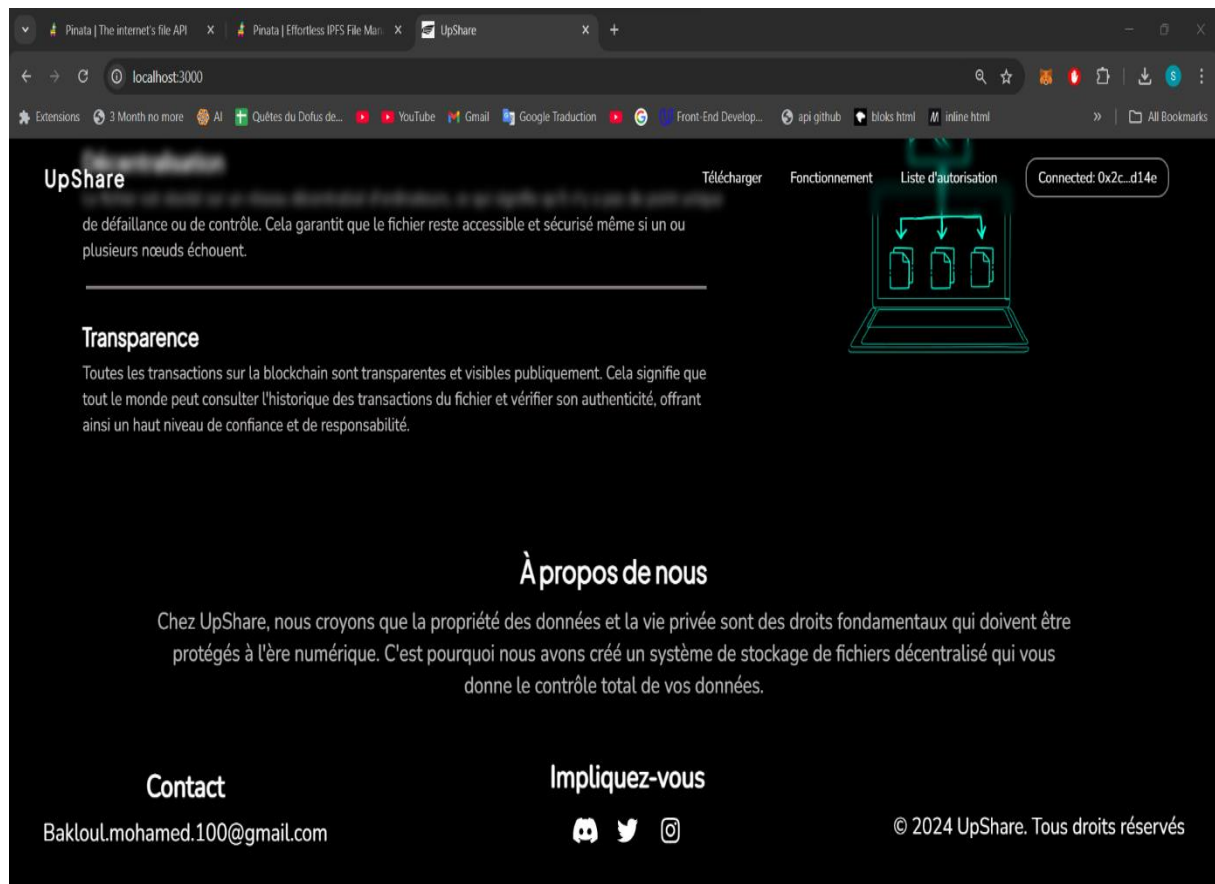


Figure 18 : Bas de page du Site web UpShare

Lorsque l'utilisateur clique sur le bouton de téléchargement sur la page d'accueil, il est redirigé vers une nouvelle page dédiée à la gestion des fichiers. Cette page est conçue pour offrir plusieurs options conviviales permettant aux utilisateurs de partager et de télécharger leurs fichiers facilement.

Sur cette page, des boutons clairement étiquetés permettent aux utilisateurs de partager ou de télécharger des fichiers en un seul clic. Cela simplifie le processus et rend l'expérience utilisateur intuitive.

En outre, la page offre une flexibilité supplémentaire en permettant aux utilisateurs de gérer leurs fichiers de deux manières. Ils peuvent soit glisser-déposer les fichiers directement depuis leur bureau dans la zone de téléchargement, soit parcourir leur système à l'aide d'une fenêtre de sélection de fichiers. Cette dualité d'options assure une accessibilité optimale, répondant ainsi aux préférences de chacun.

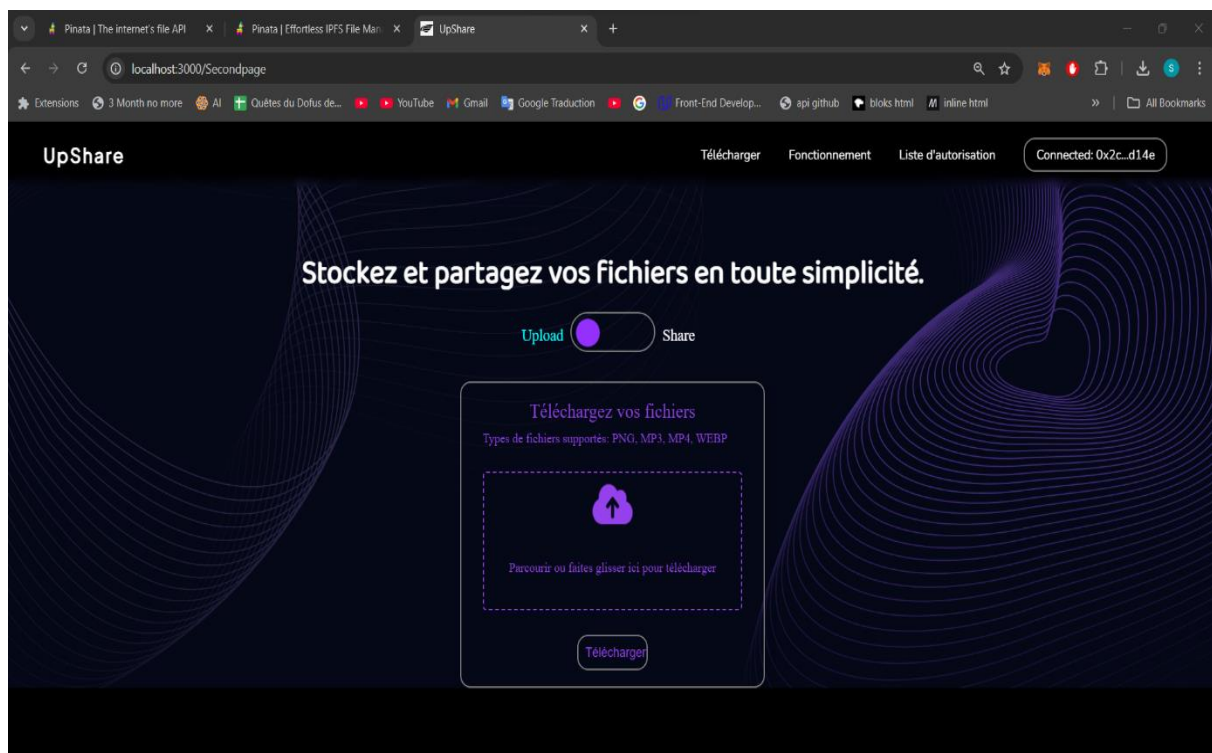


Figure 19 : Interface de Téléchargement

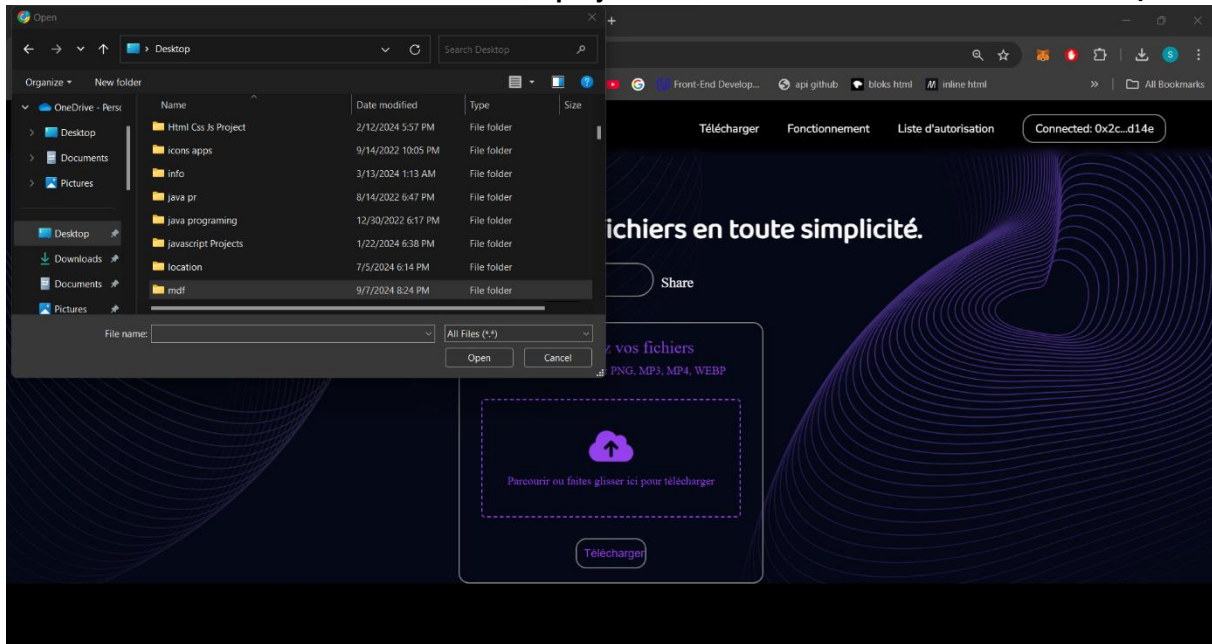


Figure 20 : Interface Parcourir un fichier

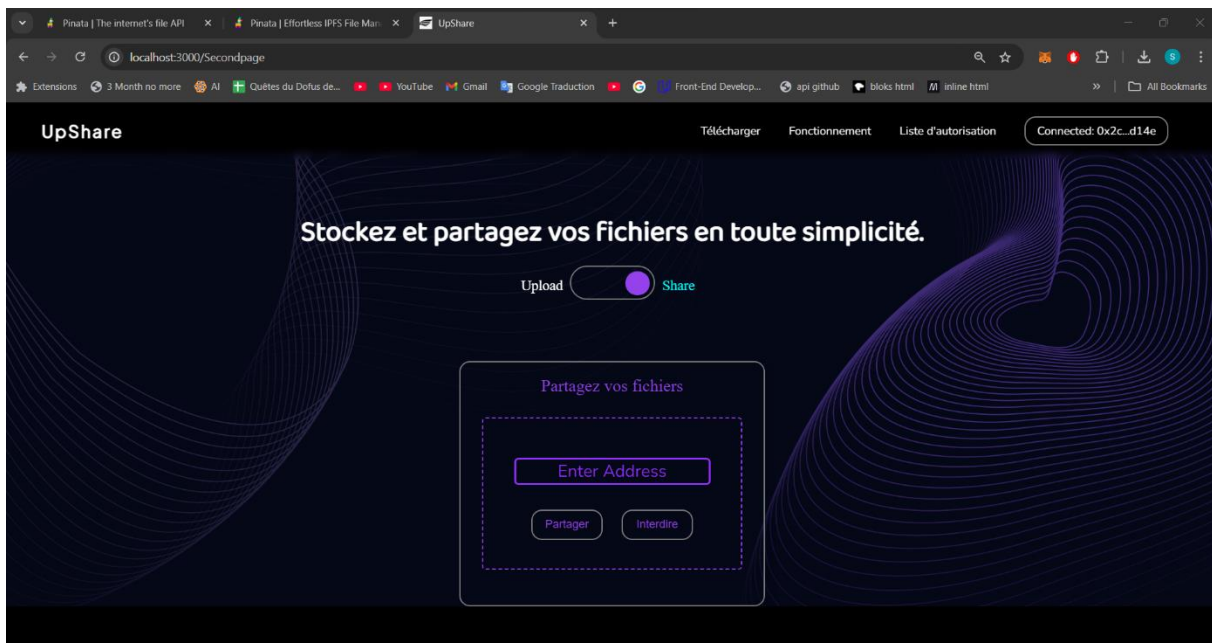


Figure 21 : Interface Partage des Fichiers

En bas de la page de téléchargement, une section "Quels services nous offrons" présente les différents services proposés, accompagnée d'une section pour les informations de contact. Cette page, intitulée "Mes téléchargements", permet aux utilisateurs de partager ou de télécharger des fichiers, tout en offrant des options simples pour gérer les fichiers via glisser-déposer ou navigation sur le bureau.

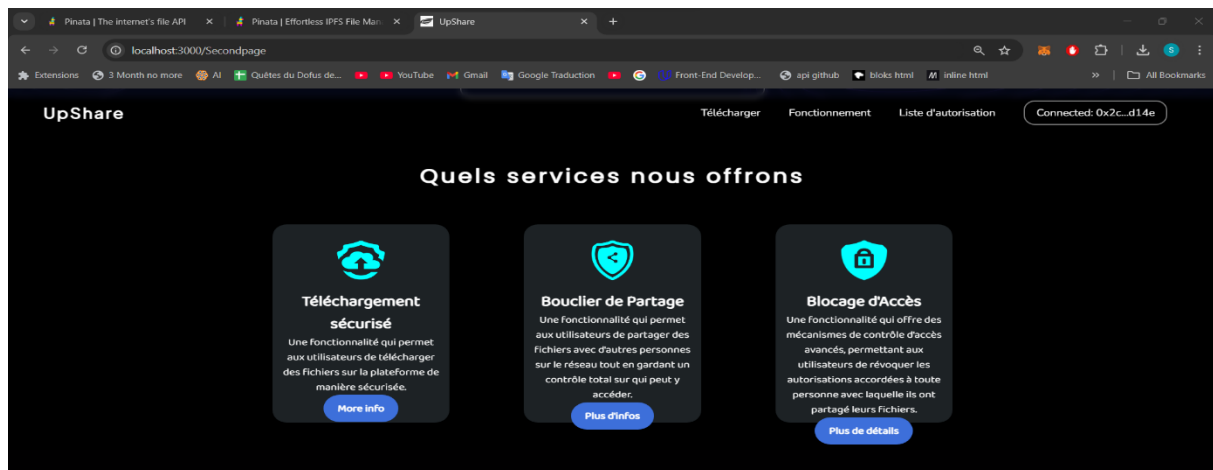


Figure 22 : Interface de service du UpShare

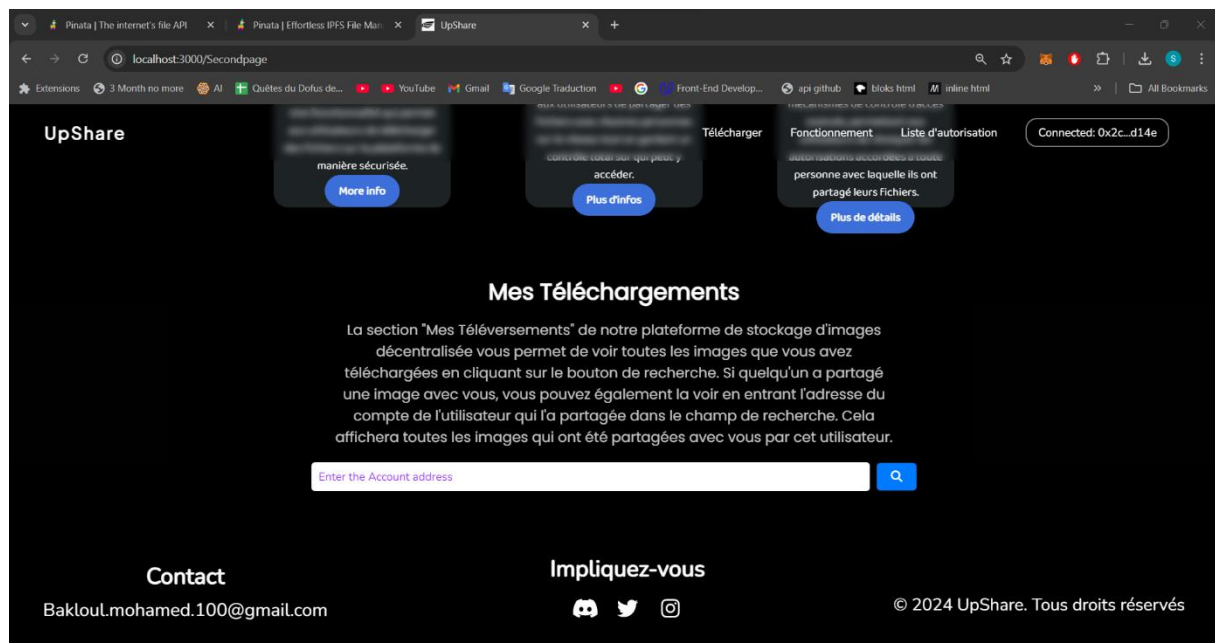


Figure 23 : Interface de consultation des fichiers partager par accès

La page Liste d'accès offre une interface complète pour la gestion des permissions d'accès. En haut de la page, la barre de navigation permet une navigation fluide vers d'autres sections du site. Au centre de la page, un formulaire permet d'ajouter des adresses pour gérer leurs accès : les utilisateurs peuvent entrer une adresse et cliquer sur le bouton "Allow" pour accorder l'accès. La liste des adresses actuelles est affichée avec leur statut (autorisé ou non autorisé), accompagné de boutons pour modifier leur statut.

Enfin, le pied de page fournit des informations de contact, des icônes de réseaux sociaux pour l'engagement, et des informations sur les droits d'auteur, offrant une vue d'ensemble complète de la gestion des accès et des moyens de contact et d'engagement.

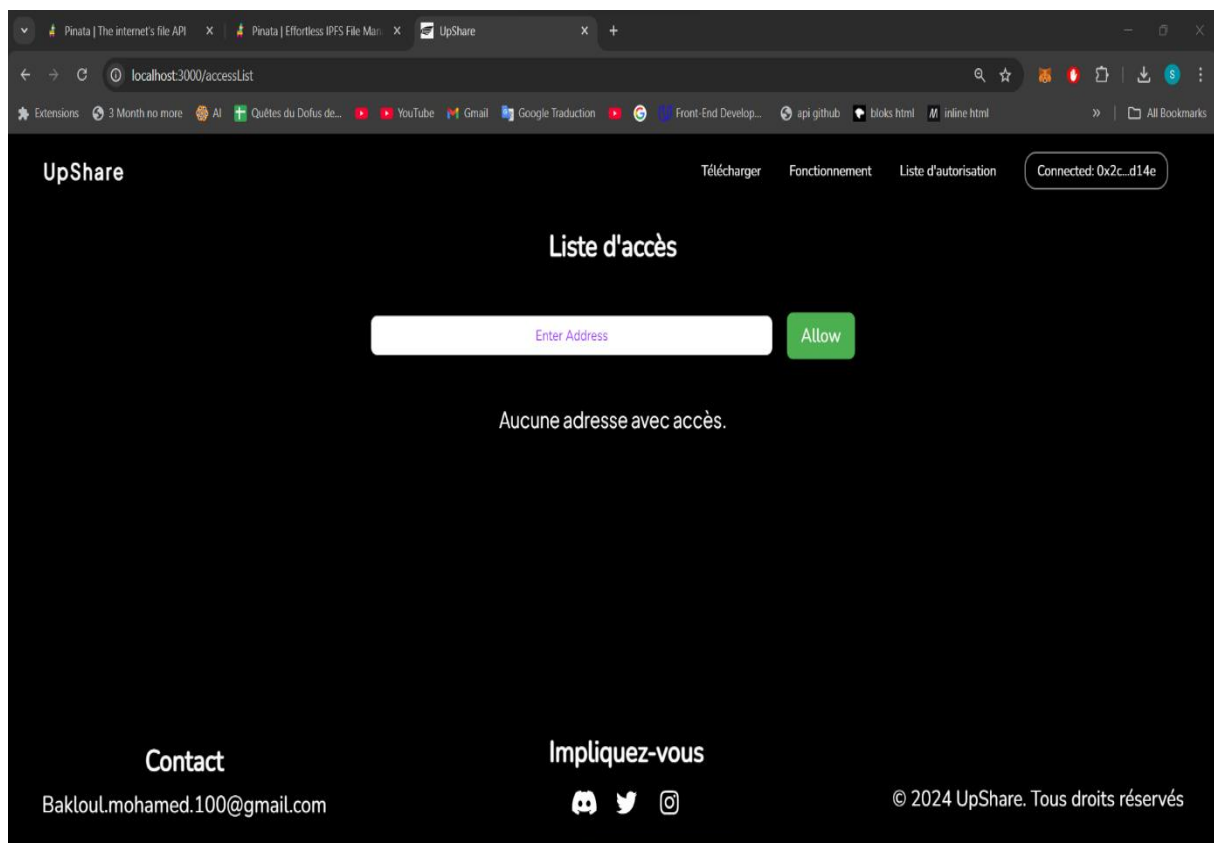


Figure 24 : Interface Liste d'accès

3.3 Résumé du processus d'Ajout d'un Fichier :

L'utilisateur commence par se connecter au site Web d'UPshare, s'assurant que MetaMask est correctement configuré pour se connecter au réseau local de Ganache. Cette étape est cruciale, car elle garantit que toutes les transactions effectuées pendant l'utilisation de l'application sont sécurisées et se déroulent sur un environnement de développement contrôlé.

Une fois connecté, l'utilisateur accède à la page de téléchargement, qui lui propose une interface conviviale pour choisir un fichier depuis son ordinateur. En sélectionnant le fichier désiré, il clique ensuite sur le bouton de téléchargement. Ce bouton déclenche un processus automatisé où le fichier est immédiatement envoyé à Pinata pour être stocké sur IPFS.

L'application utilise un contrat intelligent qui joue un rôle clé dans ce processus. Ce contrat intelligent enregistre les informations relatives au fichier sur la blockchain, garantissant ainsi l'intégrité et la traçabilité des données. À cette étape, l'utilisateur doit signer une transaction via MetaMask. Cette action permet d'établir un lien sécurisé entre l'utilisateur et le contrat intelligent, assurant que toutes les opérations sont transparentes et vérifiables.

Une fois le téléchargement terminé, l'utilisateur reçoit une confirmation visuelle indiquant que son fichier a été ajouté avec succès au réseau IPFS via Pinata. De plus, il peut consulter ses fichiers soit directement sur le site Web, soit sur la plateforme Pinata, en utilisant son adresse privée pour accéder à ses données.

Ce processus bien permet à l'utilisateur d'ajouter facilement des fichiers au réseau IPFS, tout en bénéficiant d'une intégration fluide de MetaMask pour les interactions avec la blockchain. Cette approche décentralisée non seulement assure la sécurité des données, mais renforce également l'autonomie des utilisateurs en leur permettant de gérer leurs fichiers de manière transparente et efficace.

Conclusion Générale

Ce projet s'inscrit dans le cadre de ma formation en Génie logiciel à l'école marocaine d'ingénierie, et il a constitué une étape clé dans mon parcours académique. Développer une application décentralisée pour l'archivage de documents a été une expérience extrêmement instructive et enrichissante. En utilisant des technologies modernes telles que React.js, Truffle, IPFS et Pinata, j'ai pu concevoir une plateforme robuste, sécurisée et efficace pour le stockage, le partage et la gestion des documents de manière décentralisée.

React.js a joué un rôle crucial en me permettant de créer une interface utilisateur dynamique et conviviale. Cette technologie a facilité la conception d'une interface fluide qui améliore l'expérience utilisateur en rendant l'application intuitive et facile à naviguer. Truffle a simplifié le processus de développement et de déploiement des contrats intelligents sur la blockchain Ethereum. Grâce à Truffle, j'ai pu assurer la sécurité des transactions et la transparence des opérations tout en automatisant les aspects complexes du déploiement des contrats.

En parallèle, IPFS et Pinata ont offert une solution de stockage décentralisée, garantissant à la fois la sécurité des données. IPFS permet de stocker les fichiers de manière distribuée, tandis que Pinata assure leur gestion et leur accessibilité, offrant une solution fiable pour la conservation des données.

L'intégration fluide de ces technologies avec d'autres éléments de l'écosystème décentralisé a été un atout majeur. Cela a permis de créer une application cohérente et fonctionnelle, en exploitant les synergies entre les différents outils. Ce projet a non seulement renforcé mes compétences en développement web et blockchain, mais il m'a également offert une compréhension approfondie des avantages de la décentralisation dans la gestion des données.

Je suis convaincu que cette application représente une avancée importante dans le domaine des technologies décentralisées, car elle offre aux utilisateurs une autonomie accrue et une meilleure gestion de leurs données. Dans le futur, j'envisage d'ajouter de nouvelles fonctionnalités pour enrichir l'application et de développer davantage les pages existantes, afin de continuer à améliorer l'expérience utilisateur et à répondre aux besoins évolutifs de mes utilisateurs.

Référence

<https://www.realite-virtuelle.com/metamask-qu-est-ce-que-c-est/>

<https://archive.trufflesuite.com/docs/ganache/>

<https://docs.metamask.io/>

<https://hardhat.org/docs>

<https://docs.pinata.cloud/quickstart>

<https://docs.pinata.cloud/web3/account-management/limits>

<https://app.pinata.cloud/developers/api-keys>

<https://react.dev/reference/react-dom>

<https://chatgpt.com/>

<https://www.chatpdf.com/>