Kingdom of Morocco
Abdelmalek Essaâdi University
Faculty of Sciences and Techniques - Tangier
Department of Computer Engineering
MST : Artificial Intelligence and Data Science

Atelier 6
Mini Social Media

Realiser par : DAGHMOUMI Marouan
Encadrer par : Pr.Ikram Ben abdel ouahab
10 November 2024

Table des matières

1 Introduction

1.1 Context

Dans le cadre de notre formation en Master Sciences et Techniques : Intelligence Artificielle et Science des Données, nous avons développé une application décentralisée (dApp) de média social utilisant la blockchain Ethereum. Ce projet démontre l'utilisation pratique de la technologie blockchain pour créer une plateforme sociale décentralisée.

1.2 Objectifs

Les objectifs principaux de ce projet sont :
— Création d'une application décentralisée interactive
— Implémentation des fonctionnalités de base d'un réseau social
— Utilisation de la blockchain pour le stockage et la gestion des données
— Integration avec MetaMask pour l'authentification des utilisateurs

2 Architecture du Projet

2.1 Diagramme des Fonctionnalités

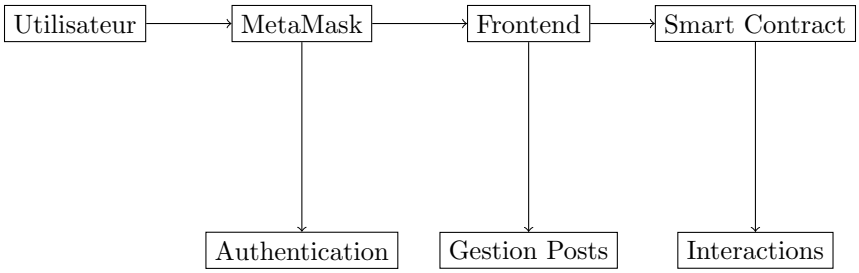

FIGURE 1 – Architecture générale de l'application

3 Configuration de l'Environnement

3.1 Installation de MetaMask

1. Télécharger l'extension MetaMask depuis le Chrome Web Store
2. Créer un nouveau wallet ou importer un existant
3. Configurer le réseau local Hardhat :
   — Network Name : Hardhat Local
   — RPC URL : http ://127.0.0.1 :8545/
   — Chain ID : 31337
   — Currency Symbol : ETH

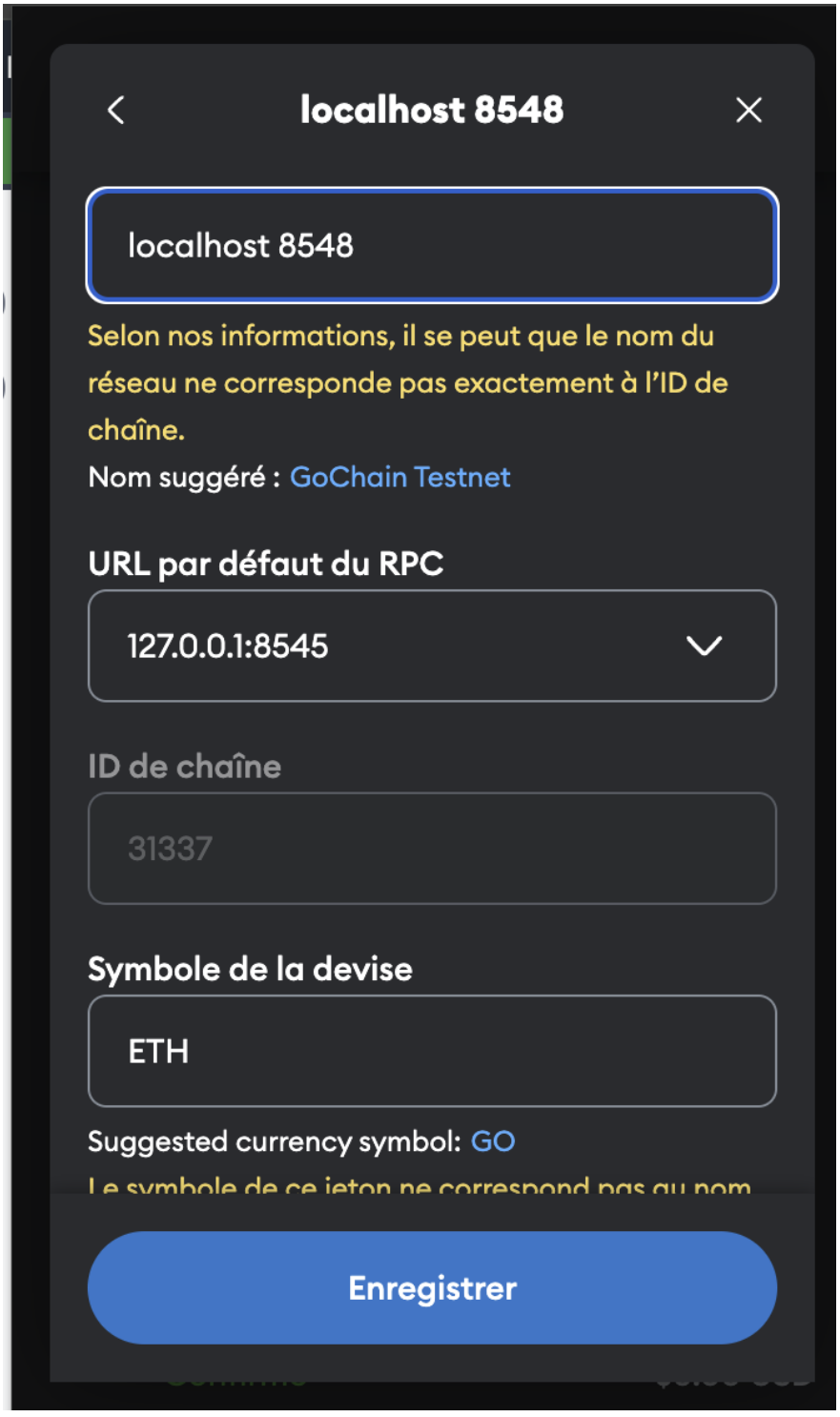

FIGURE 2 – Configuration de MetaMask

3.2 Installation de Hardhat

```
# Initialisation du projet
npm init -y

```

```
# Installation de Hardhat
npm install --save-dev hardhat

# Initialisation de Hardhat
npx hardhat

# Installation des d pendances
npm install --save-dev @nomiclabs/hardhat-ethers ethers
```

4 Fonctionnalités de l'Application

4.1 Page de Connexion

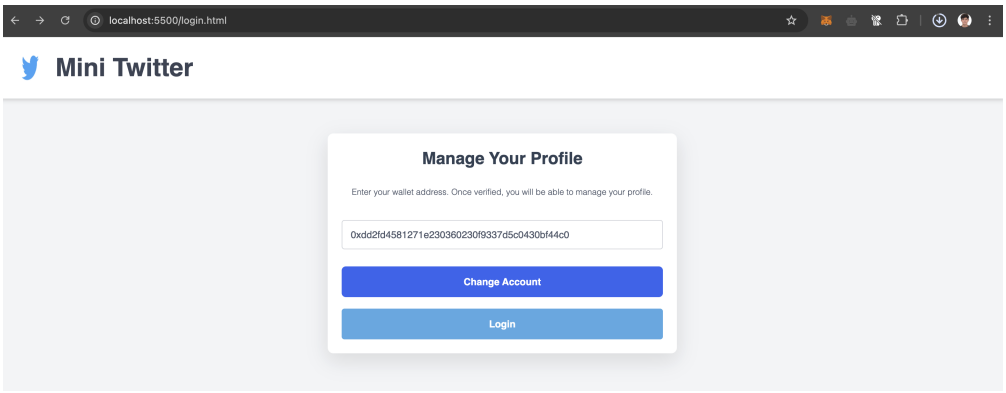

FIGURE 3 – Interface de connexion

4.2 Création de Posts

```
async function createPost() {
    try {
        const content = document.getElementById('postContent').value;
        const tx = await contract.createPost(content);
        await tx.wait();
        alert('Post  cr     avec succ s!');
    } catch (error) {
        console.error('Erreur:', error);
        alert('Erreur lors de la cr ation du post');
    }
}
```

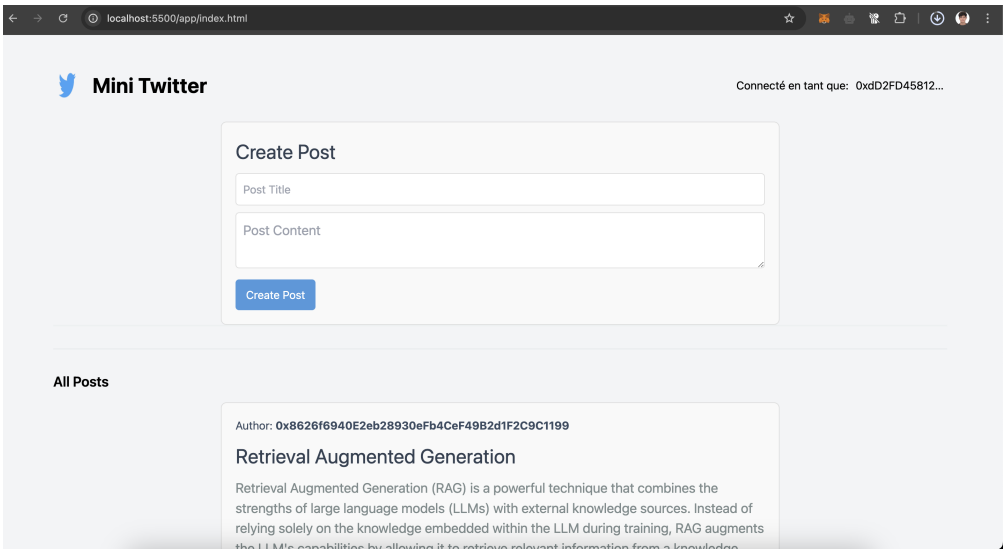

FIGURE 4 – Interface de création de post

4.3 Suppression de Posts

```
async function deletePost(postId) {
    try {
        const tx = await contract.deletePost(postId);
        await tx.wait();
        alert('Post supprim  avec succ s!');
    } catch (error) {
        console.error('Erreur:', error);
        alert('Erreur lors de la suppression');
    }
}
```

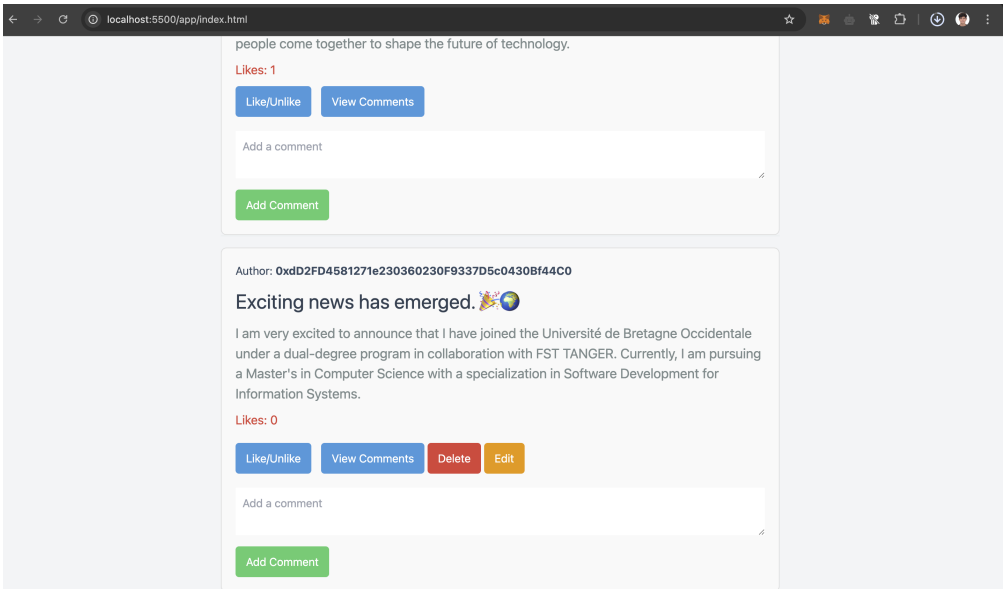

FIGURE 5 – Suppression d'un post

4.4 Commentaires

```
async function addComment(postId, content) {
    try {
        const tx = await contract.addComment(postId, content);
        await tx.wait();
        alert('Commentaire ajout  avec succ s!');
    } catch (error) {
        console.error('Erreur:', error);
        alert('Erreur lors de l'ajout du commentaire');
    }
}
```

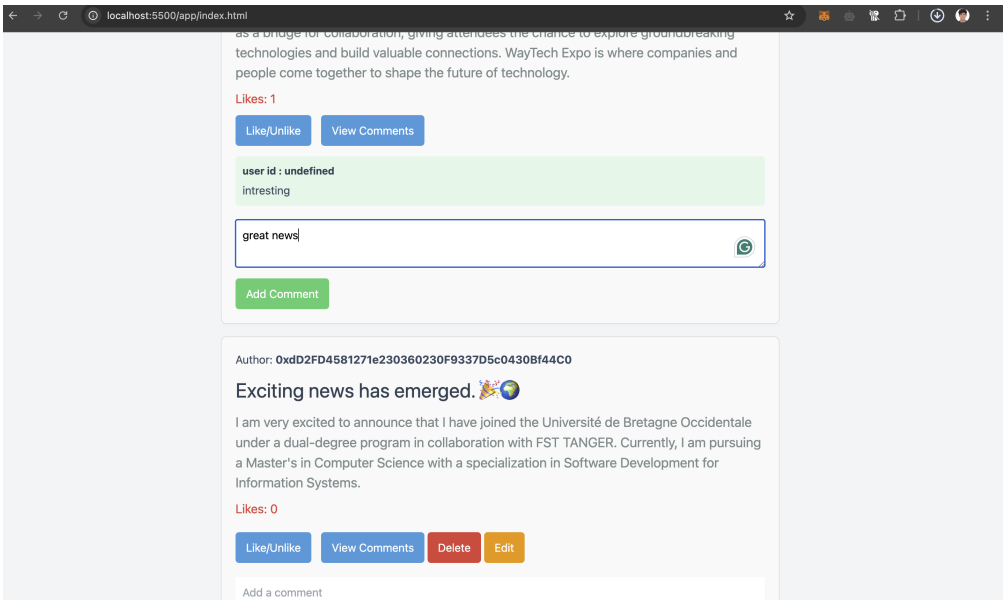

FIGURE 6 – Interface des commentaires

5 Smart Contract

5.1 Structure du Contrat

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract MiniSocial {
    struct Post {
        uint256 postId;
        string content;
        address author;
        uint256 timestamp;
        uint256 likesCount;
        bool isDeleted;
    }

    struct Comment {
        string content;
        address author;
```

```
        uint256 timestamp;
    }

    mapping(uint256 => Post) public posts;
    mapping(uint256 => Comment[]) public comments;
    uint256 public postCount;
}
```

6 Tests et Déploiement

6.1 Tests Unitaires

```
const { expect } = require("chai");

describe("MiniSocial", function () {
    let MiniSocial;
    let miniSocial;
    let owner;
    let addr1;

    beforeEach(async function () {
        MiniSocial = await ethers.getContractFactory("MiniSocial");
        [owner, addr1] = await ethers.getSigners();
        miniSocial = await MiniSocial.deploy();
        await miniSocial.deployed();
    });

    it("Should create a new post", async function () {
        await miniSocial.createPost("Test post");
        const post = await miniSocial.posts(0);
        expect(post.content).to.equal("Test post");
    });
});
```

7 Conclusion

Ce projet démontre la mise en œuvre réussie d'une application décentralisée de média social. Les principales réalisations incluent :

- Intégration réussie avec MetaMask pour l'authentification
- Implémentation des fonctionnalités CRUD pour les posts
- Système de commentaires fonctionnel
- Interface utilisateur intuitive

8 Perspectives

Les améliorations futures pourraient inclure :

- Ajout de fonctionnalités de partage
- Implémentation d'un système de hashtags
- Amélioration de l'interface utilisateur
- Optimisation des coûts de gas