RAPPORT DE PROJET

THEME : CREER UNE APPLICATION CLIENT/SERVEUR QUI INTERAGIT AVEC UN SMART CONTRACT DEPLOYE SUR LA BLOCKCHAIN.

CAS D'ULISATION : SYSTEME DE VOTE DECENTRALISE

GROUPE 1

MEMBRES DU PROJET:

LEYENOHIN MICHEE CEPHAS

MOUHI CHRIST EMMANUEL

SOMMAIRE

1. INTRODUCTION

Avec l'avènement des technologies blockchain, de nouvelles opportunités se présentent pour améliorer la transparence, la sécurité et l'efficacité des processus de vote. Ce projet vise à créer un système de vote décentralisé qui exploite les avantages offerts par la blockchain, garantissant ainsi des élections plus sécurisées et transparentes.

Le projet utilise une combinaison de technologies modernes pour construire une application robuste. Django est choisi pour le backend en raison de sa flexibilité et de ses puissantes capacités de gestion des bases de données. Next.js, un framework React, est utilisé pour le frontend pour offrir une expérience utilisateur fluide et réactive. Le smart contract, développé en Solidity, est déployé sur la blockchain Ethereum pour assurer l'intégrité des votes. Web3.py et Web3.js sont utilisés respectivement dans le backend et le frontend pour interagir avec le smart contract. La base de données PostgreSQL est utilisée pour stocker des informations hors chaîne.

Ce rapport décrit en détail les différentes étapes du projet, depuis la définition des exigences fonctionnelles jusqu'au déploiement et à la maintenance. Chaque section fournit une vue d'ensemble des décisions prises, des technologies utilisées et des processus mis en place pour atteindre les objectifs du projet.

2. DEFINITION DES EXIGENCES FONCTIONNELLES

Pour interagir efficacement avec le smart contract, plusieurs fonctionnalités clés ont été définies :

1. **Lecture des Données du Smart Contract :**
    - Capacité à lire l'état actuel des votes et des candidats depuis la blockchain.
2. **Écriture de Données dans le Smart Contract** :
    - Enregistrement des votes des utilisateurs sur la blockchain de manière immuable.
3. **Gestion des Événements émis par le Smart Contract :**
    - Surveillance des événements tels que l'enregistrement des votes pour mettre à jour l'interface utilisateur en temps réel.
4. **Authentification des Utilisateurs** :
    - Sécuriser l'accès pour garantir que seuls les utilisateurs autorisés peuvent voter

3. CHOIX DE LA BLOCKCHAIN ET DU SMART CONTRACT

Blockchain Sélectionnée : Ethereum

Smart Contract : Développé en Solidity, le smart contract gère l'enregistrement des candidats et des votes, en assurant une transparence et une sécurité totale.

4. CONCEPTION DE L' ARCHITECTURE CLIENT/ SERVEUR

Technologies choisies :

- **Backend** : Django avec Django Rest Framework
- **Frontend** : Next.js

- **Blockchain Interaction** : Web3.py pour le backend et Web3.js pour le frontend

Architecture :

- Le frontend interagit avec le backend pour les opérations d'authentification et pour récupérer les informations à afficher.
-
- Le backend agit comme une passerelle vers la blockchain, gérant les interactions avec le smart contract via Web3.py.

5. MISE EN PLACE DU SERVEUR (BACKEND)

1. **Configuration de Django** :
   - Initialisation du projet Django et configuration des dépendances.
   - Utilisation de Django REST Framework pour créer des endpoints RESTful.
2. **Endpoints RESTful** :
   - Création d'API pour la gestion des utilisateurs, des candidats et des votes.
   - Intégration de Swagger pour documenter automatiquement les API.
3. **Intégration de Web3.py** :
   - Configuration pour interagir avec la blockchain Ethereum.
   - Implémentation des appels aux méthodes du smart contract.

6. DEVELOPPEMENT DU CLIENT (FRONTEND)

- **Configuration de Next.js** :
  - Initialisation du projet Next.js et installation des dépendances nécessaires.
- **Interfaces Utilisateur** :
  - Création des pages et composants React pour permettre aux utilisateurs d'interagir avec le système de vote.
  - Intégration de Material-UI pour une interface réactive et esthétique.
- **Intégration de Web3.js** :

- Utilisation de Web3.js pour permettre au frontend d'interagir directement avec le smart contract.

7. INTEGRATION DU SMART CONTRACT

- **Appels de Méthodes** :
  - Utilisation des fonctions fournies par Web3.py pour appeler les méthodes du smart contract depuis le backend.
  - Gestion des transactions pour enregistrer les votes sur la blockchain.
- **Gestion des Autorisations et Authentification** :
  - Implémentation de JWT pour sécuriser les interactions utilisateur.
  - Mécanismes de vérification pour s'assurer que seuls les utilisateurs autorisés peuvent voter.

8.TESTS

- **Tests Unitaires** :
  - Pour chaque composant du frontend et du backend, afin de vérifier leur bon fonctionnement isolé.
- **Tests d'Intégration** :
  - Pour vérifier la communication fluide entre le client, le serveur et le smart contract.

9. DEPLOIEMENT

- **Serveur Backend** :

Le backend n’est pas encore deployé , il est en local pour le moment mais sera deployé sur Heroku

- **Client Frontend** :
  - Déployé sur un service comme Vercel pour une accessibilité optimale.
- **Smart Contract** :
  - Pas encore deployé sera déployé sur le réseau Ethereum, après des tests exhaustifs sur un environnement local avec Ganache.

10. MAINTENANCE ET AMELIORATIONS

1. **Surveillance des Performances et de la Sécurité** :
   - Surveillance continue pour garantir une performance optimale et identifier les éventuelles failles de sécurité.
2. **Correction des Bugs** :
   - Résolution rapide des problèmes signalés par les utilisateurs.
3. **Ajout de Nouvelles Fonctionnalités** :
   - Évolution continue du système en fonction des besoins des utilisateurs et des améliorations possibles du smart contract.

Conclusion

Ce projet démontre comment la blockchain peut être utilisée pour créer un système de vote décentralisé sécurisé et transparent. En combinant Django, Next.js, Web3.py, et Solidity, nous avons construit une application robuste qui répond aux besoins modernes des élections. La transparence et la sécurité offertes par la blockchain, associées à une interface utilisateur intuitive, font de ce système un modèle pour les futures applications de vote.