L'ECOLE DES EXPERTS METIERS
DE L'INFORMATIQUE

5BLOC

Virtual Real Estate Tokenization Smart Contract

Présenté par :

Mohamed Daadaa | Ibrar Hamouda | Rodney Maglo

20 Février 2025

SOMMAIRE

INTRODUCTION

Virtual Real Estate Tokenization Smart Contract

- Une dapp développée en Rust pour la blockchain Solana, utilisant le framework Anchor. L'objectif principal est de tokeniser des biens immobiliers en les représentant sous forme de tokens.
- Cette dapp vise à simplifier la gestion de l'immobier, son transfert et sa vérification, dans un environnement sécurisé et décentralisé.
- Une documentation claire sur les régles métiers ainsi que sur l'aspect technique est fournie (fichier README).

Technologies:

- Solana Blockchain
- Rust
- Anchor Framework
- IPFS

SCÉNARIO

Cas d'usage

- Le scénario envisagé fait partie du gestion décentralisée d'actifs numériques ou la dapp gére des biens immobiliers. Chaque propriété est associée à des métadonnées (nom, type, valeur et IPFS hash) qui permettent d’authentifier le bien.

Actions clés

- Un utilisateur peut **créer** un compte, émettre (mint) un token de propriété, puis **échanger** ce token avec d’autres utilisateurs, tout en respectant des contraintes temporelles. Le type de chaque bien est validé grâce à un hash spécifique, comparé à la valeur enregistrée sur IPFS. Ainsi, chaque type de bien possède un hash unique sur IPFS, ce qui permet une vérification décentralisée de son authenticité et de sa valeur.

Fonctionnalités principales

Les Smart contrats

initialize_user

permet la création d’un compte utilisateur initialisé pour la gestion de ses propriétés.

mint_property

autorise la création d’un token de propriété, en vérifiant que l’IPFS hash correspond au type de propriété attendu (résidentiel, commercial, ou luxirious).

exchange_property

Gère le transfert sécurisé des tokens de propriété entre les utilisateurs, en mettant à jour à la fois l'historique de la propriété et les listes de propriétés des utilisateurs.

verify_property_metadata

fournit une vérification additionnelle des métadonnées, garantissant que les informations du bien sont correctes et cohérentes avec l’IPFS hash indiqué.

Get Property_Datas

Fournit les données liées à un bien.

display_property_owners

Fournit la liste des propriétaires d’un bien.

Contraintes temporelles

Un utilisateur ne peut pas posséder plus de 4 propriétés. Arrêter le transfert si le destinataire est en période de refroidissement. Après chaque transfert, un refroidissement est appliqué.

Stockage ipfs

L'idée est d'attribuer un hash (clé) spécifique à notre application pour chaque type de bien stocké sur IPFS, ce qui permet de vérifier séparément le type de propriété.

RESULTATS OBTENUS

- Les solutions mises en place permettent de créer et de transférer des tokens de propriété de manière sécurisée et transparente. Le système assure une vérification initiale des métadonnées et vérifie le bon déroulement des fonctionnalités en respectant les réfles métiers.
- Des tests ont été faits pour simuler des cas réels et vérifier ces fonctionnalités. Un système pour afficher (logging) le déroulement des tests étape par étape sur le terminal ainsi qu'une bonne gestion d'erreur par des messages spécifiques.

Les tests scénario

- crée un nouveau token de propriété.
- empêche un utilisateur de posséder plus de 4 propriétés.
- permet plusieurs transactions sans délai quand le cooldown est écoulé.
- permet des échanges de propriété immédiats.
- applique le cooldown normal de 5 minutes.
- applique la pénalité de 10 minutes en cas de non-respect du cooldown.
- applique le cooldown de 5 minutes entre les transferts de propriétés.
- vérifie la validité des métadonnées de propriété.
- récupère les données d'une propriété.

PROBLÈMES RENCONTRÉS

SETUP

- Installation des version particulières pour Rust, Cargo et Anchor.
- Les dérnieres installations de cargo, rust et anchor ne garantissent pas le bon fonctionnement de l'application dès son initialisation.

TOKENISATION

- Difficulté de trouver les bonnes pratiques du framework anchor avec rust pour faire des contrats smart avec un clean code.

STOCKER SUR L'IPFS

- Sans utiliser un outil externe tel que Pinata, le stockage de métadonnées sur l'IPFS était un défi compliqué.

TESTING

- Simulion des données pour génerer les données de l'application.
- Setup de l'environnement des tests.

MERCI DE VOTRE ATTENTION