جامعة عبد المالك السعدي
Université Abdelmalek Essaâdi

كلية العلوم و التقنيات بطنجة
Faculté des Sciences et Techniques de Tanger

11/10/2024

Mini Twitter DAPP

BLOCKCHAIN & SECURITE APPLICATIVE Module

Idriss Khattabi

MST: ARTIFITIAL INTELLIGENCE AND DATA SCIENCE

Introduction :

The Mini Twitter DApp is a decentralized application that mimics the core functionalities of a social media platform, allowing users to publish, edit, like, and dislike posts. The main objective of this project is to showcase the potential of blockchain technology in creating a transparent, user-owned, and tamper-proof social media platform. Unlike traditional social media, where data is controlled by a centralized entity, this DApp leverages the Ethereum blockchain to store posts and reactions, providing users with greater control and ownership over their content.

Used Technologies :

- **Ethereum Blockchain**: The decentralized network where posts and interactions are stored, ensuring security and immutability.
- **Solidity**: The programming language used to create smart contracts that manage data storage and user interactions.
- **Web3.js**: A JavaScript library that facilitates interactions between the DApp and the Ethereum blockchain, including account management and data retrieval.
- **MetaMask**: A browser-based wallet used for authenticating users and managing Ethereum transactions directly from the browser.

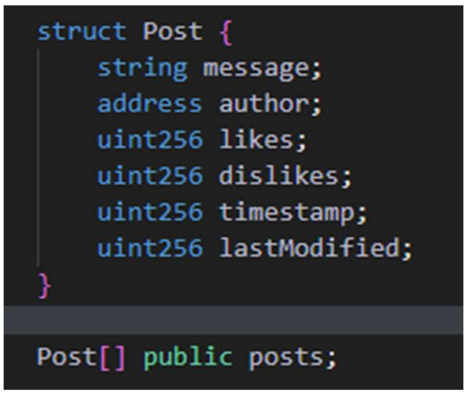
Task Details and Implementation :

➔ Smart Contract :

The smart contract, written in Solidity, is the core of the DApp, managing all data and user interactions. Key functionalities implemented in the smart contract include:

- Post Creation: Users can publish a new post, which is then stored immutably on the blockchain. Each post is associated with the user's Ethereum address.
- Post Editing: Users can edit only the posts they have authored. This is enforced through address verification, ensuring user control over their own content.
- Reaction System: Users can like or dislike posts, and these reactions are tracked on the blockchain to maintain integrity and transparency of interactions.

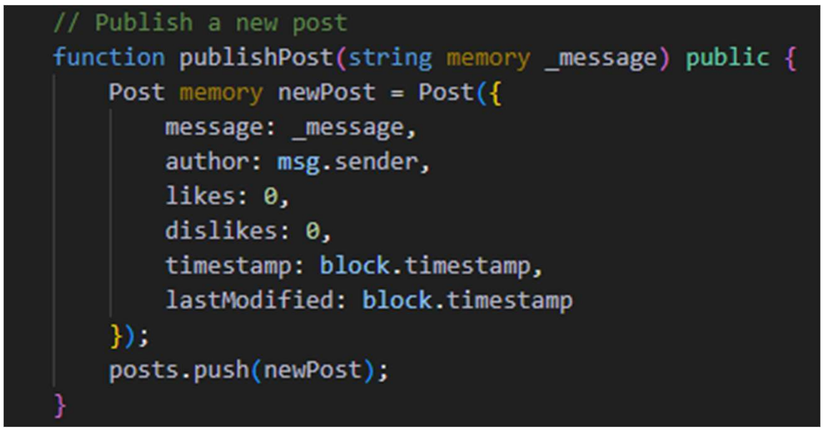
1. Contract Declaration and Struct Definition :



```
struct Post {
    string message;
    address author;
    uint256 likes;
    uint256 dislikes;
    uint256 timestamp;
    uint256 lastModified;
}

Post[] public posts;
```

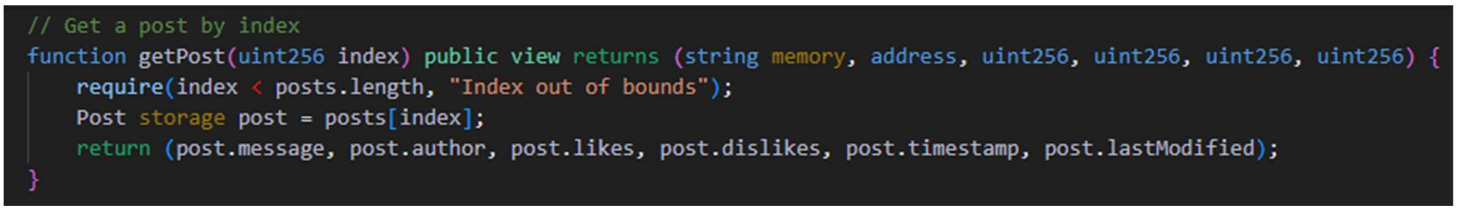
- **Post**: Defines the structure of a post, which includes:
  - message: The content of the post.
  - author: The address of the user who created the post.
  - likes and dislikes: Counters for tracking likes and dislikes.
  - timestamp: Records the time when the post was created.
  - lastModified: Updates whenever the post is edited.
- **Posts**: Declares a dynamic array called posts to store all posts created on the platform.

2. 'publishPost' Function : its Allows users to publish a new post.



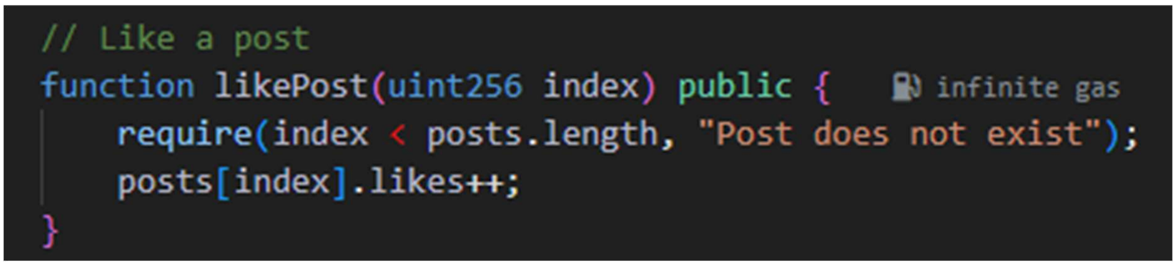
```
// Publish a new post
function publishPost(string memory _message) public {
    Post memory newPost = Post({
        message: _message,
        author: msg.sender,
        likes: 0,
        dislikes: 0,
        timestamp: block.timestamp,
        lastModified: block.timestamp
    });
    posts.push(newPost);
}
```

3. 'getPost' Function : its Retrieves the details of a specific post by its index



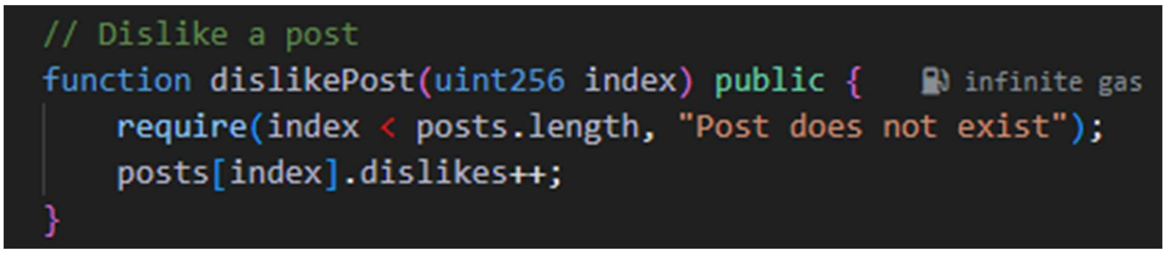
```
// Get a post by index
function getPost(uint256 index) public view returns (string memory, address, uint256, uint256, uint256, uint256) {
    require(index < posts.length, "Index out of bounds");
    Post storage post = posts[index];
    return (post.message, post.author, post.likes, post.dislikes, post.timestamp, post.lastModified);
}
```

4. ‘likePost’ Function : its Allows users to like a specific post.



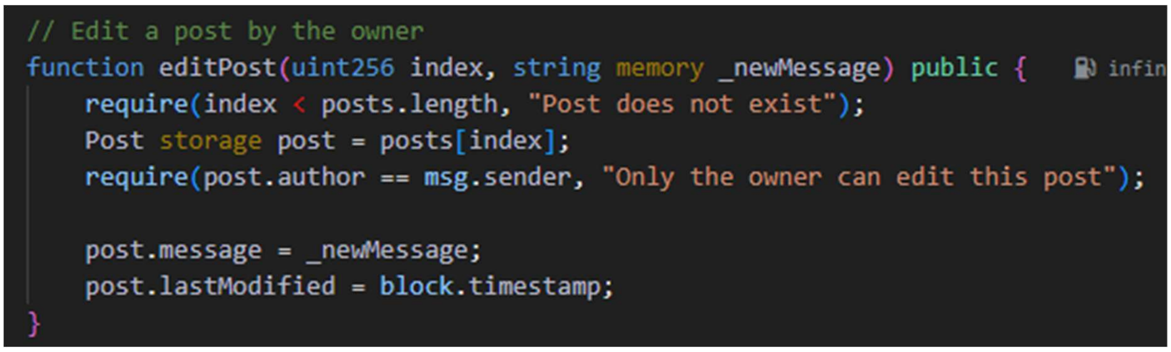
```
// Like a post
function likePost(uint256 index) public {    infinite gas
    require(index < posts.length, "Post does not exist");
    posts[index].likes++;
}
```

5. ‘dislikePost’ Function : its Allows users to dislike a specific post.



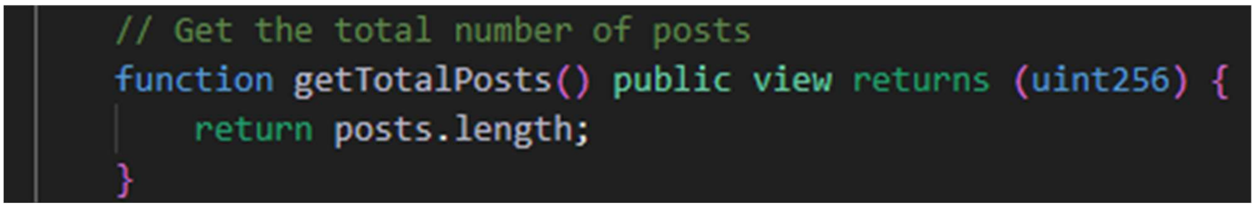
```
// Dislike a post
function dislikePost(uint256 index) public {    infinite gas
    require(index < posts.length, "Post does not exist");
    posts[index].dislikes++;
}
```

6. ‘editPost’ Function : its Allows the author of a post to edit its content.



```
// Edit a post by the owner
function editPost(uint256 index, string memory _newMessage) public {    infin
    require(index < posts.length, "Post does not exist");
    Post storage post = posts[index];
    require(post.author == msg.sender, "Only the owner can edit this post");

    post.message = _newMessage;
    post.lastModified = block.timestamp;
}
```

7. ‘getTotalPosts’ Function : its Provides the total number of posts on the platform.

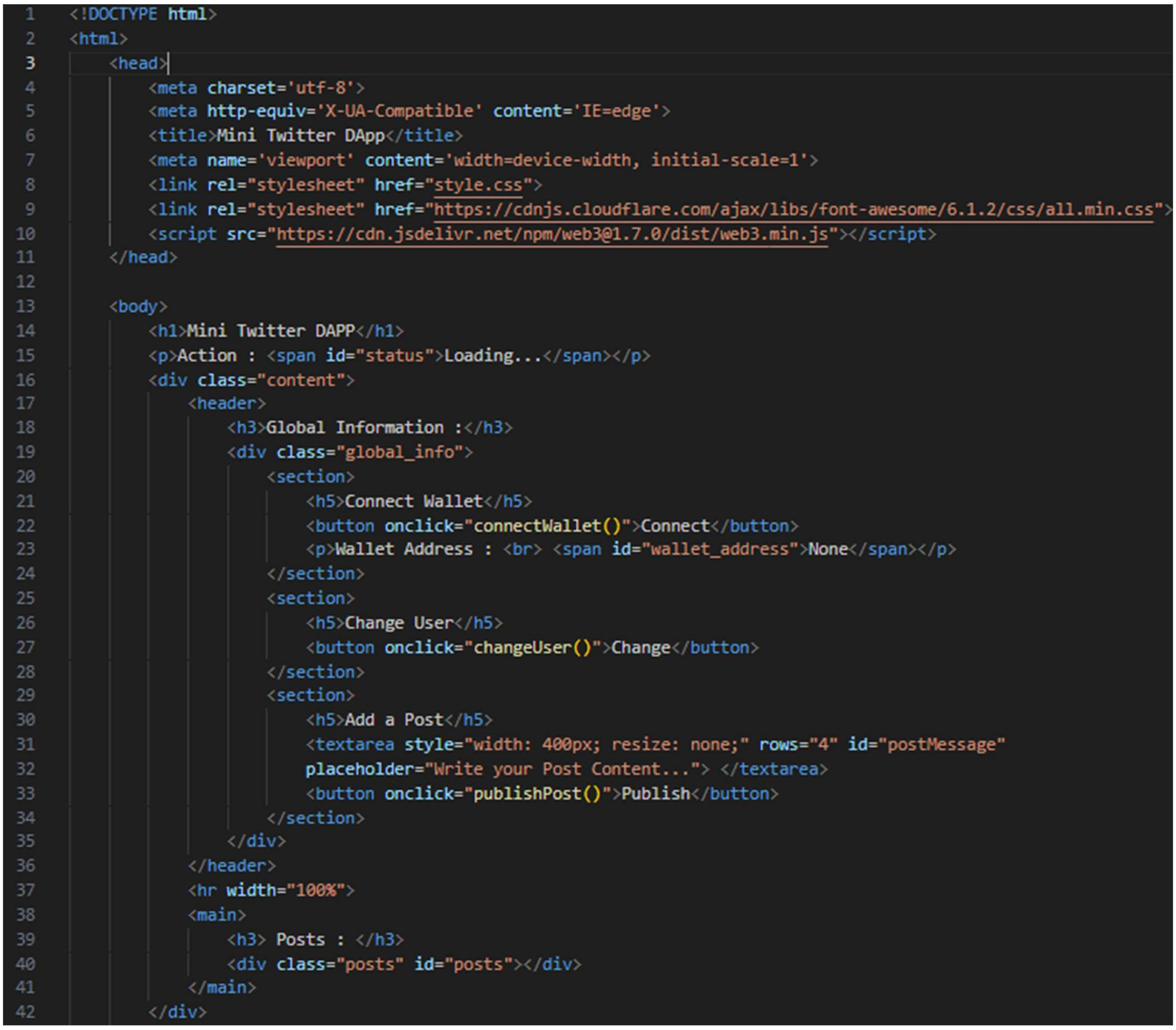


```
    // Get the total number of posts
    function getTotalPosts() public view returns (uint256) {
        return posts.length;
    }
```

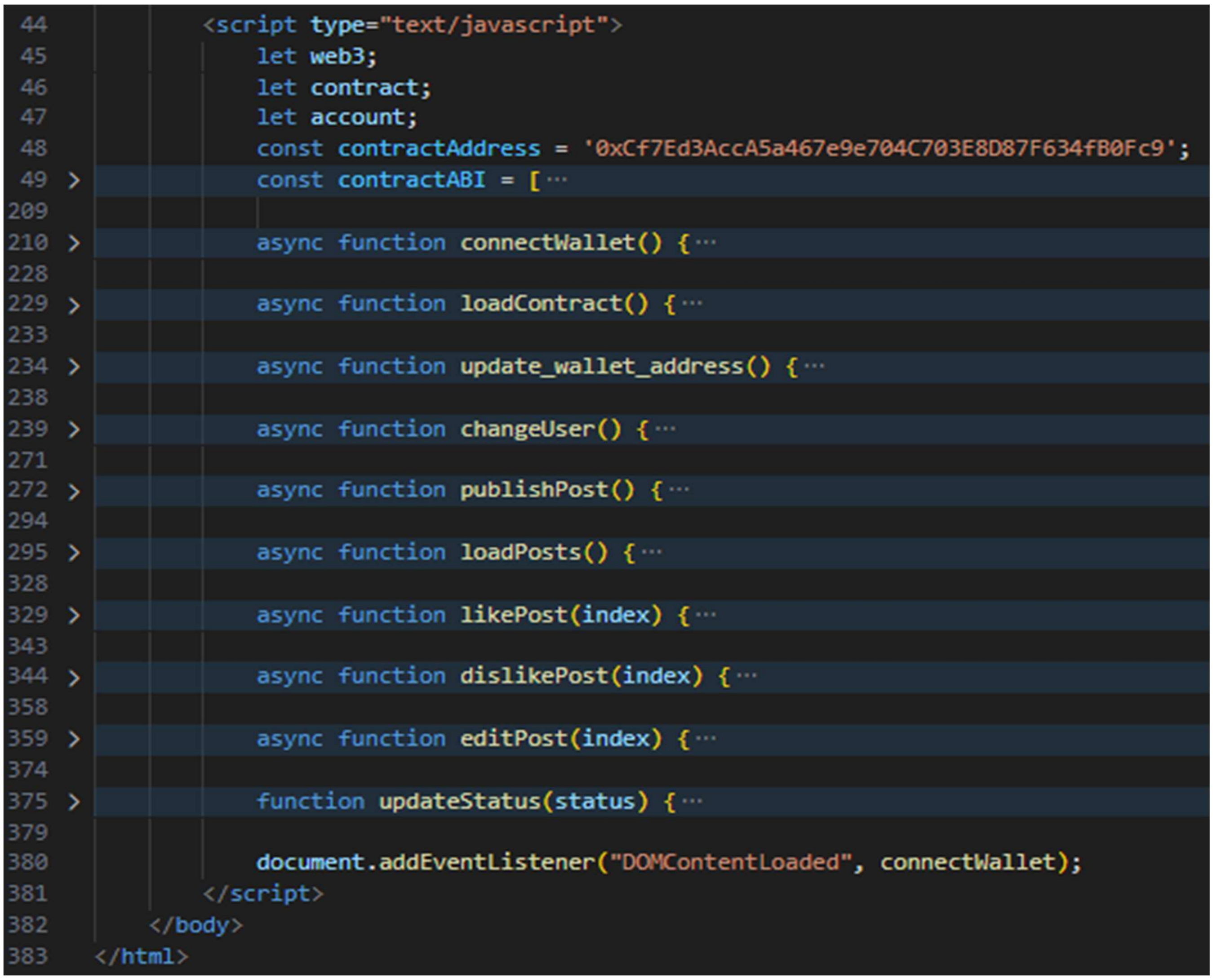
➔ Interface :

The user interface is developed with a focus on ease of use and accessibility, incorporating Web3.js to manage blockchain interactions. Key features of the interface include:

- Wallet Connection: Users connect their Ethereum wallet (MetaMask) to authenticate with the DApp, which assigns their account as their identity.
- Account Switching: The DApp includes an account-switching function, allowing users to seamlessly change accounts and continue interacting with the platform.
- User Interactions: Users can publish posts, react to posts (like/dislike), and edit their own posts directly from the interface. Feedback messages guide the user through actions and inform them of success or errors.

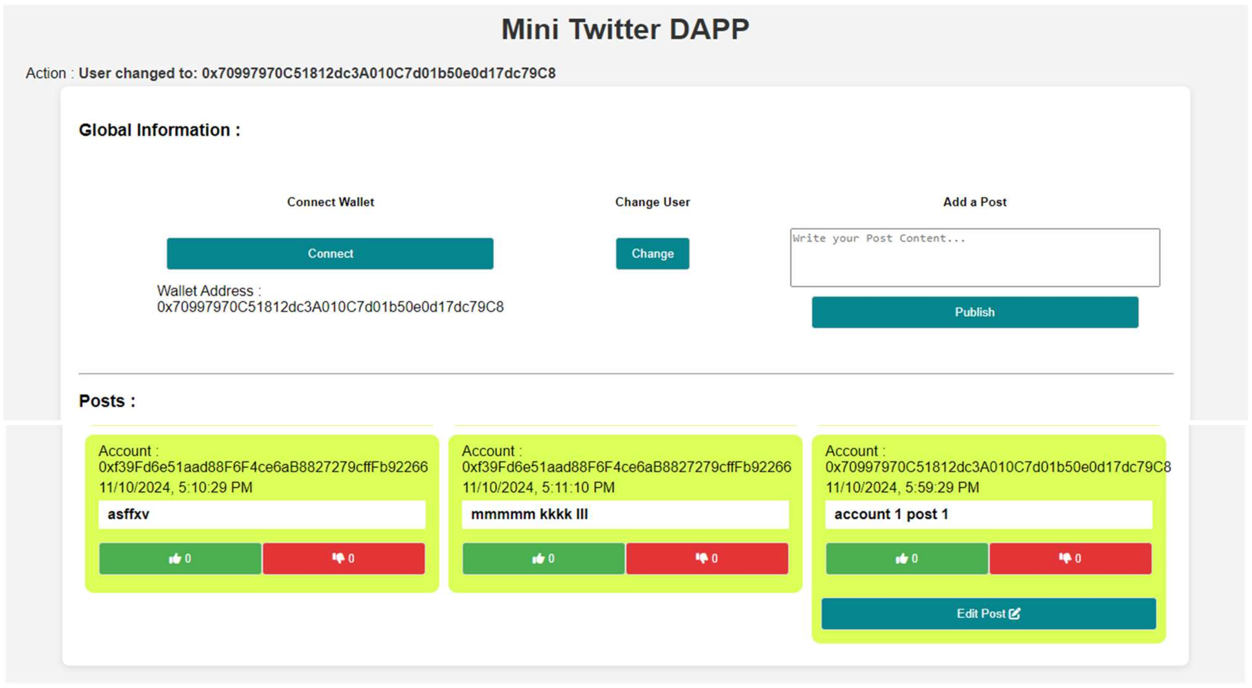


```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset='utf-8'>
        <meta http-equiv='X-UA-Compatible' content='IE=edge'>
        <title>Mini Twitter DApp</title>
        <meta name='viewport' content='width=device-width, initial-scale=1'>
        <link rel="stylesheet" href="style.css">
        <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.1.2/css/all.min.css">
        <script src="https://cdn.jsdelivr.net/npm/web3@1.7.0/dist/web3.min.js"></script>
    </head>

    <body>
        <h1>Mini Twitter DAPP</h1>
        <p>Action : <span id="status">Loading...</span></p>
        <div class="content">
            <header>
                <h3>Global Information :</h3>
                <div class="global_info">
                    <section>
                        <h5>Connect Wallet</h5>
                        <button onclick="connectWallet()">Connect</button>
                        <p>Wallet Address : <br> <span id="wallet_address">None</span></p>
                    </section>
                    <section>
                        <h5>Change User</h5>
                        <button onclick="changeUser()">Change</button>
                    </section>
                    <section>
                        <h5>Add a Post</h5>
                        <textarea style="width: 400px; resize: none;" rows="4" id="postMessage"
                        placeholder="Write your Post Content..."> </textarea>
                        <button onclick="publishPost()">Publish</button>
                    </section>
                </div>
            </header>
            <hr width="100%">
            <main>
                <h3> Posts : </h3>
                <div class="posts" id="posts"></div>
            </main>
        </div>
```



```
        <script type="text/javascript">
            let web3;
            let contract;
            let account;
            const contractAddress = '0xCf7Ed3AccA5a467e9e704C703E8D87F634fB0Fc9';
            const contractABI = [⋯

            async function connectWallet() {⋯

            async function loadContract() {⋯

            async function update_wallet_address() {⋯

            async function changeUser() {⋯

            async function publishPost() {⋯

            async function loadPosts() {⋯

            async function likePost(index) {⋯

            async function dislikePost(index) {⋯

            async function editPost(index) {⋯

            function updateStatus(status) {⋯

            document.addEventListener("DOMContentLoaded", connectWallet);
        </script>
    </body>
</html>
```

The Result :



Mini Twitter DAPP

Action : **User changed to: 0x70997970C51812dc3A010C7d01b50e0d17dc79C8**

Global Information :

Connect Wallet

Connect

Wallet Address :
0x70997970C51812dc3A010C7d01b50e0d17dc79C8

Change User

Change

Add a Post

Write your Post Content...

Publish

Posts :

Account :
0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
11/10/2024, 5:10:29 PM
asffxv
0 | 0

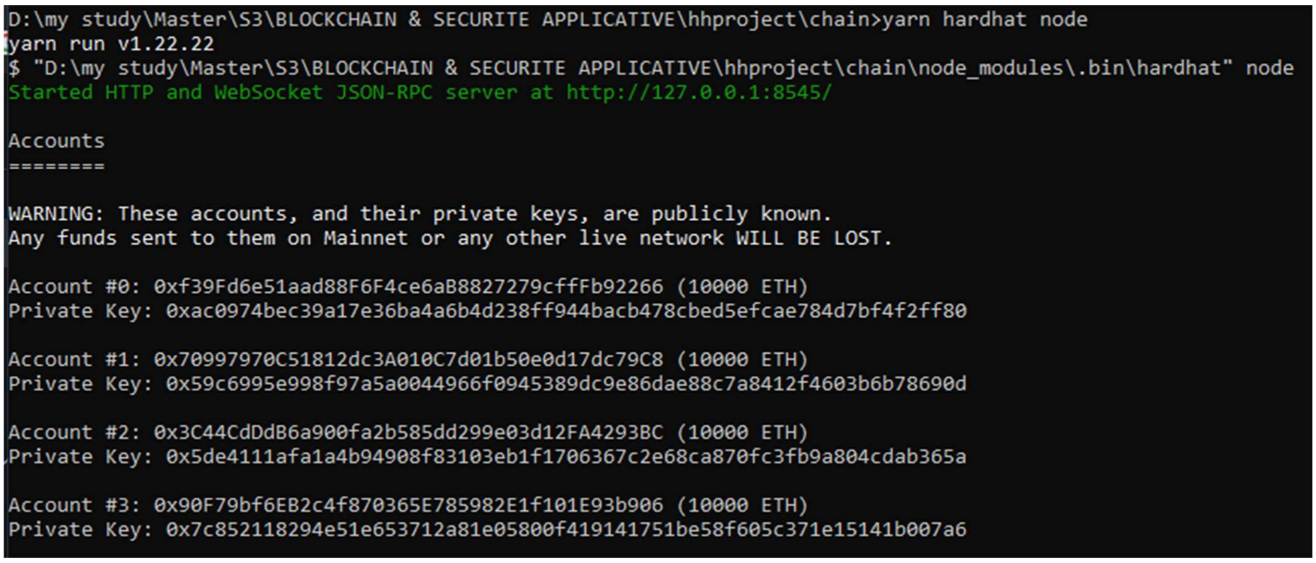
Account :
0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
11/10/2024, 5:11:10 PM
mmmmm kkkk lll
0 | 0

Account :
0x70997970C51812dc3A010C7d01b50e0d17dc79C8
11/10/2024, 5:59:29 PM
account 1 post 1
0 | 0
Edit Post

➔ Deployment and Tests :

- <u>**Deployment**</u>: The contract deploys on a test network such as Hardhat's local blockchain for development and testing without mainnet gas fees.

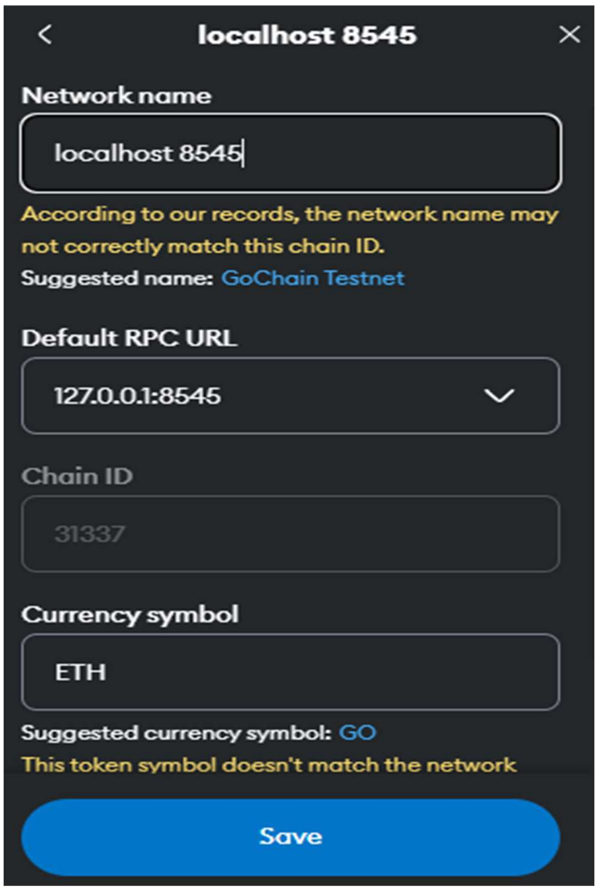
1. **Setup Hardhat** : there is 20 account



```
D:\my study\Master\S3\BLOCKCHAIN & SECURITE APPLICATIVE\hhproject\chain>yarn hardhat node
yarn run v1.22.22
$ "D:\my study\Master\S3\BLOCKCHAIN & SECURITE APPLICATIVE\hhproject\chain\node_modules\.bin\hardhat" node
Started HTTP and WebSocket JSON-RPC server at http://127.0.0.1:8545/

Accounts
========

WARNING: These accounts, and their private keys, are publicly known.
Any funds sent to them on Mainnet or any other live network WILL BE LOST.

Account #0: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266 (10000 ETH)
Private Key: 0xac0974bec39a17e36ba4a6b4d238ff944bacb478cbed5efcae784d7bf4f2ff80

Account #1: 0x70997970C51812dc3A010C7d01b50e0d17dc79C8 (10000 ETH)
Private Key: 0x59c6995e998f97a5a0044966f0945389dc9e86dae88c7a8412f4603b6b78690d

Account #2: 0x3C44CdDdB6a900fa2b585dd299e03d12FA4293BC (10000 ETH)
Private Key: 0x5de4111afa1a4b94908f83103eb1f1706367c2e68ca870fc3fb9a804cdab365a

Account #3: 0x90F79bf6EB2c4f870365E785982E1f101E93b906 (10000 ETH)
Private Key: 0x7c852118294e51e653712a81e05800f419141751be58f605c371e15141b007a6
```

2. **Connect to MetaMask**:

   ○ Add Hardhat's local network in MetaMask (http://127.0.0.1:8545, Chain ID 31337).



localhost 8545

Network name: localhost 8545

According to our records, the network name may not correctly match this chain ID.

Suggested name: GoChain Testnet

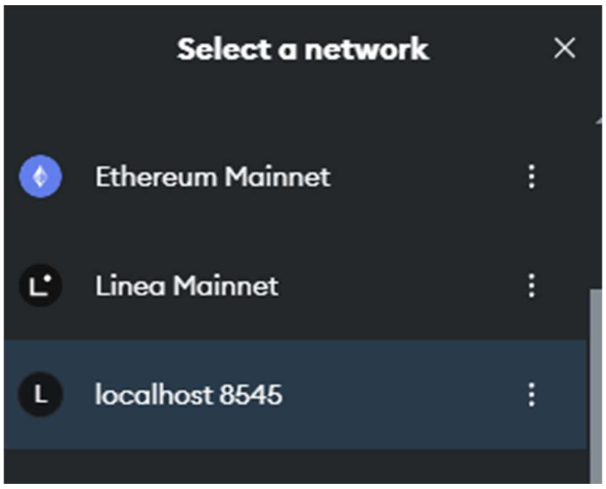
Default RPC URL: 127.0.0.1:8545

Chain ID: 31337

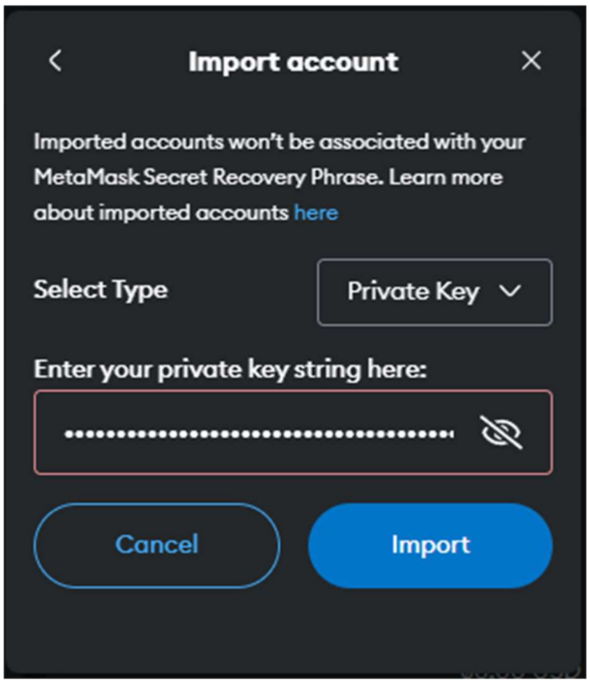
Currency symbol: ETH

Suggested currency symbol: GO

This token symbol doesn't match the network



- Import Hardhat-generated test accounts into MetaMask for testing contract interactions.



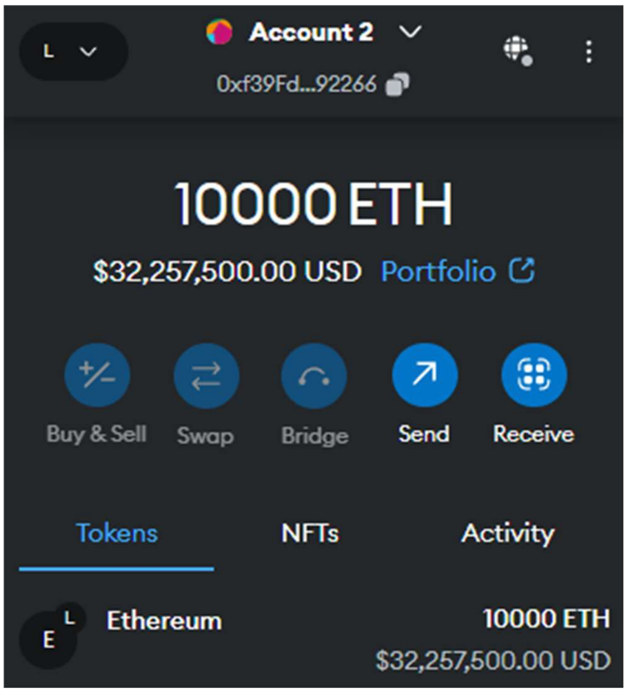
Import account

Imported accounts won't be associated with your MetaMask Secret Recovery Phrase. Learn more about imported accounts here

Select Type: Private Key

Enter your private key string here:

••••••••••••••••••••••••••••••••••••••



Account 2

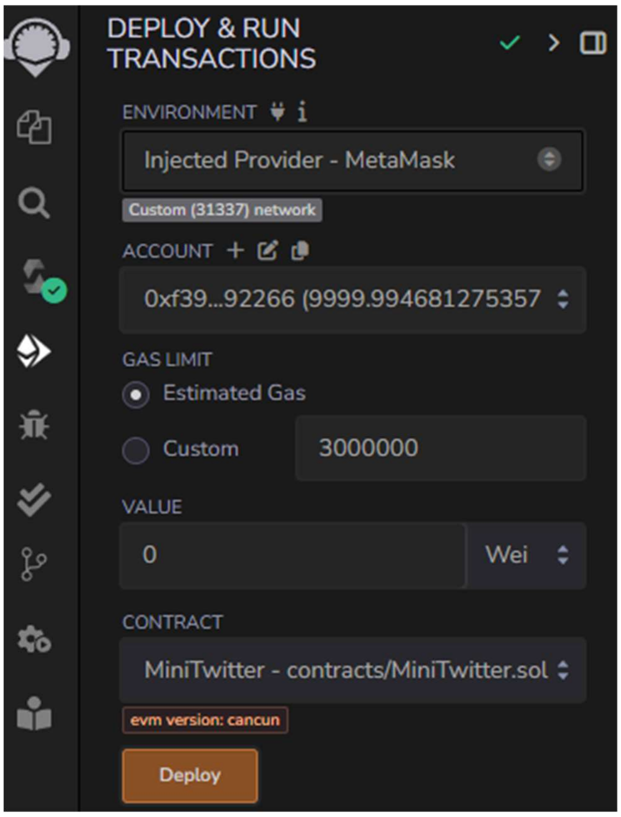
0xf39Fd...92266

10000 ETH

$32,257,500.00 USD

Ethereum 10000 ETH $32,257,500.00 USD

- Deploy the smart contract in REMIX IDE using the environment : 'integrated provider – metamask'.



DEPLOY & RUN TRANSACTIONS

ENVIRONMENT: Injected Provider - MetaMask

Custom (31337) network

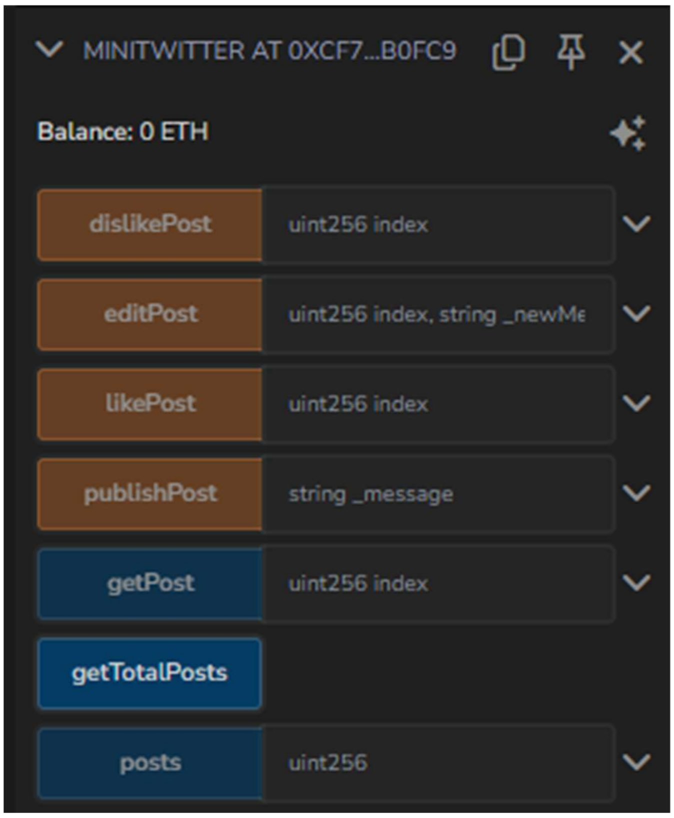
ACCOUNT: 0xf39...92266 (9999.994681275357

GAS LIMIT: Estimated Gas (selected); Custom 3000000

VALUE: 0 Wei

CONTRACT: MiniTwitter - contracts/MiniTwitter.sol

evm version: cancun



MINITWITTER AT 0XCF7...B0FC9

Balance: 0 ETH

dislikePost: uint256 index

editPost: uint256 index, string _newMe

likePost: uint256 index

publishPost: string _message

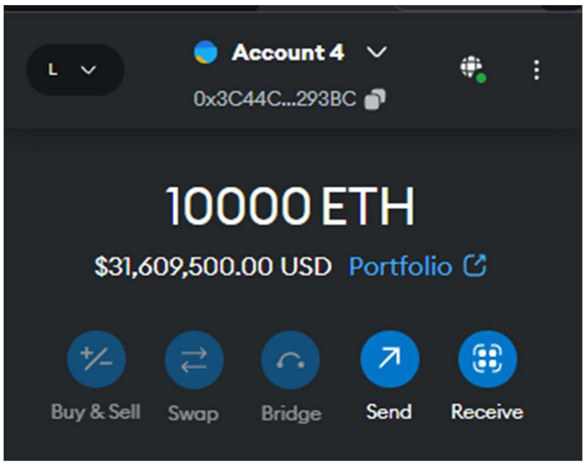
getPost: uint256 index

getTotalPosts

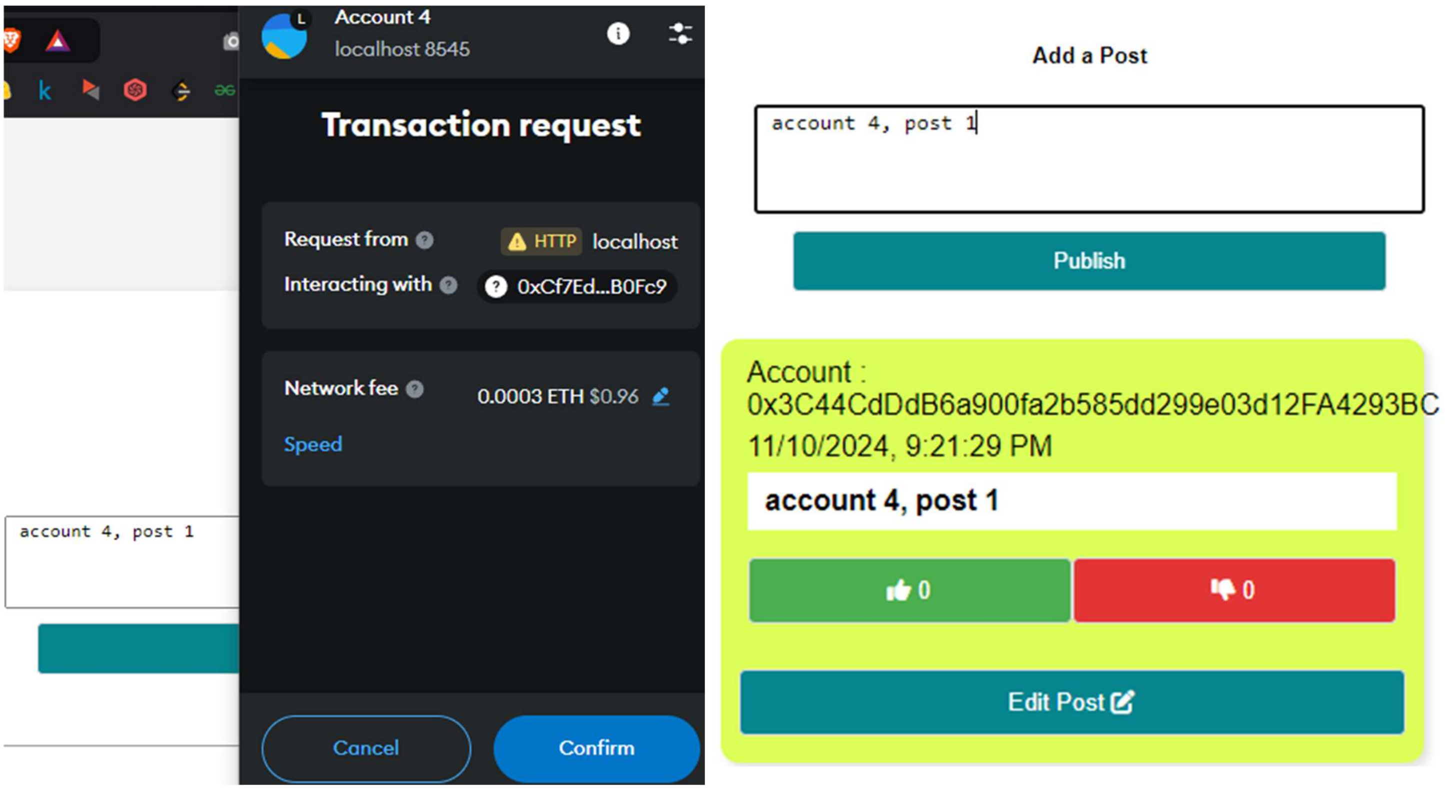
posts: uint256

- Run the Interface using XAMPP.

- **Testing**: Both manual and automated testing approaches were used. Manual testing was performed to verify wallet connectivity, post publishing, editing, and reactions. Automated tests were implemented to validate the contract's functions for data integrity, security, and user permissions.
  - **Connect with the account 4 :**



  - **Publish a post :**

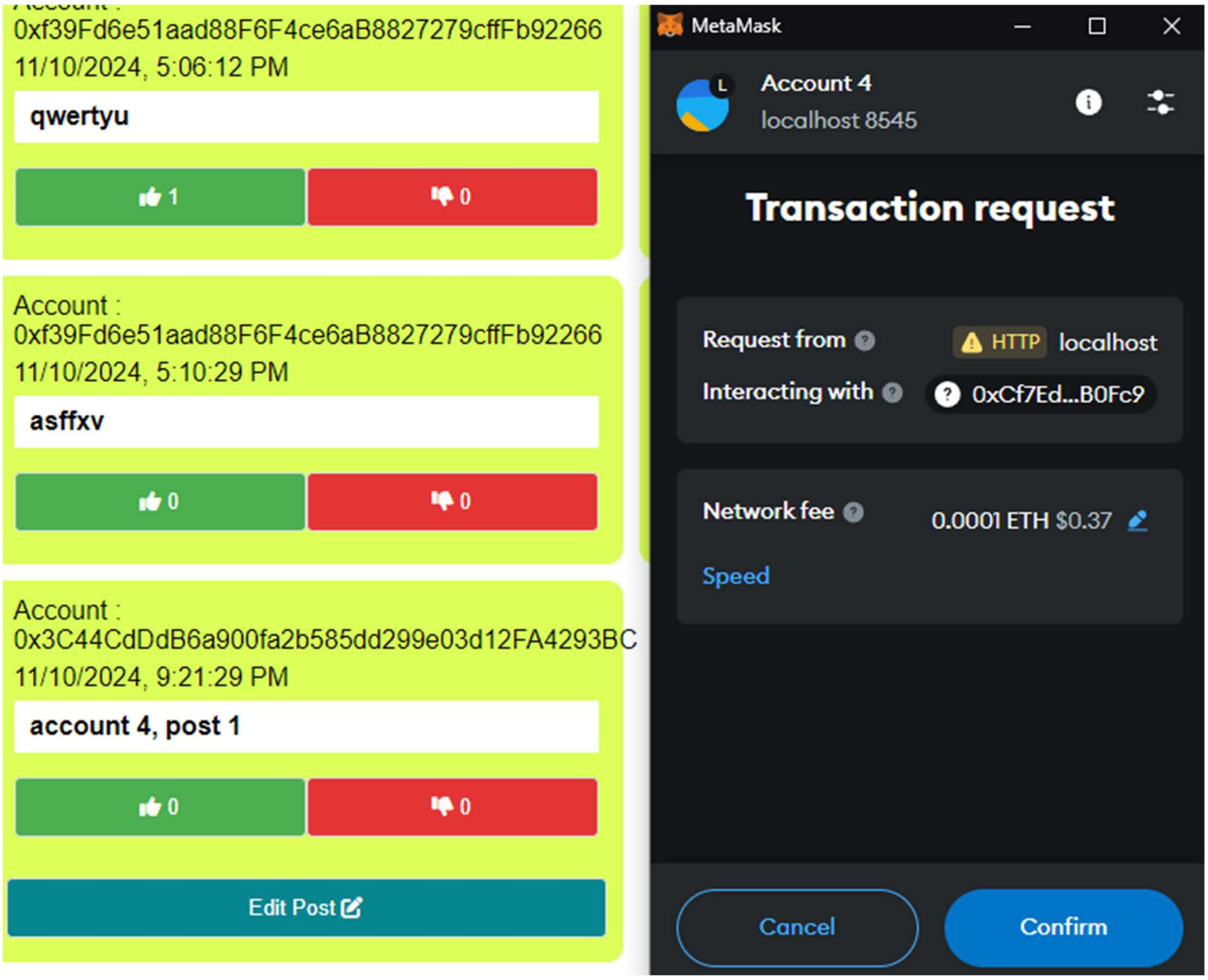


Request from HTTP localhost
Interacting with 0xCf7Ed...B0Fc9
Network fee 0.0003 ETH $0.96
Speed

account 4, post 1

Account : 0x3C44CdDdB6a900fa2b585dd299e03d12FA4293BC
11/10/2024, 9:21:29 PM
account 4, post 1
0 0

- **Add a Like :**

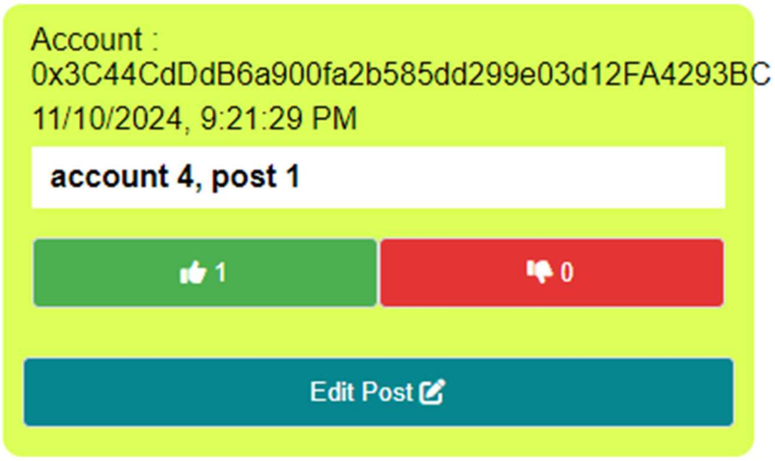


Account :
0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
11/10/2024, 5:06:12 PM
qwertyu

Account :
0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
11/10/2024, 5:10:29 PM
asffxv

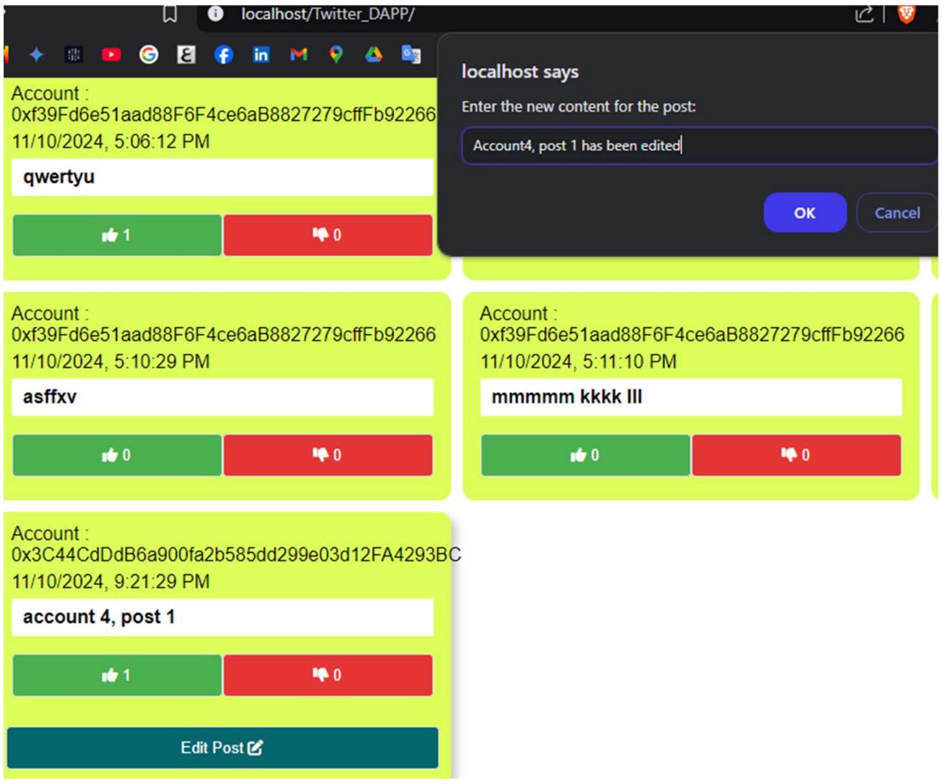
Account :
0x3C44CdDdB6a900fa2b585dd299e03d12FA4293BC
11/10/2024, 9:21:29 PM
account 4, post 1

MetaMask
Account 4
localhost 8545
Transaction request
Request from: HTTP localhost
Interacting with: 0xCf7Ed...B0Fc9
Network fee: 0.0001 ETH $0.37
Speed



Account :
0x3C44CdDdB6a900fa2b585dd299e03d12FA4293BC
11/10/2024, 9:21:29 PM
account 4, post 1

- **Edit the Post :**


localhost/Twitter_DAPP/

localhost says

Enter the new content for the post:

Account4, post 1 has been edited

OK Cancel

Account :
0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
11/10/2024, 5:06:12 PM

qwertyu

1 0

Account :
0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
11/10/2024, 5:10:29 PM

asffxv

0 0

Account :
0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
11/10/2024, 5:11:10 PM

mmmmm kkkk lll

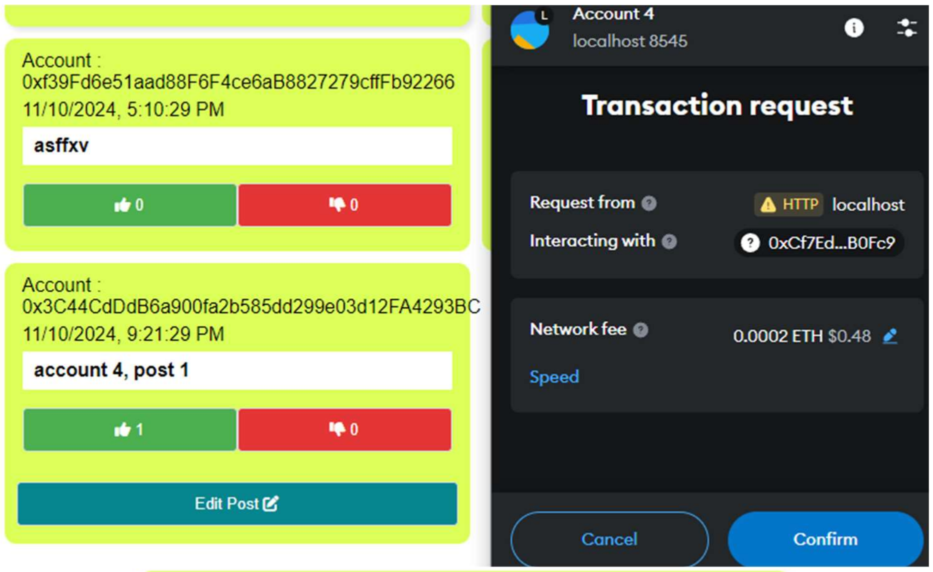
0 0

Account :
0x3C44CdDdB6a900fa2b585dd299e03d12FA4293BC
11/10/2024, 9:21:29 PM

account 4, post 1

1 0

Edit Post


Account :
0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
11/10/2024, 5:10:29 PM

asffxv

0 0

Account :
0x3C44CdDdB6a900fa2b585dd299e03d12FA4293BC
11/10/2024, 9:21:29 PM

account 4, post 1

1 0

Edit Post

Account 4
localhost 8545

Transaction request

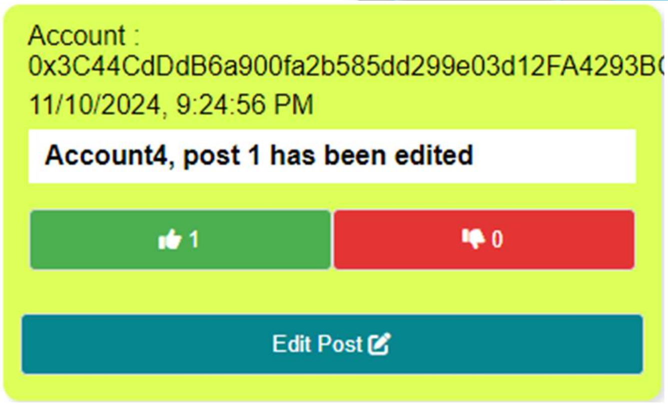
Request from HTTP localhost

Interacting with 0xCf7Ed...B0Fc9

Network fee 0.0002 ETH $0.48

Speed

Cancel Confirm

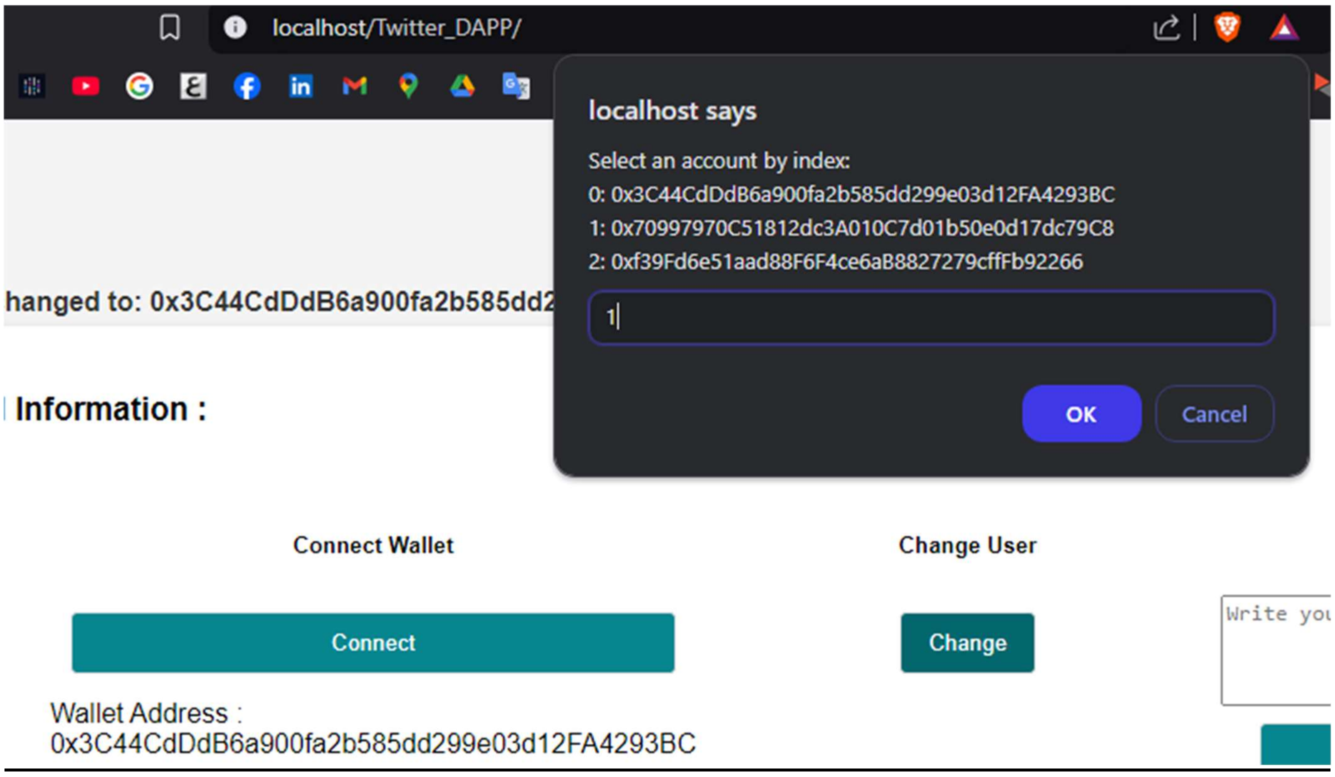

Account :
0x3C44CdDdB6a900fa2b585dd299e03d12FA4293B
11/10/2024, 9:24:56 PM

Account4, post 1 has been edited

1 0

Edit Post

- **Change the account :**



localhost/Twitter_DAPP/

localhost says

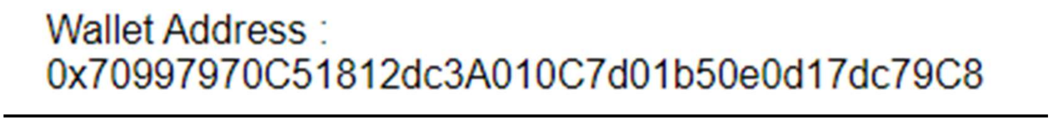
Select an account by index:
0: 0x3C44CdDdB6a900fa2b585dd299e03d12FA4293BC
1: 0x70997970C51812dc3A010C7d01b50e0d17dc79C8
2: 0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266

1

hanged to: 0x3C44CdDdB6a900fa2b585dd2

Information :

Wallet Address :
0x3C44CdDdB6a900fa2b585dd299e03d12FA4293BC



Wallet Address :
0x70997970C51812dc3A010C7d01b50e0d17dc79C8

Conclusion :

The Mini Twitter DApp demonstrates how blockchain can be used to create a decentralized and user-controlled social media platform, providing users with transparency, security, and ownership over their interactions. Through a seamless integration of Ethereum blockchain, Web3.js, and MetaMask, this DApp delivers a decentralized alternative to traditional social platforms.