PES UNIVERSITY

100 feet Ring Road, BSK 3rd Stage

Bengaluru 560085

Department of Computer Science and Engineering

Department of Computer Science and Engineering

B. Tech. CSE - 6th Semester

Jan – May 2025

UE22CS342BA5

BLOCKCHAIN

PROJECT REPORT

on

# *Blockchain based social media Dapp*

PATHIPATI KRISHNA REVANTH - PES1UG22CS409

PAVAN CHANDU C S - PES1UG22CS410

PRAGNAN M U - PES1UG22CS421

K R ADITHYA – PES2UG22CS244

# Blockchain-Based Social Media DApp

## Abstract

In the fast-changing world of decentralized technologies, this project introduces a censorship-free decentralized social media platform built as a Decentralized Application (DApp). The project uses blockchain to establish a secure, transparent, and tamper-proof system where users can share content, like, and comment without centralized authority. Smart contracts in Solidity and MetaMask integration offer a trustless and user-managed environment.

## 1. Introduction

Decentralized social media platforms tend to be criticized on the basis of censorship, data leaks, and control over content. Decentralized applications (DApps) provide a better alternative by utilizing blockchain to decentralize control. This project plans to create a social media DApp that maintains transparency, user control, and censorship resistance through storing interactions on-chain.

## 2. Existing System

Existing platforms like Facebook, Twitter, and Instagram are governed by central authorities, which exposes them to moderation bias, algorithmic manipulation, and single-point failures. While there are some Web3 social media initiatives, they tend to be limited in features or hard to use for non-technical users

## 3. Proposed System

This DApp provides a decentralized platform where users can:

- Create posts with text and media
- Like posts without duplicate likes
- Add comments on posts
- View posts and associated comments
- Interact directly with the Ethereum testnet (sepolia) blockchain via MetaMask

This is achieved through the use of smart contracts that maintain state immutably and verifiably on the blockchain.

---

## 4. System Architecture

The architecture consists of:-

- Frontend: Developed with React.js and styled with Tailwind CSS.
- Blockchain: Smart contracts on the Ethereum testnet.
- Wallet Integration: MetaMask is employed for signing and authentication of transactions.
- Deployment : Remix IDE

---

## 5. Implementation Details

The core logic resides in the Solidity smart contract SocialMedia.sol:

- **Post Creation:** Users invoke createPost() to publish content.
- **Like Mechanism:** likePost() ensures one like per user per post.
- **Commenting System:** addComment() stores comments per post, tracked by a mapping.
- **Events:** Emitted for post creation, liking, and commenting to notify the frontend.

The frontend interacts with the smart contract using ethers.js, and MetaMask handles wallet connections and transaction confirmations.

---

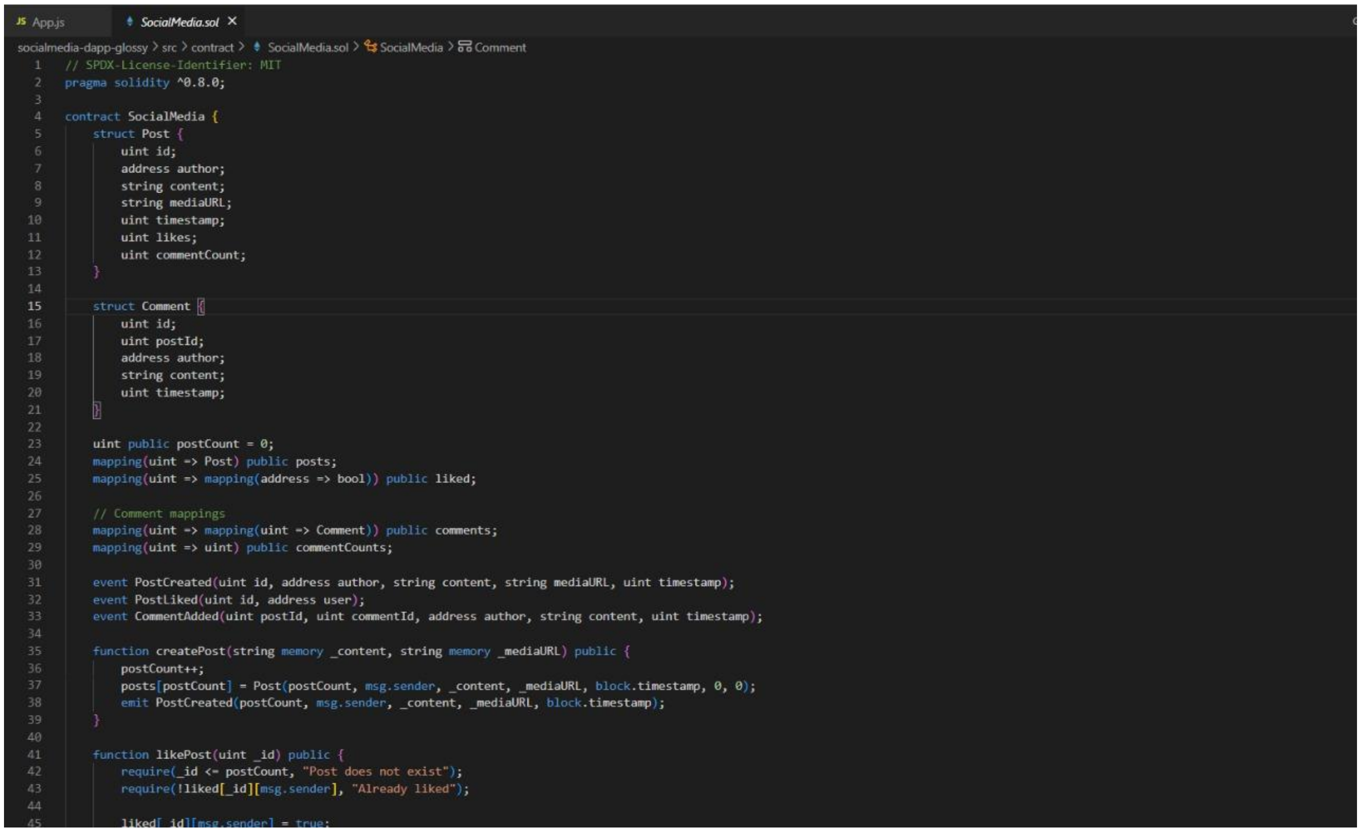
## 6. Technologies Used

- **Solidity:** For writing the smart contract
- **React.js:** For building the user interface
- **Tailwind CSS:** For responsive design
- **Ethers.js:** Blockchain interaction

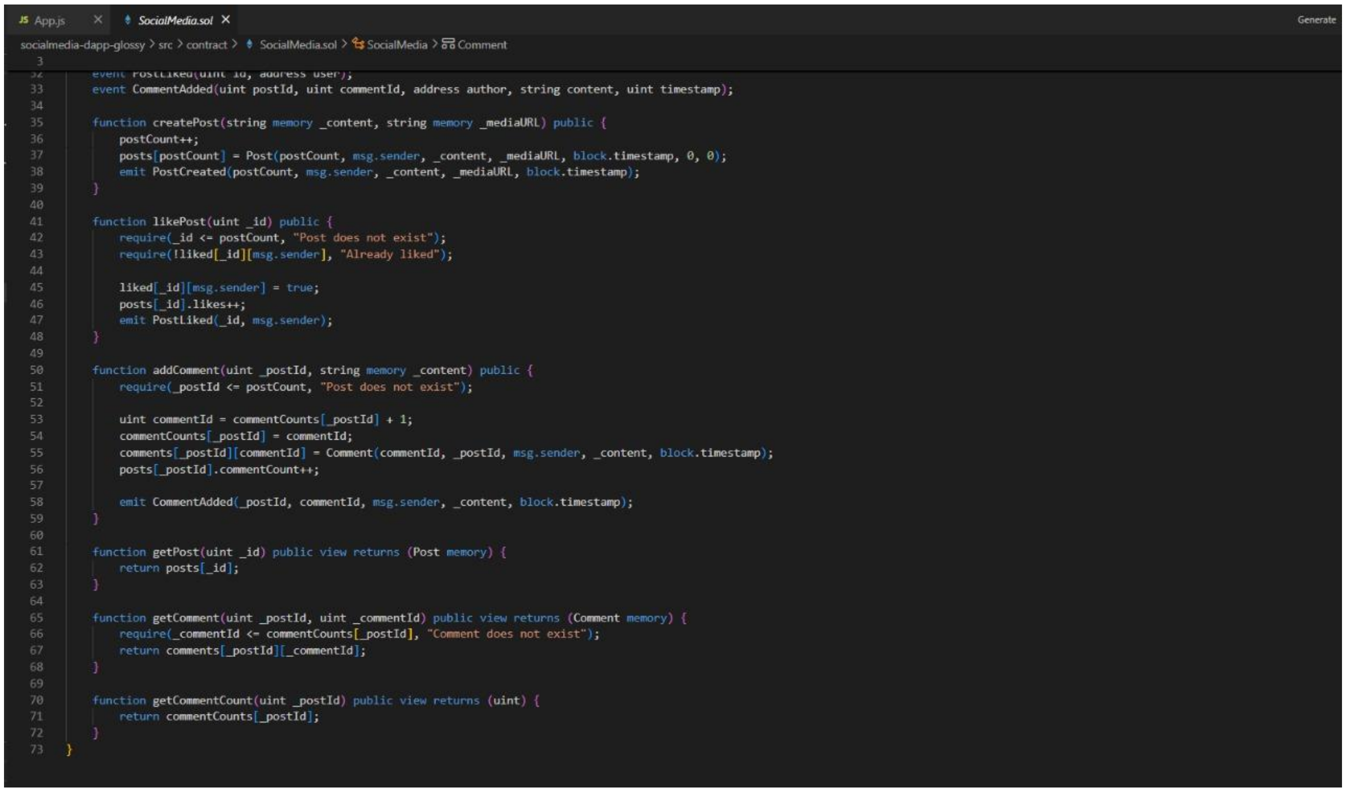
- **MetaMask:** Wallet integration
- **Remix IDE:** For deploying

---

## 7. Screenshots


App.js
SocialMedia.sol
socialmedia-dapp-glossy > src > contract > SocialMedia.sol > SocialMedia > Comment

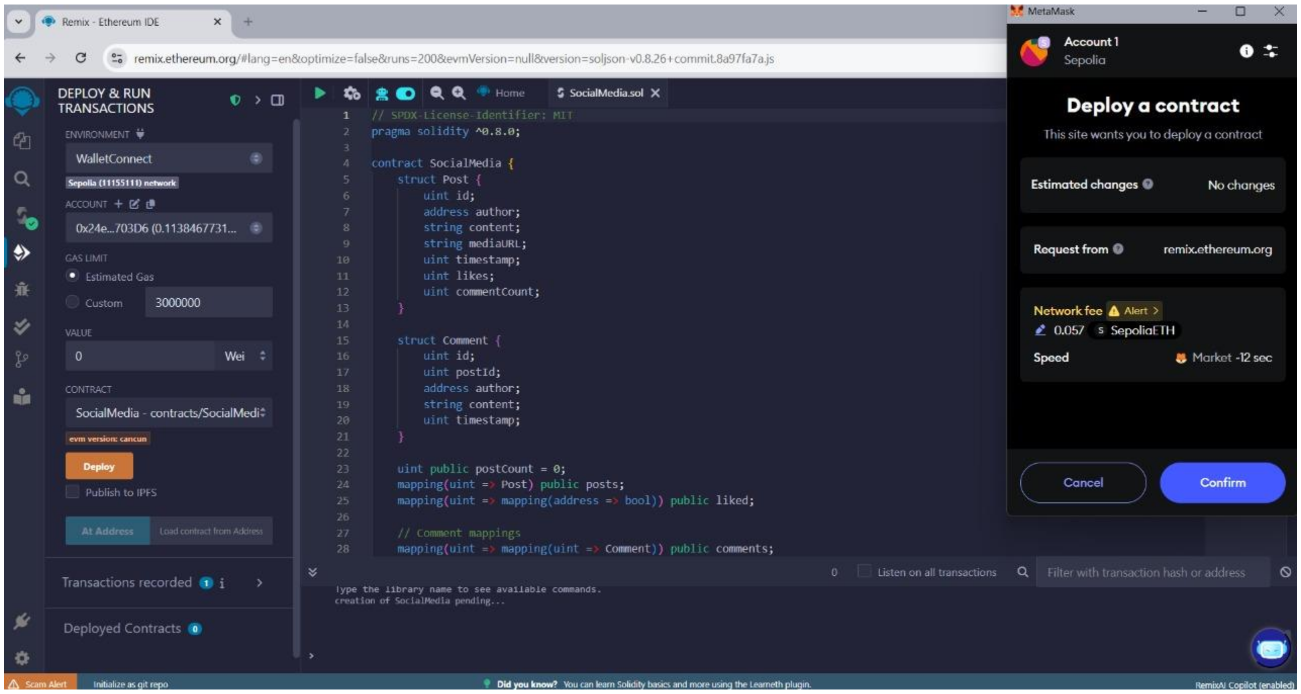
```solidity
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract SocialMedia {
    struct Post {
        uint id;
        address author;
        string content;
        string mediaURL;
        uint timestamp;
        uint likes;
        uint commentCount;
    }

    struct Comment {
        uint id;
        uint postId;
        address author;
        string content;
        uint timestamp;
    }

    uint public postCount = 0;
    mapping(uint => Post) public posts;
    mapping(uint => mapping(address => bool)) public liked;

    // Comment mappings
    mapping(uint => mapping(uint => Comment)) public comments;
    mapping(uint => uint) public commentCounts;

    event PostCreated(uint id, address author, string content, string mediaURL, uint timestamp);
    event PostLiked(uint id, address user);
    event CommentAdded(uint postId, uint commentId, address author, string content, uint timestamp);

    function createPost(string memory _content, string memory _mediaURL) public {
        postCount++;
        posts[postCount] = Post(postCount, msg.sender, _content, _mediaURL, block.timestamp, 0, 0);
        emit PostCreated(postCount, msg.sender, _content, _mediaURL, block.timestamp);
    }

    function likePost(uint _id) public {
        require(_id <= postCount, "Post does not exist");
        require(!liked[_id][msg.sender], "Already liked");

        liked[_id][msg.sender] = true;
```



App.js | SocialMedia.sol

socialmedia-dapp-glossy > src > contract > SocialMedia.sol > SocialMedia > Comment

```solidity
    event PostLiked(uint id, address user);
    event CommentAdded(uint postId, uint commentId, address author, string content, uint timestamp);

    function createPost(string memory _content, string memory _mediaURL) public {
        postCount++;
        posts[postCount] = Post(postCount, msg.sender, _content, _mediaURL, block.timestamp, 0, 0);
        emit PostCreated(postCount, msg.sender, _content, _mediaURL, block.timestamp);
    }

    function likePost(uint _id) public {
        require(_id <= postCount, "Post does not exist");
        require(!liked[_id][msg.sender], "Already liked");

        liked[_id][msg.sender] = true;
        posts[_id].likes++;
        emit PostLiked(_id, msg.sender);
    }

    function addComment(uint _postId, string memory _content) public {
        require(_postId <= postCount, "Post does not exist");

        uint commentId = commentCounts[_postId] + 1;
        commentCounts[_postId] = commentId;
        comments[_postId][commentId] = Comment(commentId, _postId, msg.sender, _content, block.timestamp);
        posts[_postId].commentCount++;

        emit CommentAdded(_postId, commentId, msg.sender, _content, block.timestamp);
    }

    function getPost(uint _id) public view returns (Post memory) {
        return posts[_id];
    }

    function getComment(uint _postId, uint _commentId) public view returns (Comment memory) {
        require(_commentId <= commentCounts[_postId], "Comment does not exist");
        return comments[_postId][_commentId];
    }

    function getCommentCount(uint _postId) public view returns (uint) {
        return commentCounts[_postId];
    }
}
```



Remix - Ethereum IDE

remix.ethereum.org/#lang=en&optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.26+commit.8a97fa7a.js

DEPLOY & RUN TRANSACTIONS

ENVIRONMENT: WalletConnect
Sepolia (11155111) network
ACCOUNT: 0x24e...703D6 (0.1138467731...
GAS LIMIT: (●) Estimated Gas ( ) Custom 3000000
VALUE: 0 Wei
CONTRACT: SocialMedia - contracts/SocialMedi
evm version: cancun
Deploy
Publish to IPFS
At Address | Load contract from Address
Transactions recorded 1
Deployed Contracts 0

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

contract SocialMedia {
    struct Post {
        uint id;
        address author;
        string content;
        string mediaURL;
        uint timestamp;
        uint likes;
        uint commentCount;
    }

    struct Comment {
        uint id;
        uint postId;
        address author;
        string content;
        uint timestamp;
    }

    uint public postCount = 0;
    mapping(uint => Post) public posts;
    mapping(uint => mapping(address => bool)) public liked;

    // Comment mappings
    mapping(uint => mapping(uint => Comment)) public comments;
```

```
Type the library name to see available commands.
creation of SocialMedia pending...
```

MetaMask

Account 1
Sepolia

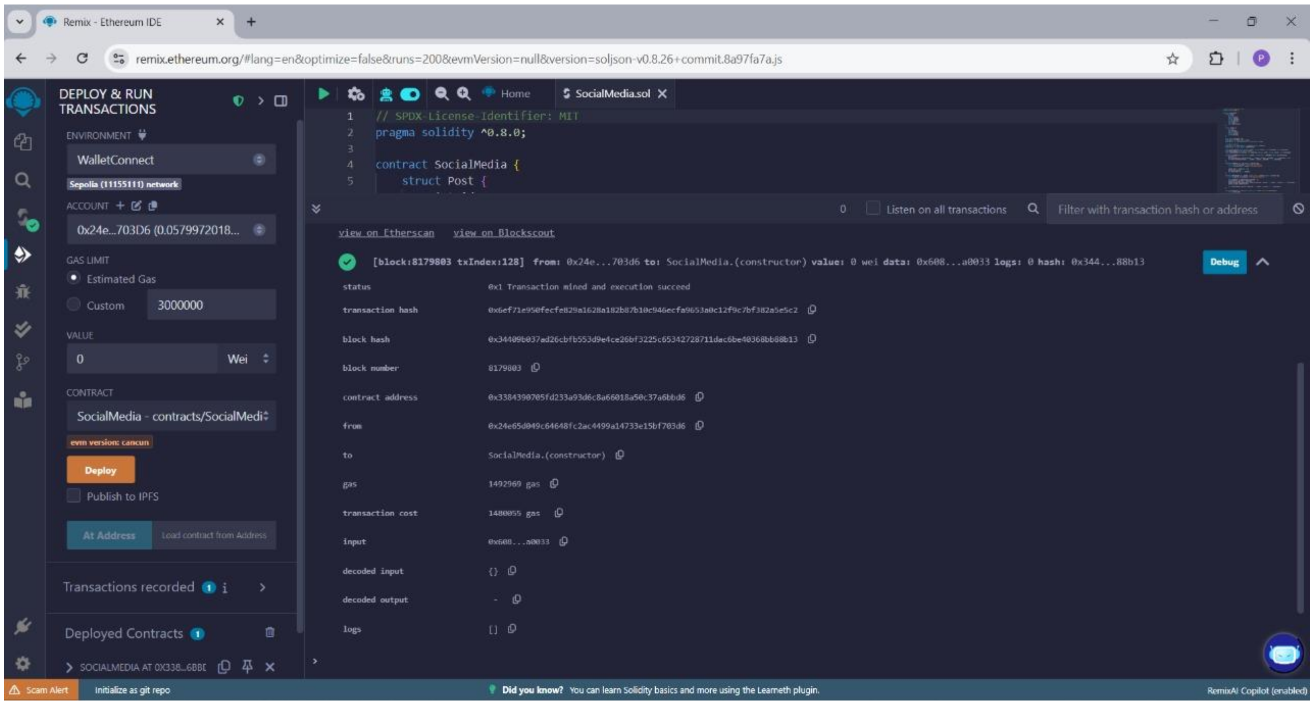
**Deploy a contract**

This site wants you to deploy a contract

Estimated changes: No changes
Request from: remix.ethereum.org
Network fee: Alert > 0.057 SepoliaETH
Speed: Market -12 sec

Cancel | Confirm

Home
SocialMedia.sol
0
Listen on all transactions
Filter with transaction hash or address
Scam Alert
Initialize as git repo
Did you know? You can learn Solidity basics and more using the Learneth plugin.
RemixAI Copilot (enabled)



DEPLOY & RUN TRANSACTIONS

ENVIRONMENT: WalletConnect

Sepolia (11155111) network

ACCOUNT: 0x24e...703D6 (0.0579972018...

GAS LIMIT: Estimated Gas

Custom 3000000

VALUE: 0 Wei

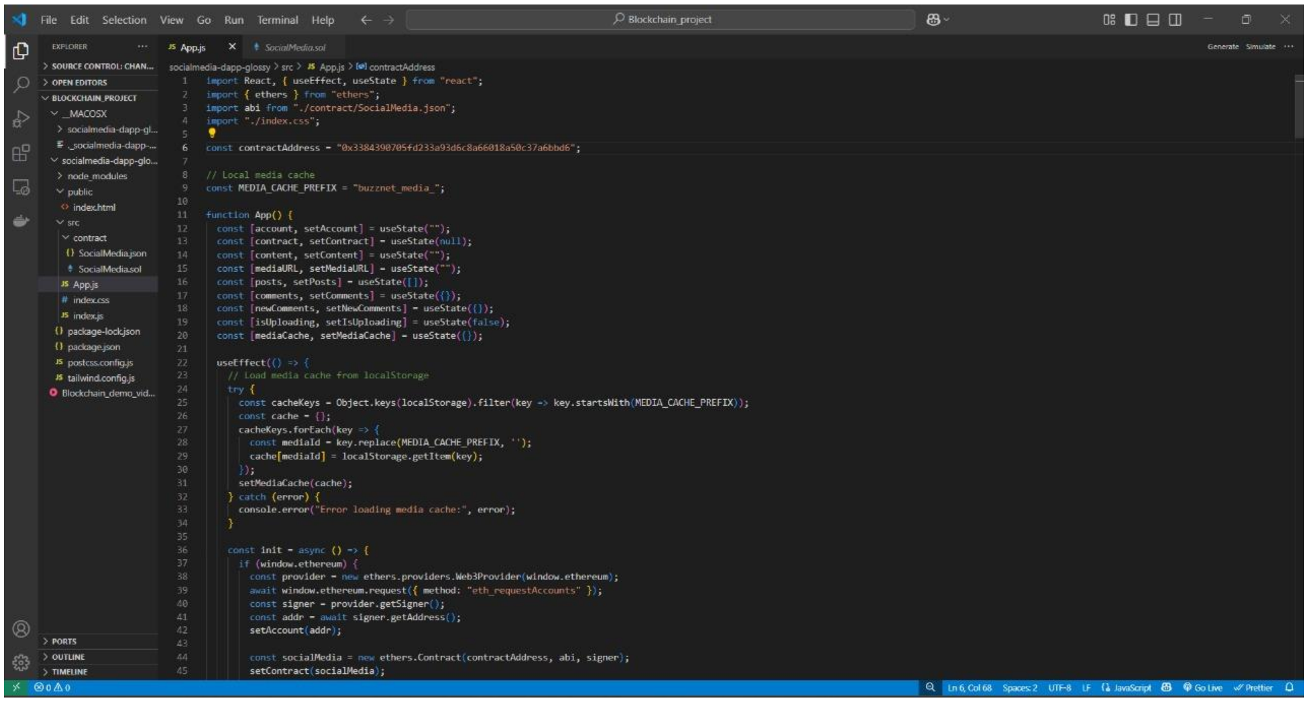
CONTRACT: SocialMedia - contracts/SocialMedia

evm version: cancun

Deployed Contracts: SOCIALMEDIA AT 0X338...6BBE

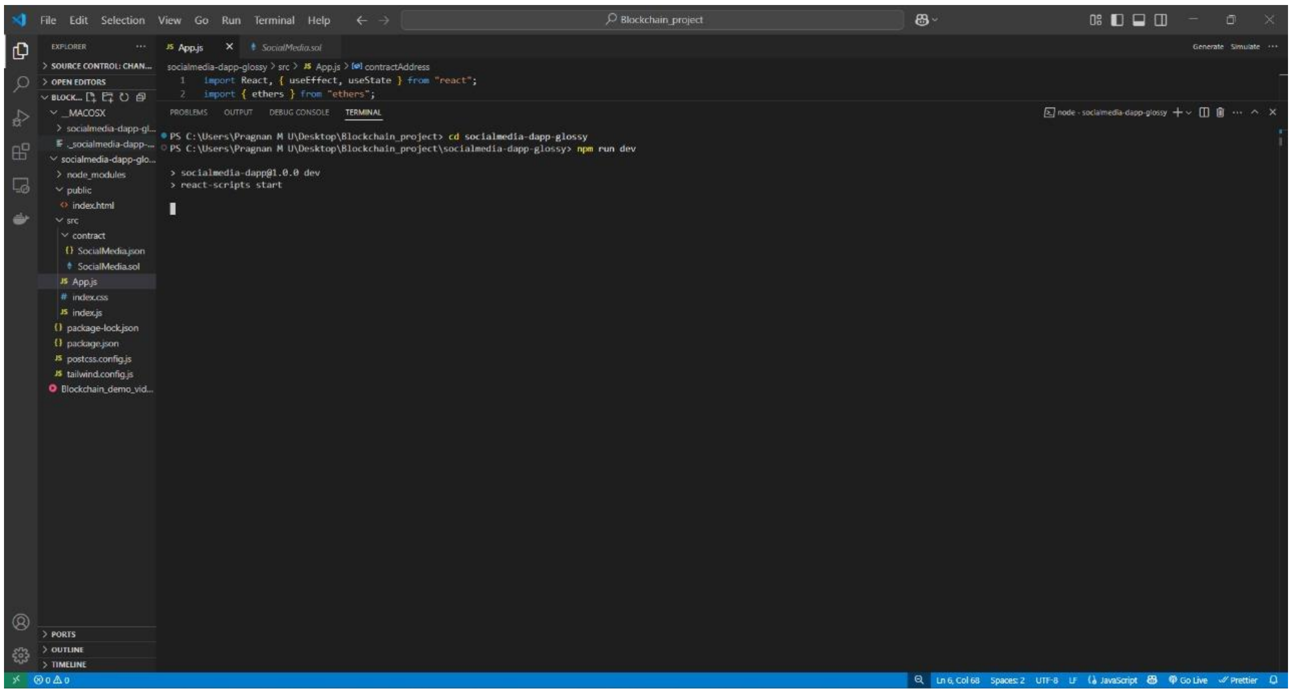
```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract SocialMedia {
    struct Post {
```

```
[block:8179803 txIndex:128] from: 0x24e...703d6 to: SocialMedia.(constructor) value: 0 wei data: 0x608...a0033 logs: 0 hash: 0x344...88b13
status              0x1 Transaction mined and execution succeed
transaction hash    0x6ef71e950fecfe829a1628a182b87b10c946ecfa9653a0c12f9c7bf382a5e5c2
block hash          0x34409b037ad26cbfb553d9e4ce26bf3225c65342728711dac6be40368bb88b13
block number        8179803
contract address    0x3384390705fd233a93d6c8a66018a50c37a6bbd6
from                0x24e65d049c64648fc2ac4499a14733e15bf703d6
to                  SocialMedia.(constructor)
gas                 1492969 gas
transaction cost    1480055 gas
input               0x608...a0033
decoded input       {}
decoded output       -
logs                []
```



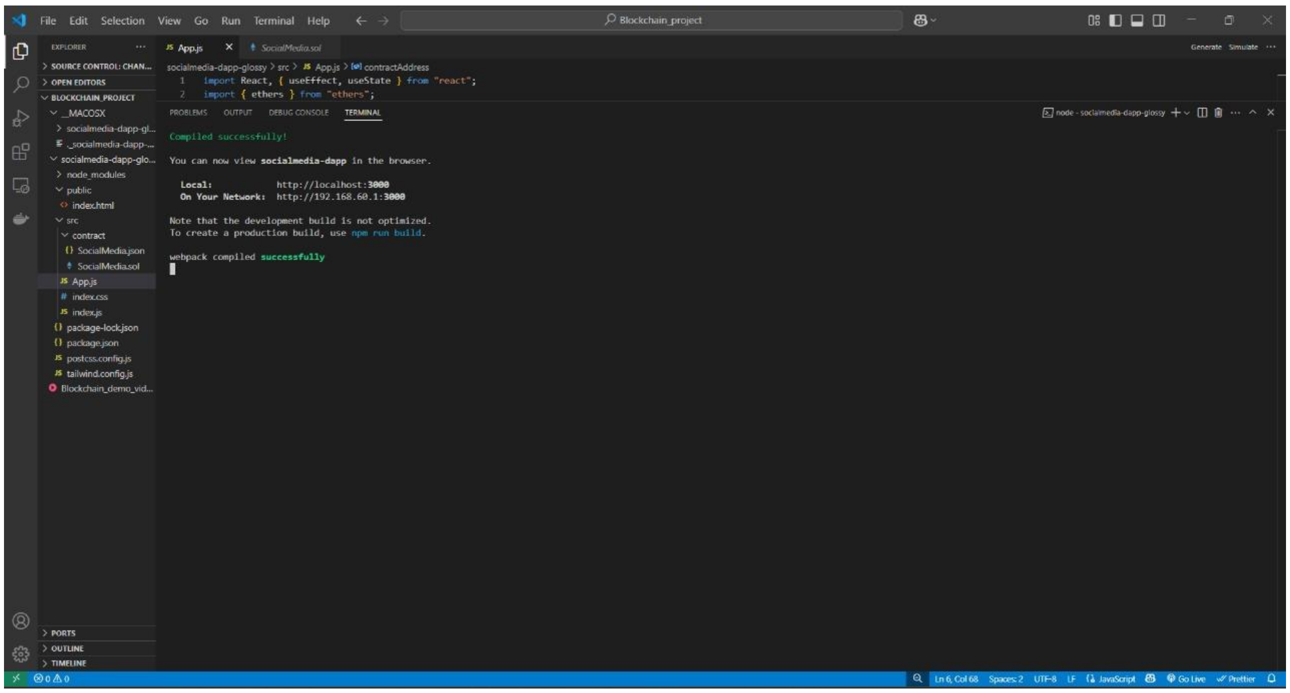
socialmedia-dapp-glossy > src > App.js > contractAddress

```javascript
import React, { useEffect, useState } from "react";
import { ethers } from "ethers";
import abi from "./contract/SocialMedia.json";
import "./index.css";

const contractAddress = "0x3384390705fd233a93d6c8a66018a50c37a6bbd6";

// Local media cache
const MEDIA_CACHE_PREFIX = "buzznet_media_";

function App() {
  const [account, setAccount] = useState("");
  const [contract, setContract] = useState(null);
  const [content, setContent] = useState("");
  const [mediaURL, setMediaURL] = useState("");
  const [posts, setPosts] = useState([]);
  const [comments, setComments] = useState({});
  const [newComments, setNewComments] = useState({});
  const [isUploading, setIsUploading] = useState(false);
  const [mediaCache, setMediaCache] = useState({});

  useEffect(() => {
    // Load media cache from localStorage
    try {
      const cacheKeys = Object.keys(localStorage).filter(key => key.startsWith(MEDIA_CACHE_PREFIX));
      const cache = {};
      cacheKeys.forEach(key => {
        const mediaId = key.replace(MEDIA_CACHE_PREFIX, '');
        cache[mediaId] = localStorage.getItem(key);
      });
      setMediaCache(cache);
    } catch (error) {
      console.error("Error loading media cache:", error);
    }

    const init = async () => {
      if (window.ethereum) {
        const provider = new ethers.providers.Web3Provider(window.ethereum);
        await window.ethereum.request({ method: "eth_requestAccounts" });
        const signer = provider.getSigner();
        const addr = await signer.getAddress();
        setAccount(addr);

        const socialMedia = new ethers.Contract(contractAddress, abi, signer);
        setContract(socialMedia);
```



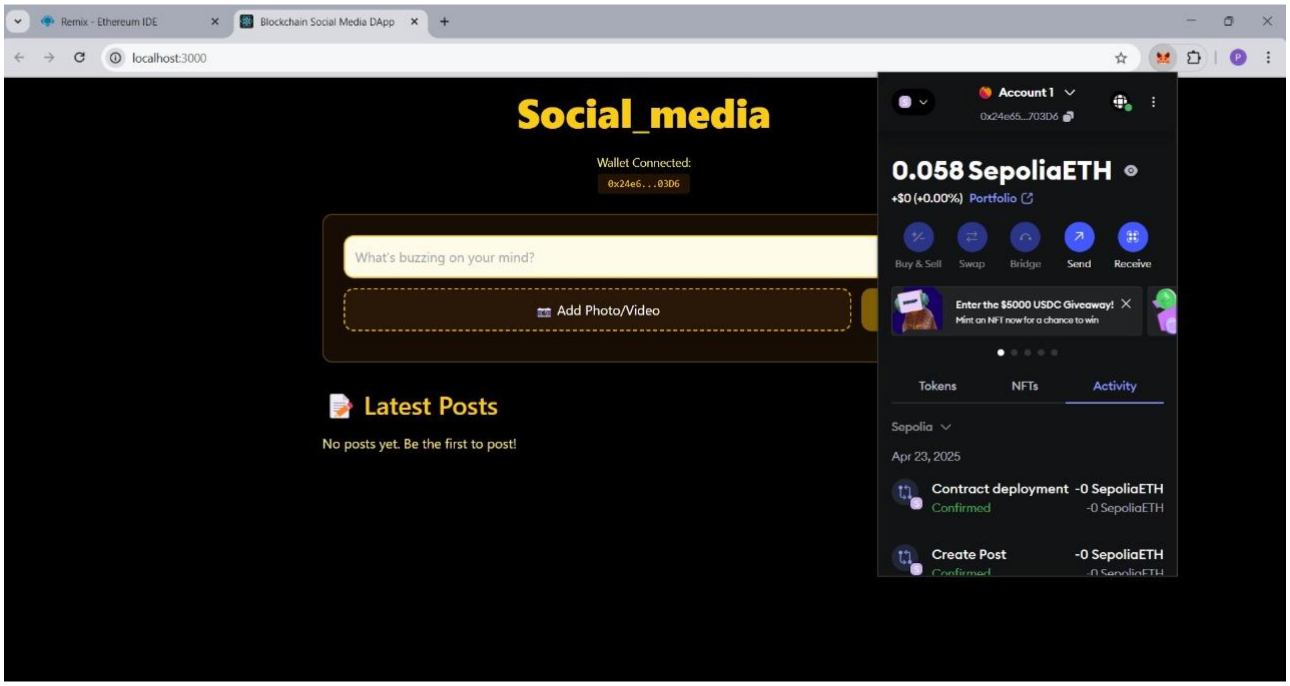
```
import React, { useEffect, useState } from "react";
import { ethers } from "ethers";
```

```
PS C:\Users\Pragnan M U\Desktop\Blockchain_project> cd socialmedia-dapp-glossy
PS C:\Users\Pragnan M U\Desktop\Blockchain_project\socialmedia-dapp-glossy> npm run dev

> socialmedia-dapp@1.0.0 dev
> react-scripts start
```



```
import React, { useEffect, useState } from "react";
import { ethers } from "ethers";
```

```
Compiled successfully!

You can now view socialmedia-dapp in the browser.

  Local:            http://localhost:3000
  On Your Network:  http://192.168.60.1:3000

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```


Remix - Ethereum IDE

Blockchain Social Media DApp

localhost:3000

Social_media

Wallet Connected:

0x24e6...03D6

What's buzzing on your mind?

📷 Add Photo/Video

📝 Latest Posts

No posts yet. Be the first to post!

Account 1

0x24e65...703D6

0.058 SepoliaETH

+$0 (+0.00%) Portfolio

Buy & Sell Swap Bridge Send Receive

Enter the $5000 USDC Giveaway!

Mint an NFT now for a chance to win

Tokens NFTs Activity

Sepolia

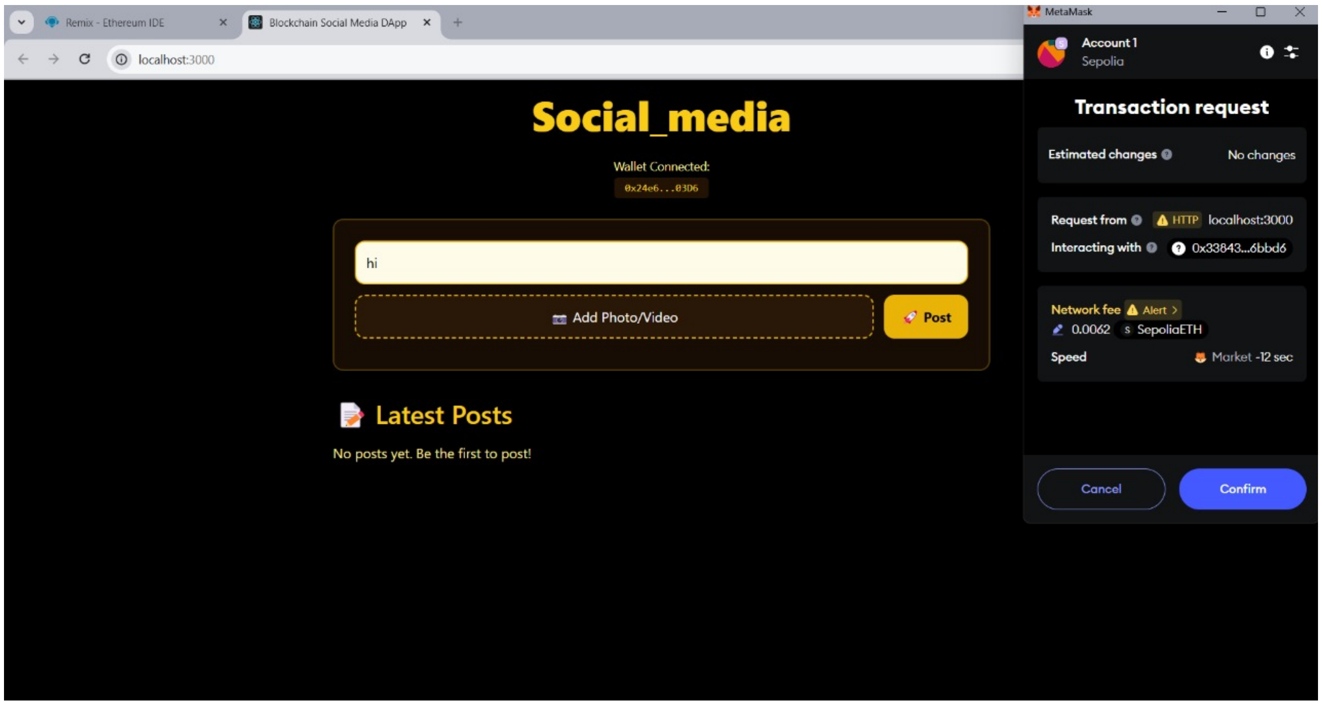
Apr 23, 2025

Contract deployment -0 SepoliaETH

Confirmed -0 SepoliaETH

Create Post -0 SepoliaETH

Confirmed -0 SepoliaETH


Remix - Ethereum IDE

Blockchain Social Media DApp

localhost:3000

Social_media

Wallet Connected:

0x24e6...03D6

hi

📷 Add Photo/Video

🚀 Post

📝 Latest Posts

No posts yet. Be the first to post!

MetaMask

Account 1

Sepolia

Transaction request

Estimated changes: No changes

Request from: HTTP localhost:3000

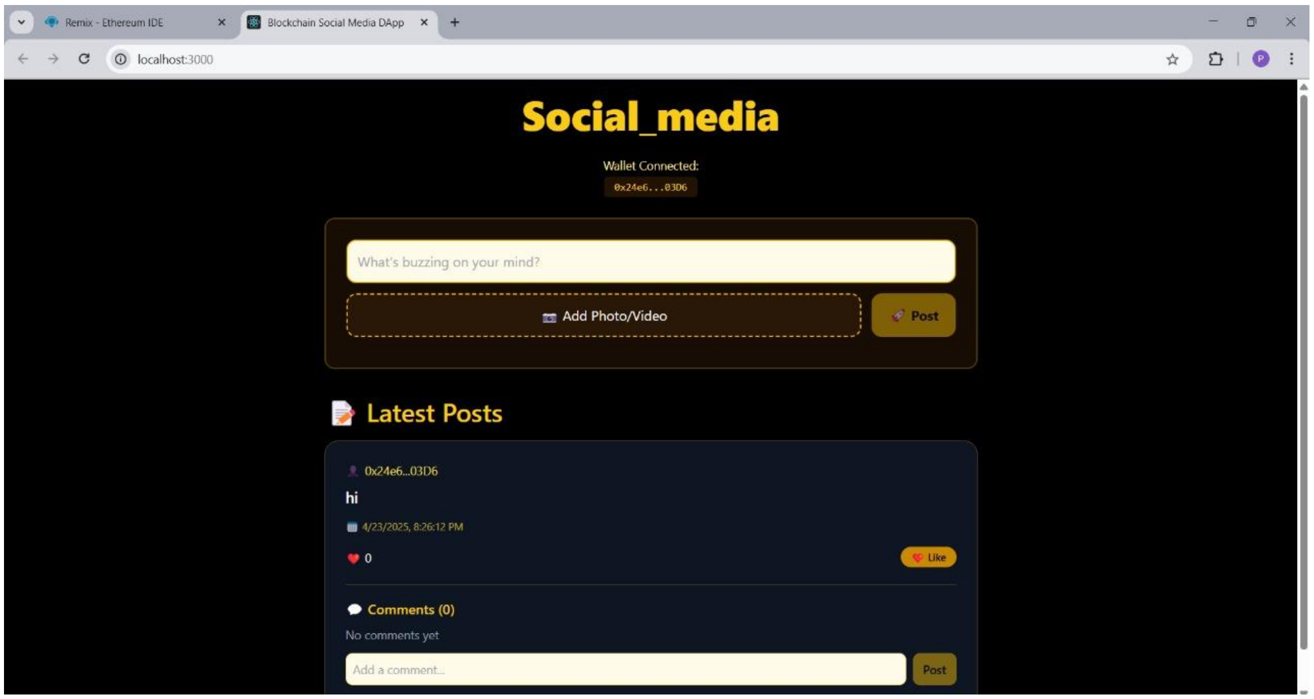
Interacting with: 0x33843...6bbd6

Network fee: Alert

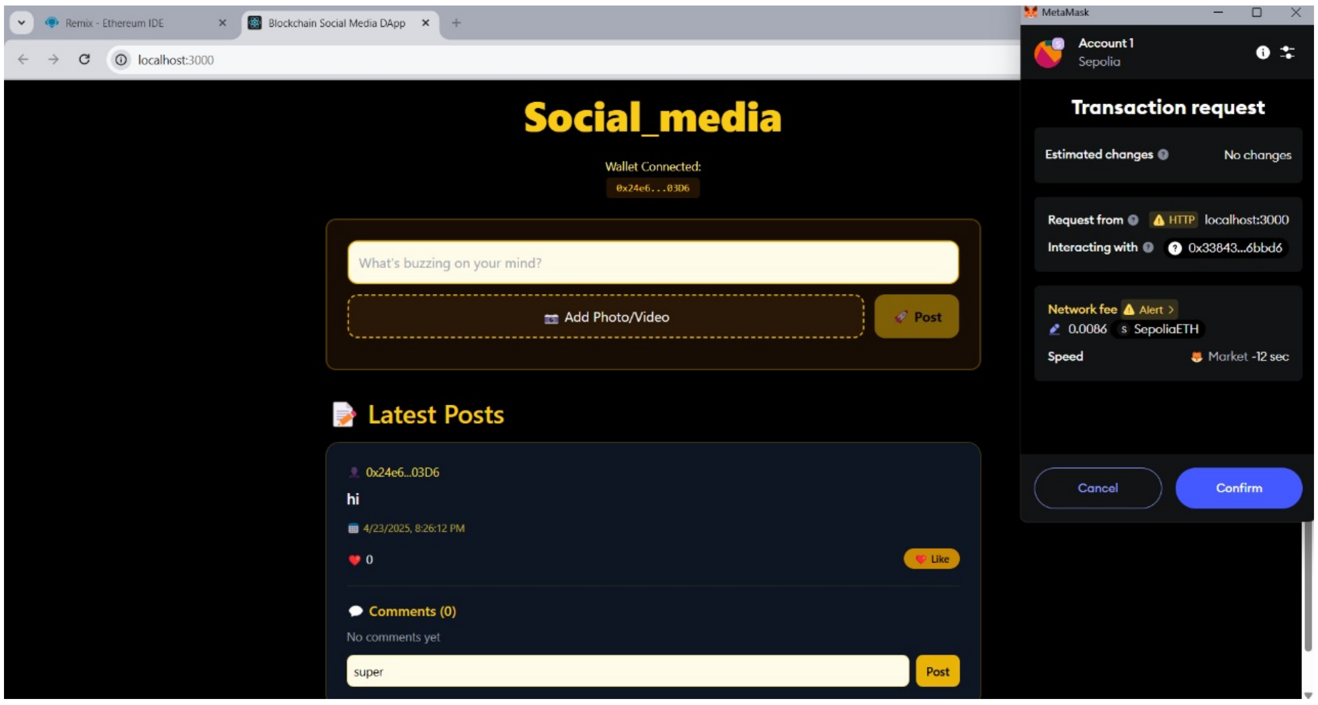
0.0062 SepoliaETH

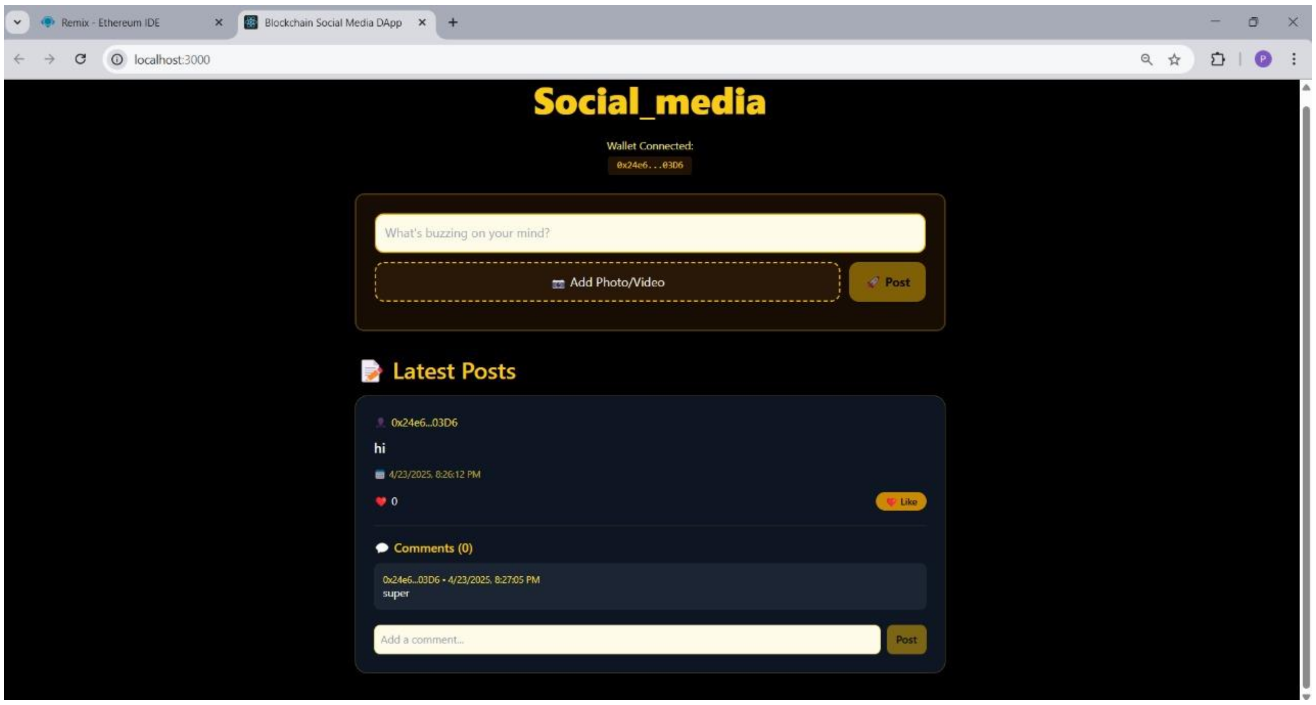
Speed: Market ~12 sec

Cancel Confirm

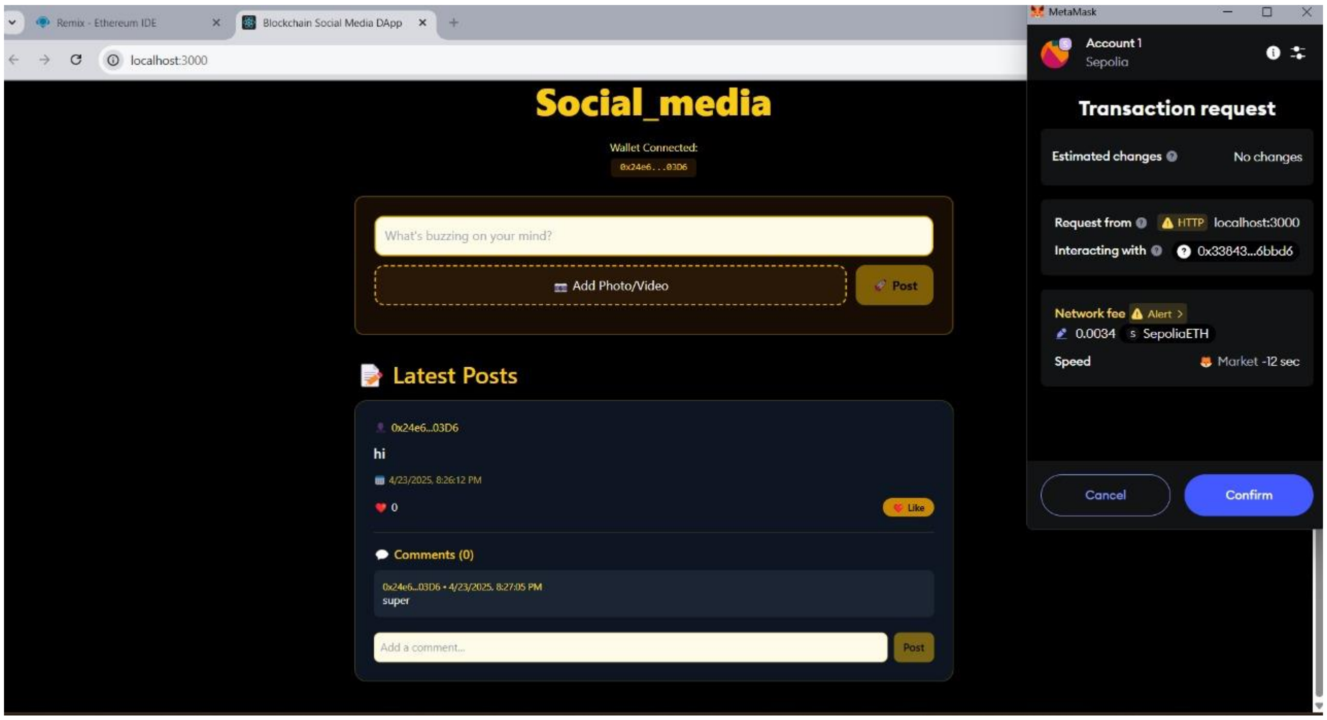


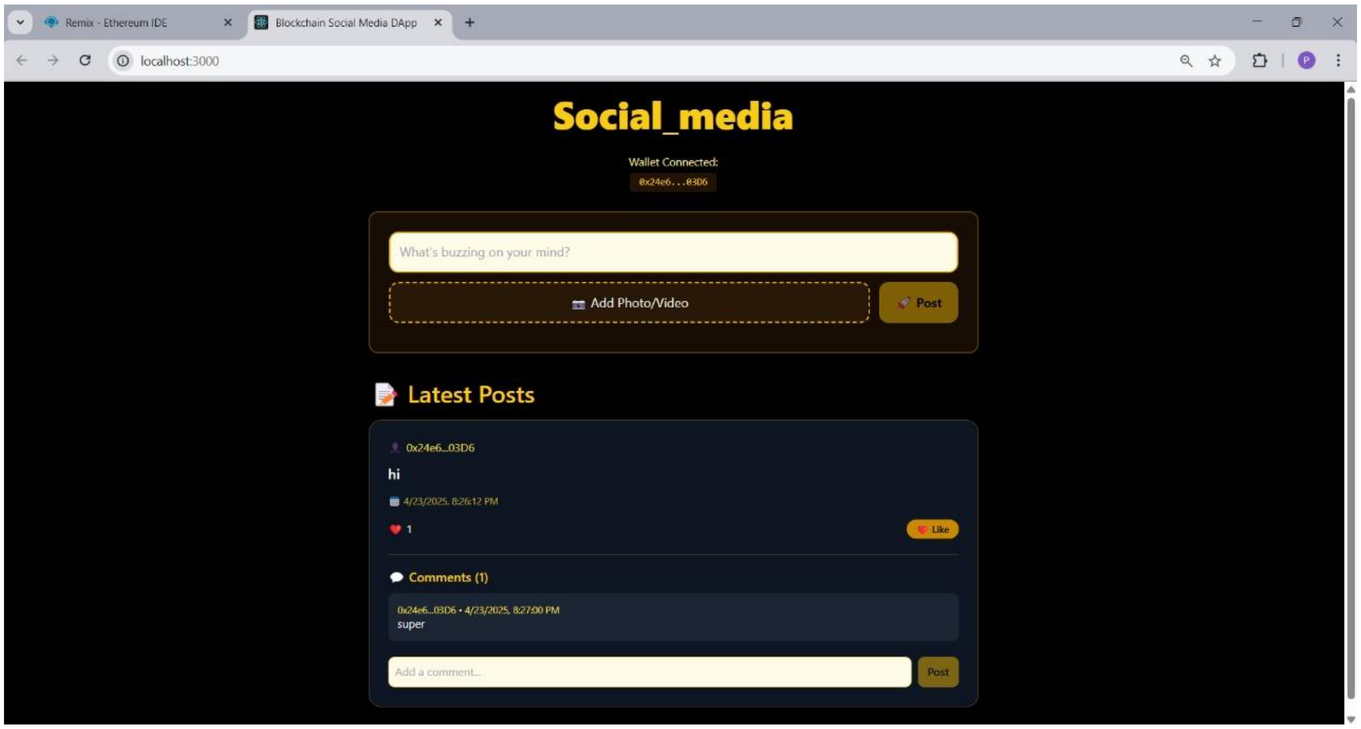
Posting comments



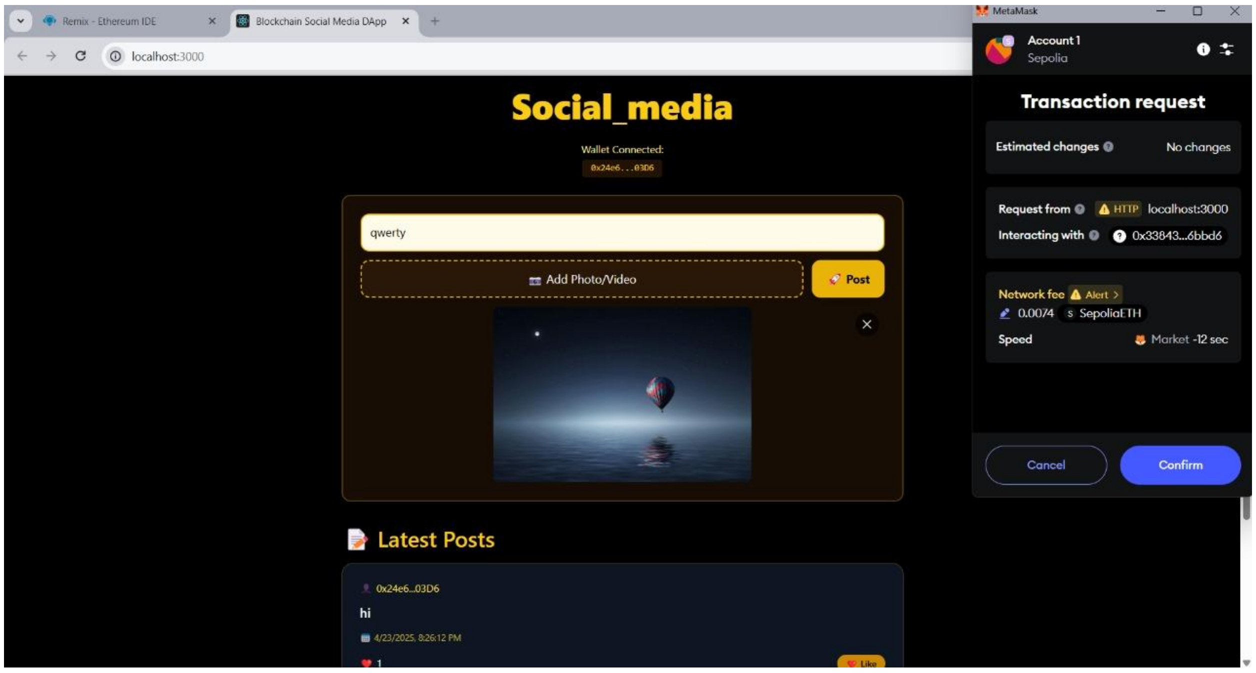


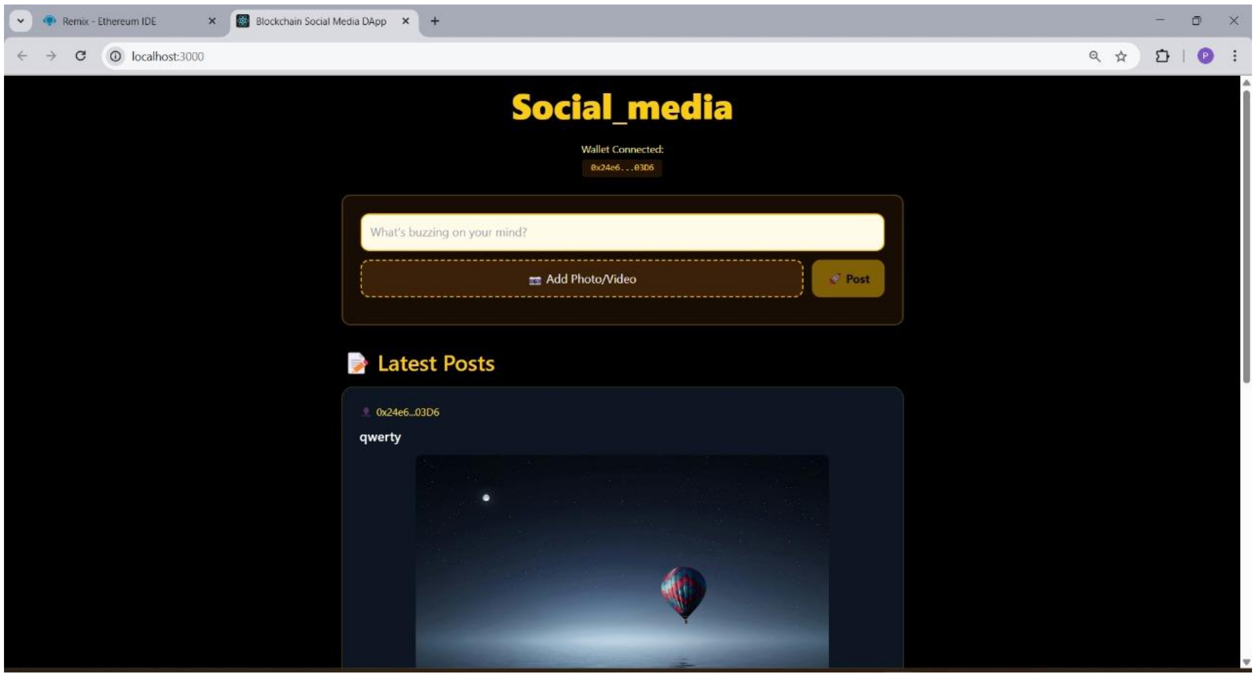
Liking the post made by user

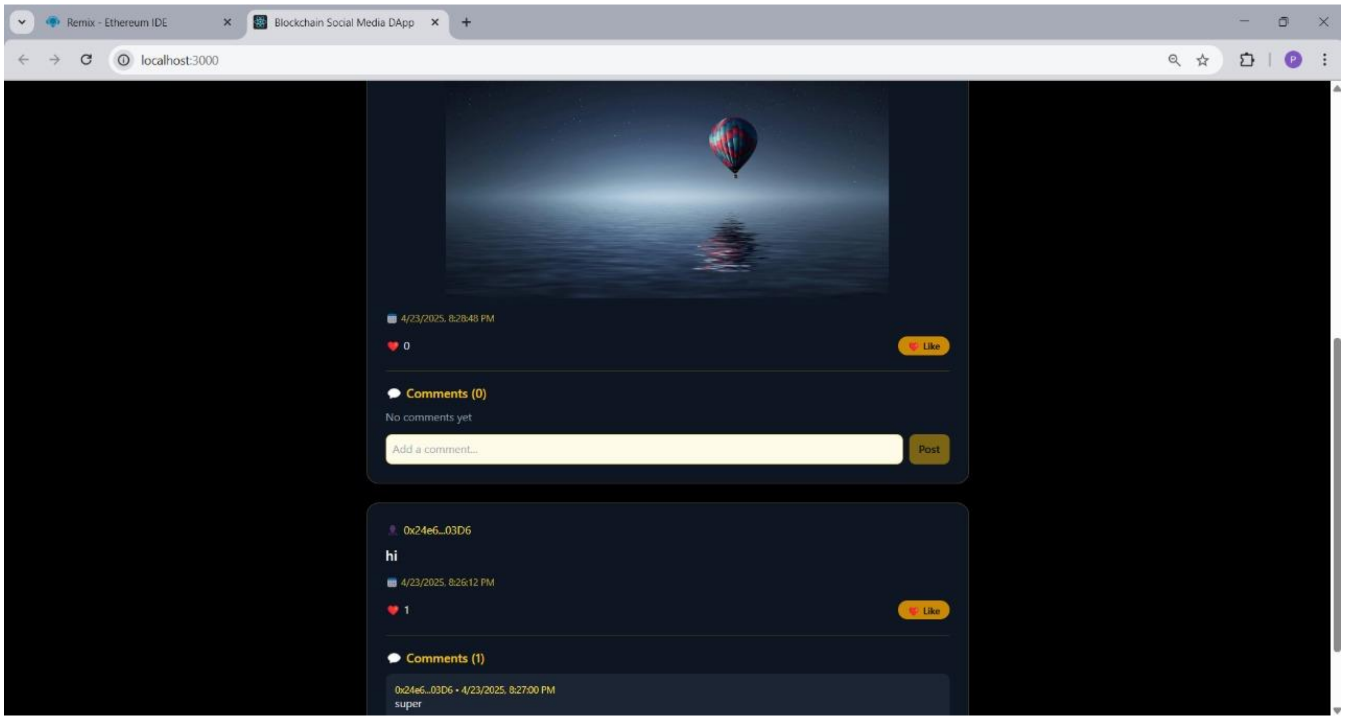




## Posting images




Remix - Ethereum IDE
Blockchain Social Media DApp
localhost:3000
Social_media
Wallet Connected:
0x24e6...03D6
What's buzzing on your mind?
Add Photo/Video
Post
Latest Posts
0x24e6...03D6
qwerty


Remix - Ethereum IDE
Blockchain Social Media DApp
localhost:3000
4/23/2025, 8:28:48 PM
0
Like
Comments (0)
No comments yet
Add a comment...
Post
0x24e6...03D6
hi
4/23/2025, 8:26:12 PM
1
Like
Comments (1)
0x24e6...03D6 • 4/23/2025, 8:27:00 PM
super

## 8. Results

The application successfully demonstrates the viability of decentralized social networking:

- Effortless post and comment posting
- On-chain verification of likes
- Wallet integration controlled by the user
- Frontend updates in real-time through contract events

---

## 9. Advantages and Limitations

**Advantages:**

- Censorship-resistant
- User ownership and privacy
- Transparent and verifiable interactions

**Limitations:**

- Scalability concerns on public blockchains
- Gas fees may hinder frequent interactions
- UI may need enhancements for wider adoption

---

## 10. Conclusion and Future Scope

This project demonstrates the viability of a decentralized social media
platform based on Ethereum and smart
contracts. Future improvements may involve decentralized storage (e.g., IPFS), NFT-based content, and cross-chain compatibility

---

## References

1. Ethereum Documentation: https://ethereum.org/en/developers/
2. MetaMask Documentation: https://docs.metamask.io/
3. Solidity Language Docs: https://docs.soliditylang.org/
4. Ethers.js: https://docs.ethers.org/
5. https://cloud.google.com/application/web3/faucet/ethereum/sepolia

---