

SYSTEME DE GESTION DE DOCUMENTS MEDICAUX BASE SUR LA BLOCKCHAIN ET IPFS

Module : **Blockchain**
2eme Année **Master IASD**
Projet de Fin du module



Encadrée par : Pr. Ikram BENADBELOUAHAB

Réalisée par :

- Ayman Boufarhi
- Idriss Khattabi
- Chihab Eddine Liefriid
- Ammar Amzil

Contents

1. Introduction :	2
2. Architecture du Système	2
Infrastructure Blockchain	2
Intégration IPFS	3
Écosystème d'Applications	3
Application Web (ReactJS)	3
Application Desktop (Tkinter et CustomTkinter)	3
4. Conception des Contrats Intelligents	5
3.1 Contrat de Gestion des Patients	5
Inscription des Patients	5
Récupération des Informations du Patient	6
Gestion des Dossiers Médicaux	6
3.2 Contrat de Gestion des Médecins	6
Inscription des Médecins	6
Contrôle d'Accès Patient	7
Mécanismes d'Octroi et de Révocation d'Accès	7
3.3 Contrat d'Audit	7
Suivi des Actions	7
Types d'Événements Supportés	7
Transparence Administrative	7
4. Rôles et Permissions Utilisateurs	8
4.1 Administrateur (Admin)	8
4.2 Médecin	9
4.3 Patient	11
5. Mécanismes d'Authentification	12
Authentification Application Desktop	12
Authentification Application Web	13
6. Détails Techniques de l'Implémentation	13
Technologie Blockchain	13
Intégration IPFS	13
Technologies Frontend	13
Développement des Contrats Intelligents	14
7. Conclusion :	14
8. Référence :	14

1. Introduction :

À l'ère de la numérisation croissante des soins de santé et des préoccupations de confidentialité des données, notre projet présente un système innovant de gestion de documents médicaux basé sur la blockchain, qui révolutionne la façon dont les dossiers médicaux sont stockés, accessibles et gérés. En tirant parti de la technologie blockchain et du système de fichiers InterPlanétaire (IPFS), nous avons développé une plateforme sécurisée, transparente et décentralisée qui garantit l'intégrité, la confidentialité et l'accessibilité des documents médicaux.

Le système comprend deux applications principales - une application web développée avec ReactJS et une application de bureau créée à l'aide de Tkinter - qui offrent une interface complète et conviviale pour la gestion des dossiers médicaux sur différentes plateformes. Notre solution répond aux défis critiques de la gestion des dossiers médicaux en mettant en œuvre des mécanismes robustes de contrôle d'accès, de stockage sécurisé des documents et d'audit transparent.

Les objectifs principaux de ce projet sont de créer un système de gestion de dossiers médicaux inviolable qui protège les informations des patients contre les accès et modifications non autorisés. Nous avons développé un mécanisme de gestion des accès granulaire qui permet aux patients de contrôler qui peut consulter leurs dossiers médicaux, tout en concevant des applications offrant une expérience utilisateur cohérente et fluide sur les environnements web et de bureau.

Notre approche technologique représente une avancée significative dans la gestion des dossiers médicaux. Nous avons utilisé la technologie blockchain pour garantir l'immutabilité des données, la transparence et la conservation sécurisée des dossiers. Les contrats intelligents régissent l'ensemble de l'écosystème, offrant des interactions automatisées et sans confiance entre les utilisateurs.

L'implémentation comprend le système de fichiers InterPlanétaire (IPFS) pour un stockage de fichiers décentralisé et distribué, offrant une redondance des données améliorée et des capacités d'adressage de contenu. Nous avons développé des mécanismes d'authentification flexibles, incluant des connexions par nom d'utilisateur et mot de passe pour les applications de bureau, et des authentifications basées sur un compte blockchain pour les plateformes web.

Note : the Desktop APP and The Web APP are using the same Contracts (contracts addresses and ABI), and the same IPFS.

2. Architecture du Système

Infrastructure Blockchain

Notre infrastructure blockchain repose sur une architecture décentralisée et sécurisée, conçue spécifiquement pour la gestion des documents médicaux. Nous utilisons Ethereum comme plateforme de blockchain principale, exploitant sa robustesse et sa flexibilité pour le déploiement de contrats intelligents.

Les contrats intelligents, développés en Solidity, sont au cœur de notre architecture. Trois contrats principaux gouvernent l'écosystème :

1. **Contrat de Gestion des Patients** : Gère l'enregistrement des patients, la récupération des informations et la gestion des dossiers médicaux.
2. **Contrat de Gestion des Médecins** : Supervise l'enregistrement des médecins, la gestion des accès patients et les mécanismes de révocation.
3. **Contrat d'Audit** : Enregistre et trace toutes les actions significatives dans le système, garantissant une transparence et une traçabilité complètes.

Ces contrats interagissent de manière sécurisée et transparente, en utilisant des mécanismes de validation et de consensus pour assurer l'intégrité des données.

Intégration IPFS

Le système de fichiers InterPlanétaire (IPFS) est intégré comme solution de stockage décentralisé pour les documents médicaux. Contrairement au stockage traditionnel centralisé, IPFS offre plusieurs avantages significatifs :

- **Décentralisation** : Les fichiers sont distribués sur un réseau pair à pair, éliminant les points de défaillance uniques.
- **Adressage de contenu** : Chaque document est identifié par son hachage unique, garantissant l'intégrité et l'immuabilité.
- **Redondance** : Les documents sont automatiquement répliqués sur plusieurs nœuds, améliorant la disponibilité et la résilience.

Lors du téléchargement d'un document, le système génère un hachage IPFS unique qui est ensuite stocké dans le contrat blockchain. Ce processus assure que les métadonnées sont sécurisées et vérifiables sur la blockchain, tandis que le contenu réel reste stocké de manière efficace et décentralisée via IPFS.

Écosystème d'Applications

Application Web (ReactJS)

L'application web a été développée en utilisant ReactJS, offrant une interface utilisateur moderne, réactive et performante. Les caractéristiques clés incluent :

- **Architecture Composant** : Utilisation de composants React réutilisables pour une modularité et une maintenabilité optimale.
- **Authentification Blockchain** : Connexion directe via des portefeuilles Ethereum (MetaMask).
- **Interface Responsive** : Design adaptatif fonctionnant sur différents appareils et tailles d'écran.
- **Intégration Web3** : Utilisation de la bibliothèque 'ethers' pour l'interaction avec les contrats intelligents.

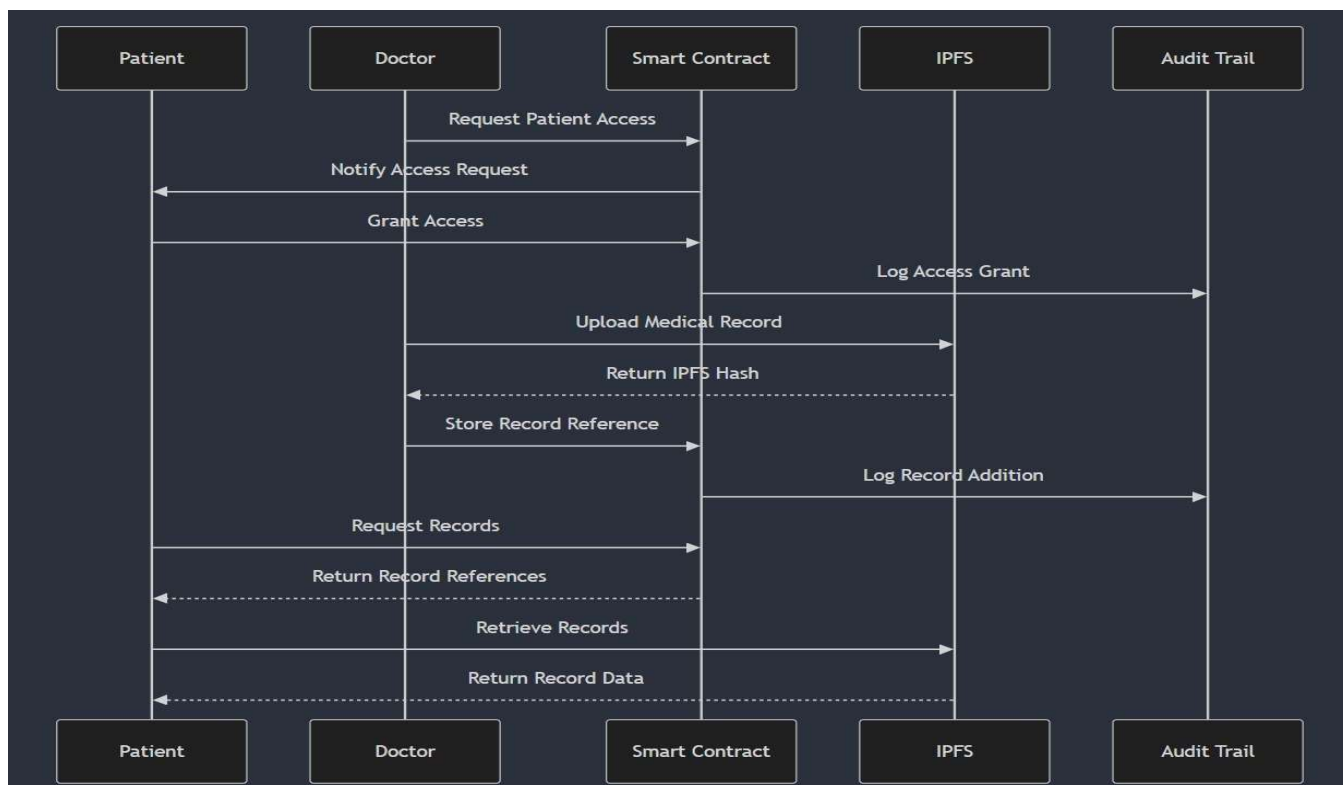
Application Desktop (Tkinter et CustomTkinter)

L'application de bureau, construite avec Tkinter et CustomTkinter, offre une alternative locale à l'application web :

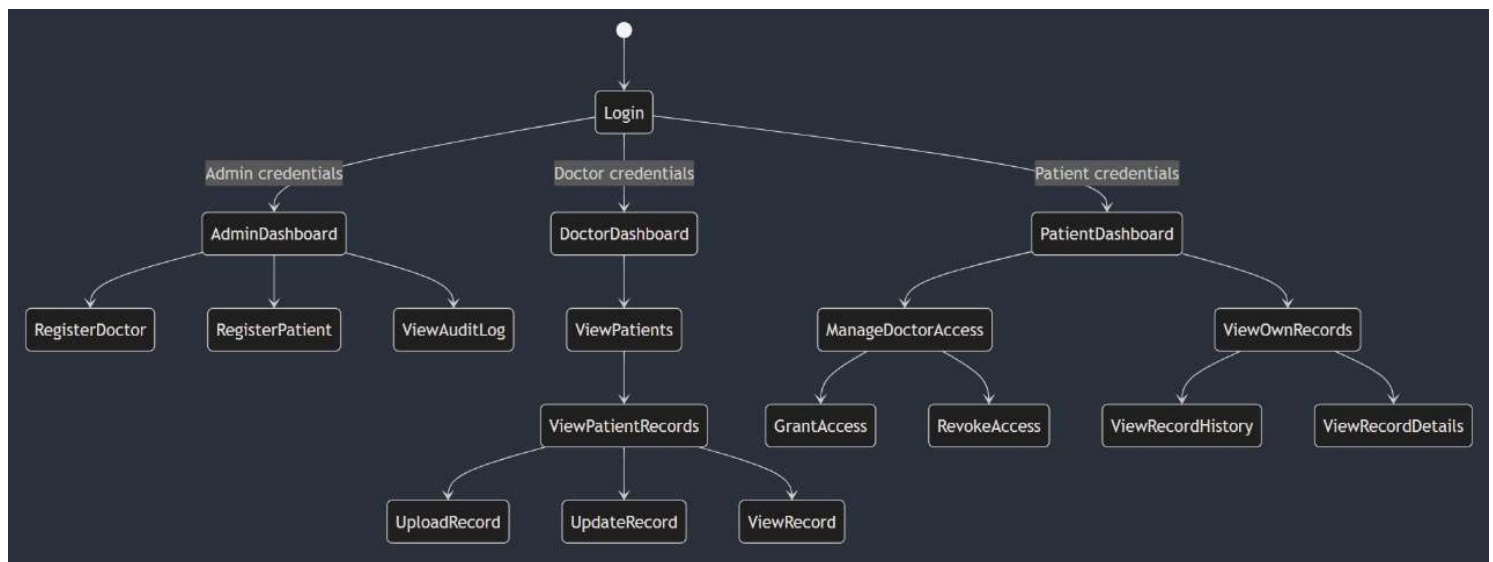
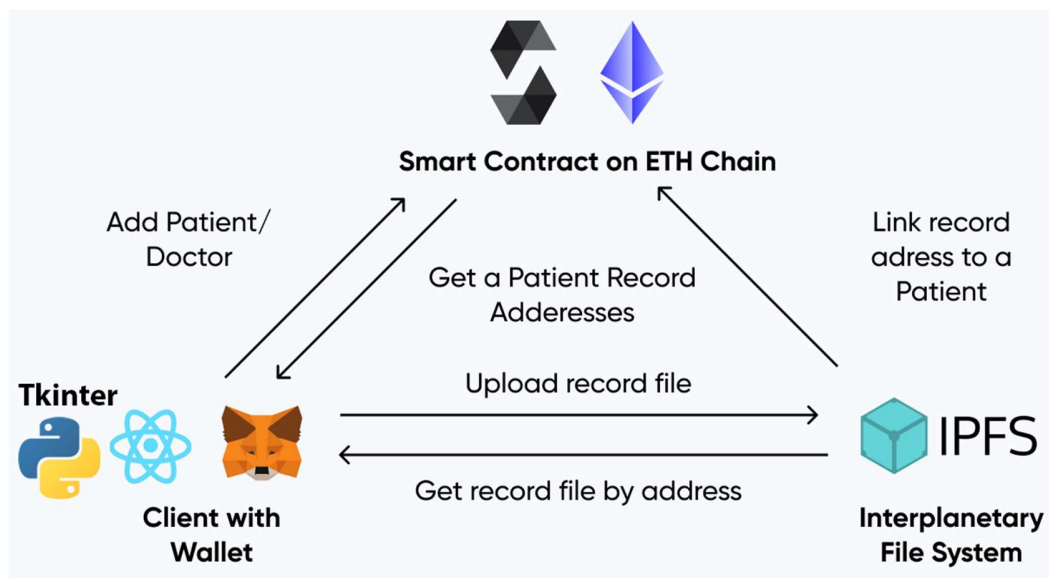
- **Interface Utilisateur Native** : Design personnalisé utilisant CustomTkinter pour une apparence moderne.
- **Authentification par Nom d'Utilisateur/Mot de Passe** : Mécanisme de connexion traditionnel adapté aux environnements desktop.
- **Intégration Blockchain** : Communication avec les contrats intelligents via des bibliothèques Python comme 'web3.py' et 'eth_account.py'.
- **Gestion Multiplateforme** : Compatibilité avec Windows, macOS et Linux.

L'architecture globale est conçue pour offrir une expérience utilisateur cohérente et sécurisée à travers différentes plateformes, en maintenant l'intégrité et la confidentialité des données médicales.

Voilà le diagramme de séquence :



Voilà L'architectures globale :



4. Conception des Contrats Intelligents

3.1 Contrat de Gestion des Patients

Le contrat de gestion des patients est conçu pour gérer l'ensemble du cycle de vie des informations et des documents des patients de manière sécurisée et transparente.

Inscription des Patients

Le processus d'inscription des patients comprend plusieurs étapes de validation :

- Vérification de l'unicité de l'identifiant patient
- Génération d'un identifiant unique sur la blockchain
- Stockage chiffré des informations de base

Structure de données d'un patient :

```
• struct Patient {
•     string username;
```

```

•     string role;
•     bool isRegistered;
•     }
•     mapping(address => Patient) public patients;

```

Récupération des Informations du Patient

Mécanismes de récupération sécurisés :

- Liste des médecins ayant un accès
- Historique des modifications d'accès
- Protection contre les accès non autorisés

Gestion des Dossiers Médicaux

Fonctionnalités avancées de gestion documentaire :

- Ajout de nouveaux documents médicaux (par le médecin)
- Horodatage précis de chaque ajout
- Restriction d'accès basée sur les permissions

Structure de données d'un Medical Record :

```

• struct MedicalRecord {
•     string ipfsHash;
•     string fileType;
•     string fileName;
•     string title;
•     string resume;
•     uint256 timestamp;
•     address doctor;
•     bool isActive;
•     string previousVersion;
• }
• mapping(address => MedicalRecord[]) public patientRecords;

```

3.2 Contrat de Gestion des Médecins

Inscription des Médecins

Processus d'inscription rigoureux :

- Validation par l'administrateur
- Vérification des identifiants professionnels
- Attribution d'un identifiant unique
- Création d'un profil blockchain sécurisé

Structure de données d'un médecin :

```

• struct Doctor {
•     string username;

```

```
•     string role;  
•     bool isRegistered;  
•     }  
•     mapping(address => Doctor) public doctors;
```

Contrôle d'Accès Patient

Système de gestion des accès sophistiqué :

- Liste dynamique des patients autorisés
- Mécanisme de demande d'accès
- Validation par le patient

```
mapping(address => mapping(address => bool)) public doctorPatientAccess;
```

Mécanismes d'Octroi et de Révocation d'Accès

Processus de gestion des permissions :

- Demande d'accès initiée par le médecin
- Révocation possible à tout moment par le patient
- Historique complet des modifications d'accès

3.3 Contrat d'Audit

Suivi des Actions

Mécanisme de traçabilité comprehensive :

- Enregistrement de chaque action significative
- Métadonnées détaillées
- Impossibilité de modifier les logs historiques

Types d'Événements Supportés

Événements tracés :

- DOCTOR_REGISTERED : Inscription d'un nouveau médecin
- PATIENT_REGISTERED : Inscription d'un nouveau patient
- RECORD_ADDED : Ajout d'un nouveau document médical
- ACCESS_GRANTED : Octroi d'un accès à un médecin
- ACCESS_REVOKED : Révocation d'un accès

Transparence Administrative

Fonctionnalités de supervision :

- Accès administrateur aux logs complets

- Export des données d'audit
- Protection contre les falsifications

La structure d'événement d'audit :

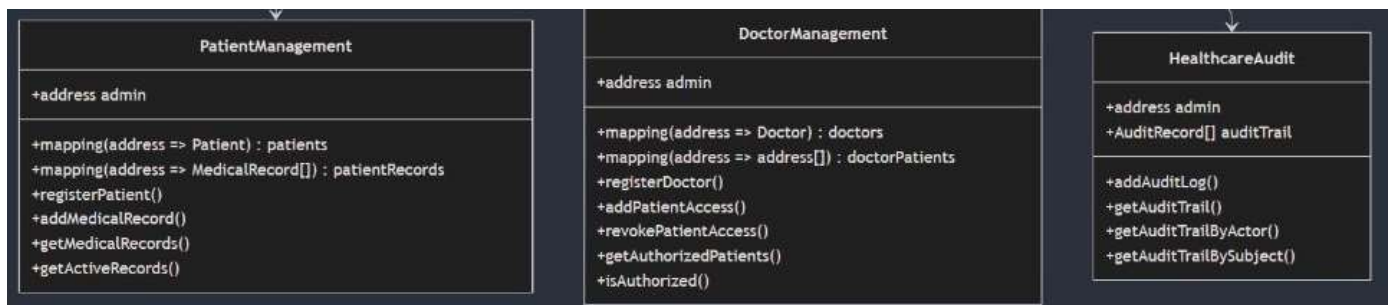
```

• struct AuditRecord {
•     address actor;           // Who performed the action
•     string actionType;       // Type of action (e.g., "RECORD_ADDED",
    "ACCESS_GRANTED")
•     address subject;         // Who the action was performed on
•     string details;
•     uint256 timestamp;
• }
• AuditRecord[] public auditTrail;

```

Cette conception de contrats intelligents offre une approche complète, sécurisée et transparente de la gestion des documents médicaux, en plaçant le contrôle et la confidentialité au cœur du système.

Voilà l'architecture des contrats :

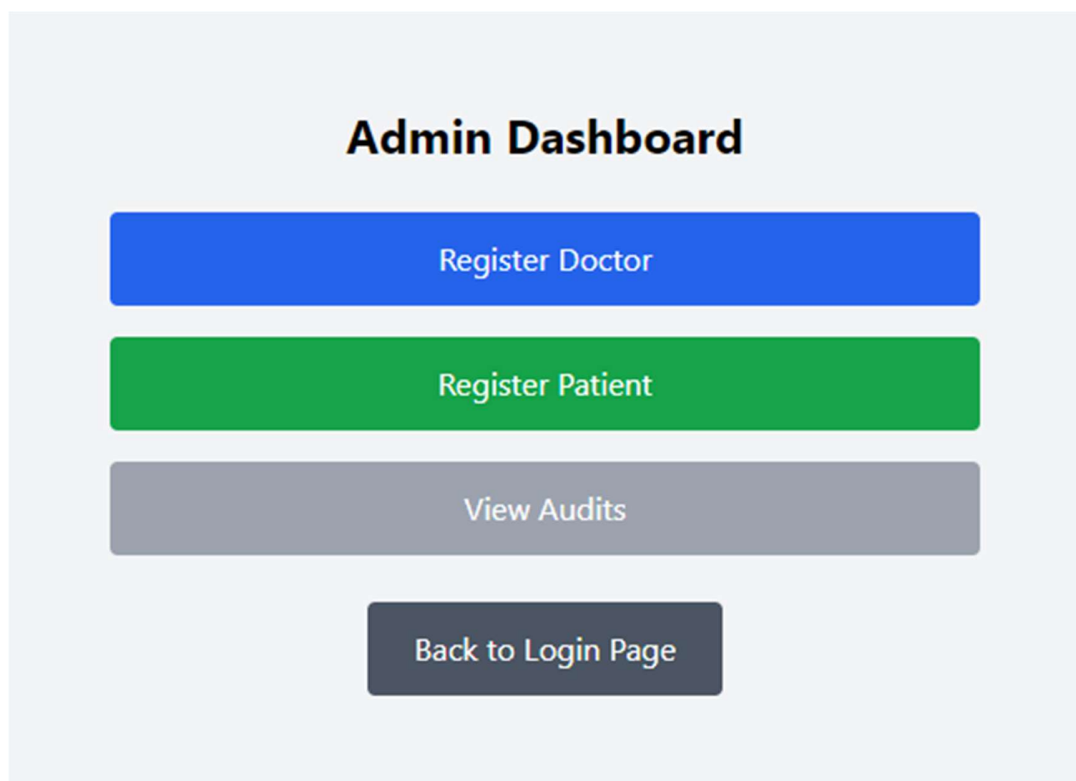
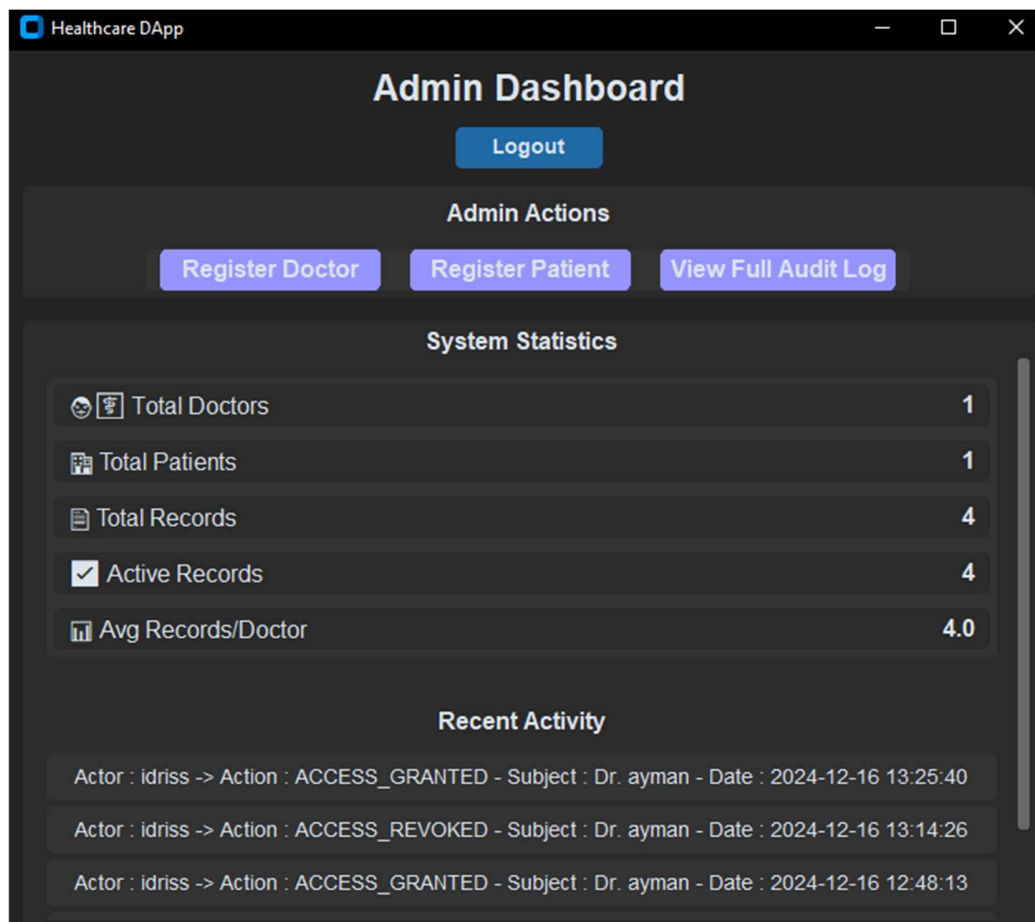


4. Rôles et Permissions Utilisateurs

4.1 Administrateur (Admin)

L'administrateur est le Déploiera initial des contrats intelligents, il peut :

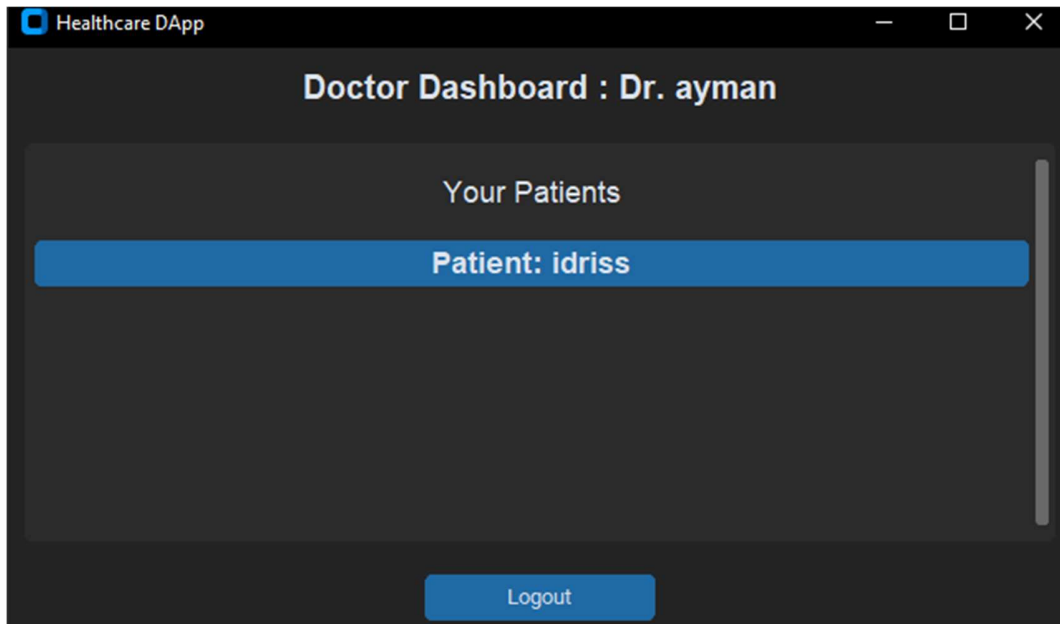
- Accès aux logs complets d'événements
- Enregistrement des Utilisateurs (docteur ou patient)
- Exportation des données d'audit
- Voir des Statistiques : le nombre les docteurs, patients et les document enregistrer.



4.2 Médecin

Le médecin avoir un Dashboard pour la gestion des patients et des documents médicaux :

- Liste des patients avec accès autorisé
- Ajout de nouveaux documents médicaux
- Modifier des documents existants



[Back to Home](#)

Doctor Dashboard: ayman

Doctor Wallet : 0x70997970C51812dc3A010C7d01b50e0d17dc79C8

Your Patients

Patient Name : idriss

Previous Records

file title: test 1 | file name: 1484.jpg | 12/16/2024, 12:24:52 PM

[View File](#)

file title: title | file name: projet.txt | 12/16/2024, 12:49:34 PM

[View File](#)

Description: description

file title: test 2 | file name: output_image.png | 12/16/2024, 1:26:36 PM

[View File](#)

Description: description exist

file title: test 3 | file name: a002.jpg | 12/16/2024, 1:27:31 PM

[View File](#)

Add Record for idriss

Record Title

[Choose File](#)

No file chosen

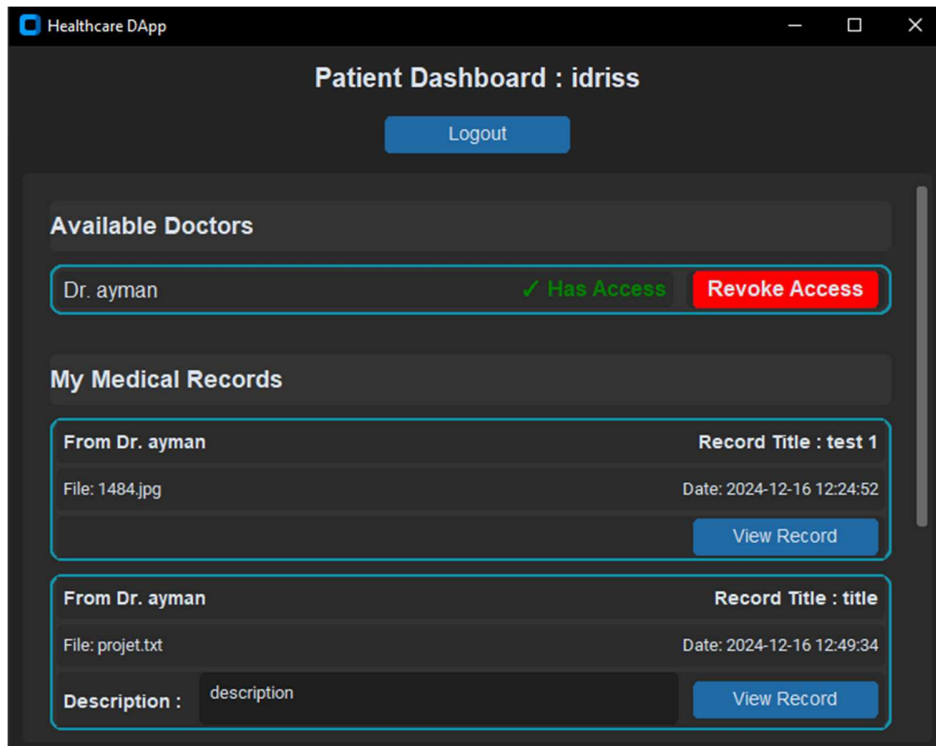
Description

[Add Record](#)

4.3 Patient

La patiente avoir un Dashboard pour visualiser les médecins enregistrer pour la gestion granulaire des permissions, et aussi voir son document médical

- Accord/révocation d'accès aux médecins
- Liste de tous les documents médicaux
- Ouverture dans l'application native
- Aucune possibilité de modification



[Back to Home](#)

Patient Dashboard:

Patient Information

Name: idriss

Wallet Address: 0x3C44CdDd86a900fa2b585dd299e03d12FA4293BC

Available Doctors

ayman	Revoke Access
-------	-------------------------------

Medical Records

Title : test 1 From Dr.ayman	1484.jpg 12/16/2024, 12:24:52 PM	View File
Title : title From Dr.ayman	projet.txt 12/16/2024, 12:49:34 PM	View File
Description: description		
Title : test 2 From Dr.ayman	output_image.png 12/16/2024, 1:26:36 PM	View File
Description: description exist		

Cette architecture de rôles et permissions assure un équilibre optimal entre sécurité, confidentialité et facilité d'utilisation, en plaçant le patient au centre de la gestion de ses données médicales.

5. Mécanismes d'Authentification

Authentification Application Desktop

- Authentification traditionnelle par nom d'utilisateur et mot de passe

- Stockage dans un fichier JSON, chaque utilisateur avoir leur username, mot de passe (chiffrer par une fonction de hachage), clé privée de leur portefeuille pour faire des transactions.

Authentification Application Web

- Authentification basée sur les comptes blockchain
- Authentification par nom d'utilisateur et clé de leur portefeuille
- Connexion via portefeuille Ethereum (MetaMask)
- Signature de messages cryptographiques
- Validation de l'identité via adresse blockchain
- Aucun stockage de mot de passe traditionnel





6. Détails Techniques de l'Implémentation

Technologie Blockchain

Notre solution repose sur la plateforme Ethereum, choisie pour sa robustesse et sa flexibilité dans le déploiement de contrats intelligents. Nous avons développé ces contrats en utilisant Solidity, le langage de programmation spécifique aux applications blockchain. Pour le développement et les tests, nous avons opté pour le réseau de test Hardhat, qui offre un environnement de développement complet et sécurisé. Le consensus Proof-of-Stake permet des transactions rapides et écoénergétiques, alignant notre solution avec les standards modernes de technologie blockchain.

Intégration IPFS

L'intégration d'IPFS dans notre système se fait via un nœud local, assurant un stockage décentralisé et hautement sécurisé des documents médicaux. Chaque document reçoit un hachage unique, garantissant son intégrité et sa traçabilité. Pour assurer un isolement optimal et renforcer la sécurité, nous avons implémenté le système IPFS dans un conteneur Docker, permettant une isolation complète de l'environnement de stockage.

Name	Image	Status	CPU (%)	Port(s)
 ipfs e99f85caed50	ipfs/kubo:latest	Running	1.22%	4001:4001  5001:5001  8080:8080  Show less

Technologies Frontend

Le frontend de notre application combine deux technologies distinctes pour offrir une expérience utilisateur cohérente. La version web est développée avec React.js, offrant une interface dynamique et réactive, tandis que l'application desktop utilise Tkinter et CustomTkinter pour une intégration native. Les bibliothèques Web3 jouent un rôle crucial : ethers pour le web et web3.py avec eth_account.py pour le desktop, permettant une interaction

transparente avec la blockchain. L'ensemble du design est conçu de manière responsive, s'adaptant parfaitement aux différents appareils et tailles d'écran.

Développement des Contrats Intelligents

Le développement des contrats intelligents a été réalisé en Solidity, le langage de référence pour les applications blockchain Ethereum. Nous avons utilisé Remix IDE pour le prototypage initial et Hardhat comme environnement de développement principal. Une attention particulière a été portée à la qualité du code avec des tests unitaires complets, une vérification formelle des contrats et un audit de sécurité rigoureux. Le processus de déploiement et de migration des contrats a été conçu pour garantir une transition fluide et sécurisée entre les différents environnements de développement et de production.

7. Conclusion :

Notre projet représente une avancée significative dans la gestion numérique des documents médicaux. En intégrant blockchain, IPFS et des applications multiplateformes, nous avons créé une solution innovante qui résout les principaux défis de sécurité, confidentialité et accessibilité des données médicales.

Les réalisations clés incluent un système décentralisé garantissant l'intégrité des données, un contrôle total pour le patient sur ses informations médicales, et des applications web et desktop offrant une expérience utilisateur transparente et sécurisée.

Les perspectives futures sont prometteuses : extension à différents types de documents médicaux, intégration avec des systèmes existants, et amélioration des mécanismes de partage de données. Ce projet démontre le potentiel des technologies émergentes pour transformer la gestion des informations médicales.

Plus qu'une simple solution technique, notre système pose les bases d'un écosystème numérique de santé plus sûr, transparent et centré sur le patient.

8. Référence :

- <https://github.com/TomSchimansky/CustomTkinter>

- D. B. Khadse, Uddesh Nikam, Nisha Mate : DAPP to Store Electronic Medical Health Records on Ethereum Blockchain and IPFS

- Agbo, C. C., Mahmoud, Q. H., & Eklund, J. M. (2019). Blockchain Technology in Healthcare: A Systematic Review. Healthcare, 7(2), 56.

- <https://legacy.reactjs.org/docs/getting-started.html>

-