**Blockchain Lab Experiment 6**

**Shashwat Tripathi**

**D20A Roll No: 64**

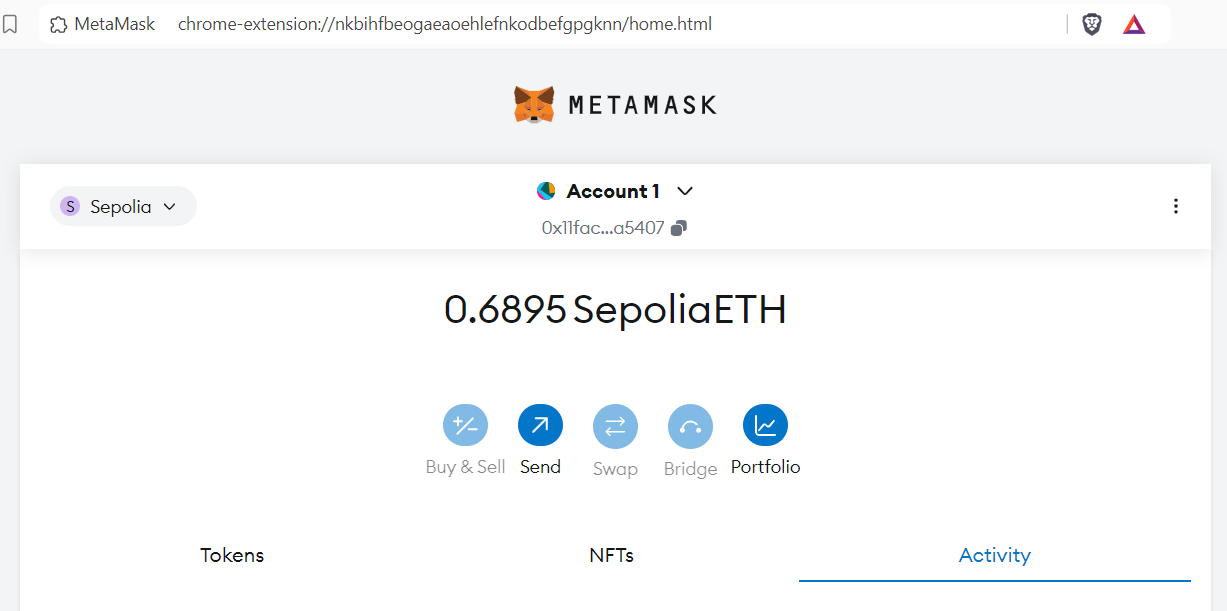
**Aim:** To develop a blockchain-powered web application using Solidity programming language on Remix IDE and embedding Metamask Wallet

**Code:**

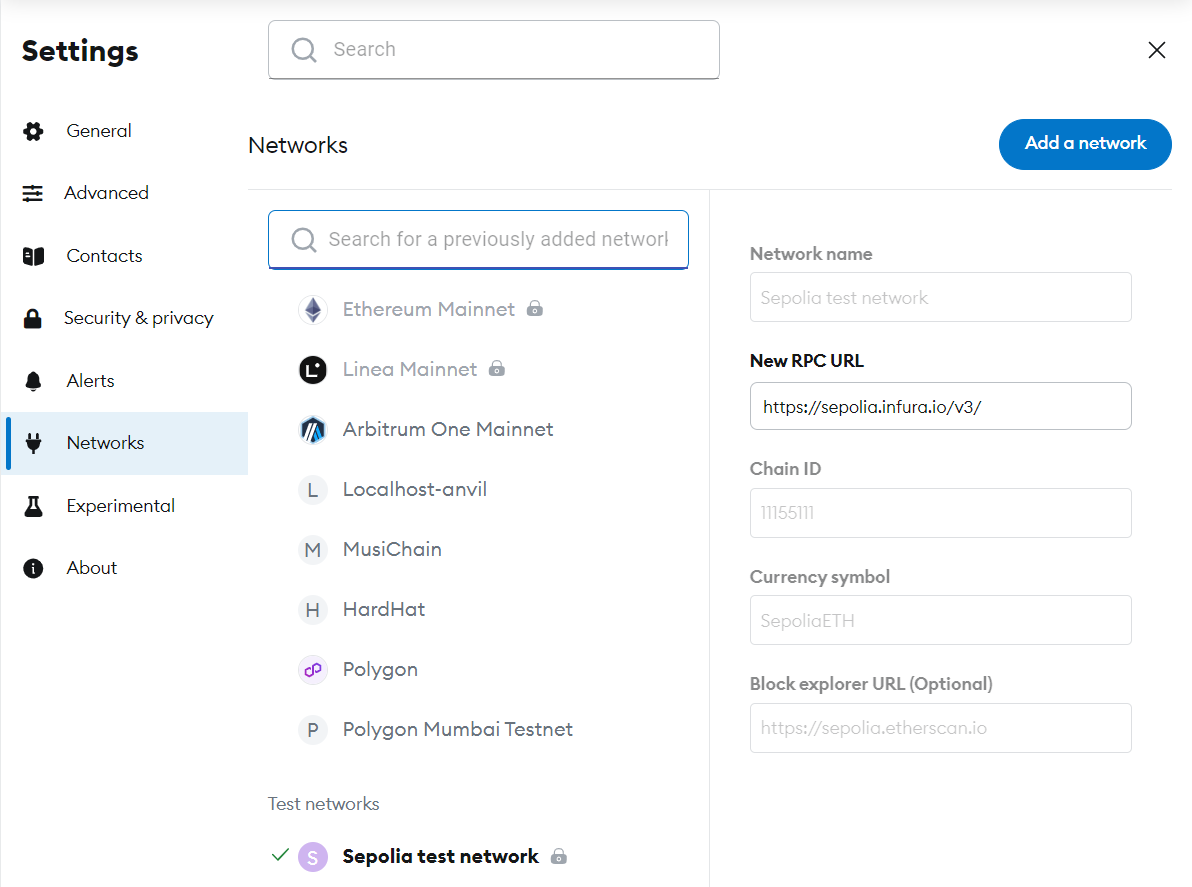
1. **Set Up MetaMask:**

* Install MetaMask
* Create or Import an Account:
* Fund Your Wallet : Sepolia Testnet (0.5 ETH per day) / RSK Testnet (0.05 RBTC per day)

I am using Sepolia TestNet and I have sufficient funds in my account.



1. **Connect the Sepolia Testnet / RSK Testnet to Remix IDE**



1. **Create a Simple Solidity Smart Contract based on the MiniPoject chosen**

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

contract CBDC {

string public name = "Central Bank Digital Currency";

string public symbol = "CBDC";

uint8 public decimals = 18;

uint256 public totalSupply;

mapping(address => uint256) public balanceOf;

mapping(address => mapping(address => uint256)) public allowance;

address public centralBank;

event Transfer(address indexed from, address indexed to, uint256 value);

event Approval(address indexed owner, address indexed spender, uint256 value);

event Mint(address indexed to, uint256 value);

event Burn(address indexed from, uint256 value);

modifier onlyCentralBank() {

require(msg.sender == centralBank, "Only central bank can execute this");

\_;

}

constructor(uint256 \_initialSupply) {

centralBank = msg.sender;

mint(centralBank, \_initialSupply);

}

function mint(address \_to, uint256 \_amount) public onlyCentralBank {

totalSupply += \_amount;

balanceOf[\_to] += \_amount;

emit Mint(\_to, \_amount);

emit Transfer(address(0), \_to, \_amount);

}

function burn(uint256 \_amount) public onlyCentralBank {

require(balanceOf[centralBank] >= \_amount, "Insufficient balance");

balanceOf[centralBank] -= \_amount;

totalSupply -= \_amount;

emit Burn(centralBank, \_amount);

emit Transfer(centralBank, address(0), \_amount);

}

function transfer(address \_to, uint256 \_amount) public returns (bool success) {

require(balanceOf[msg.sender] >= \_amount, "Insufficient balance");

balanceOf[msg.sender] -= \_amount;

balanceOf[\_to] += \_amount;

emit Transfer(msg.sender, \_to, \_amount);

return true;

}

function approve(address \_spender, uint256 \_amount) public returns (bool success) {

allowance[msg.sender][\_spender] = \_amount;

emit Approval(msg.sender, \_spender, \_amount);

return true;

}

function transferFrom(address \_from, address \_to, uint256 \_amount) public returns (bool success) {

require(balanceOf[\_from] >= \_amount, "Insufficient balance");

require(allowance[\_from][msg.sender] >= \_amount, "Allowance exceeded");

balanceOf[\_from] -= \_amount;

balanceOf[\_to] += \_amount;

allowance[\_from][msg.sender] -= \_amount;

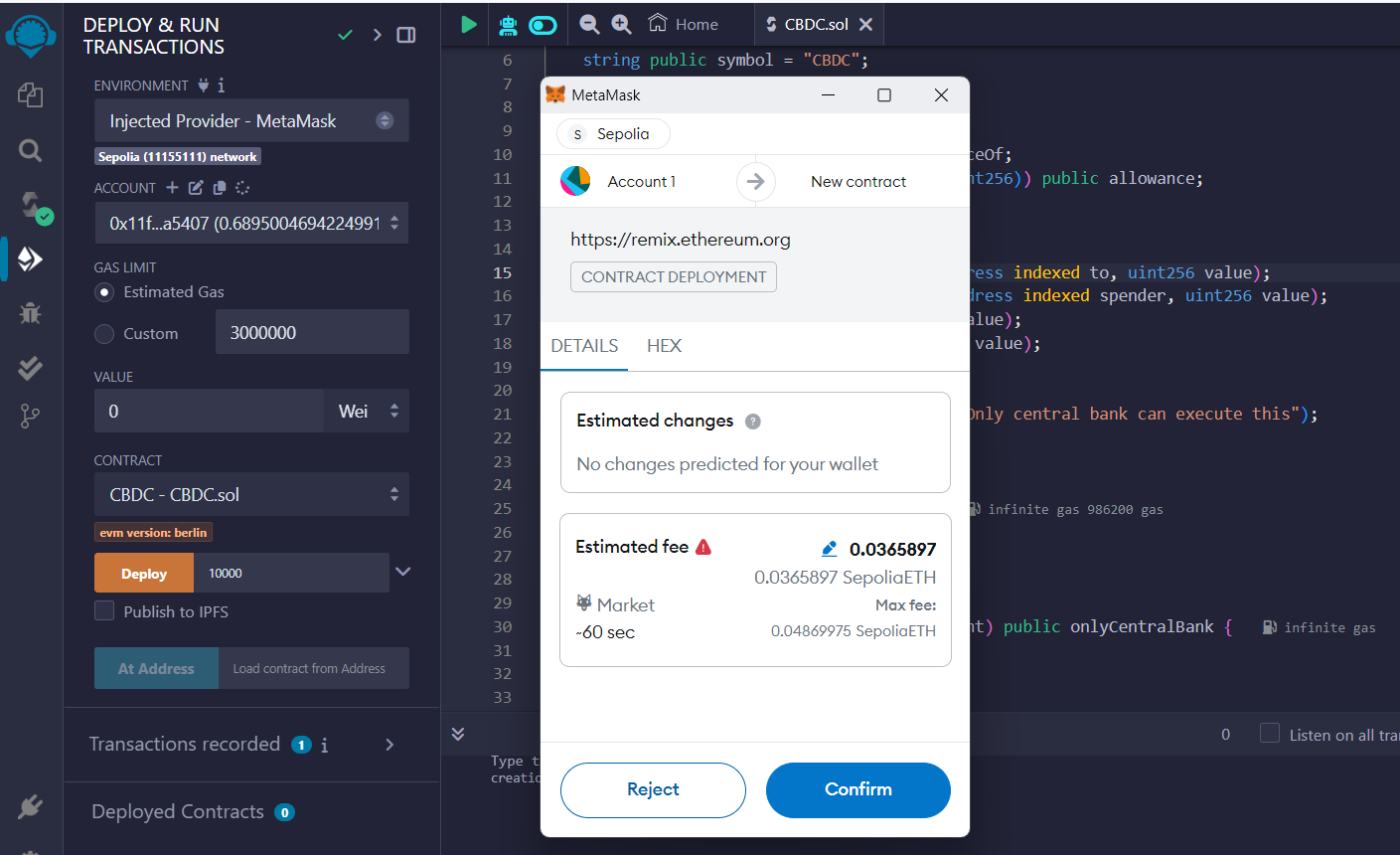
emit Transfer(\_from, \_to, \_amount);

return true;

}

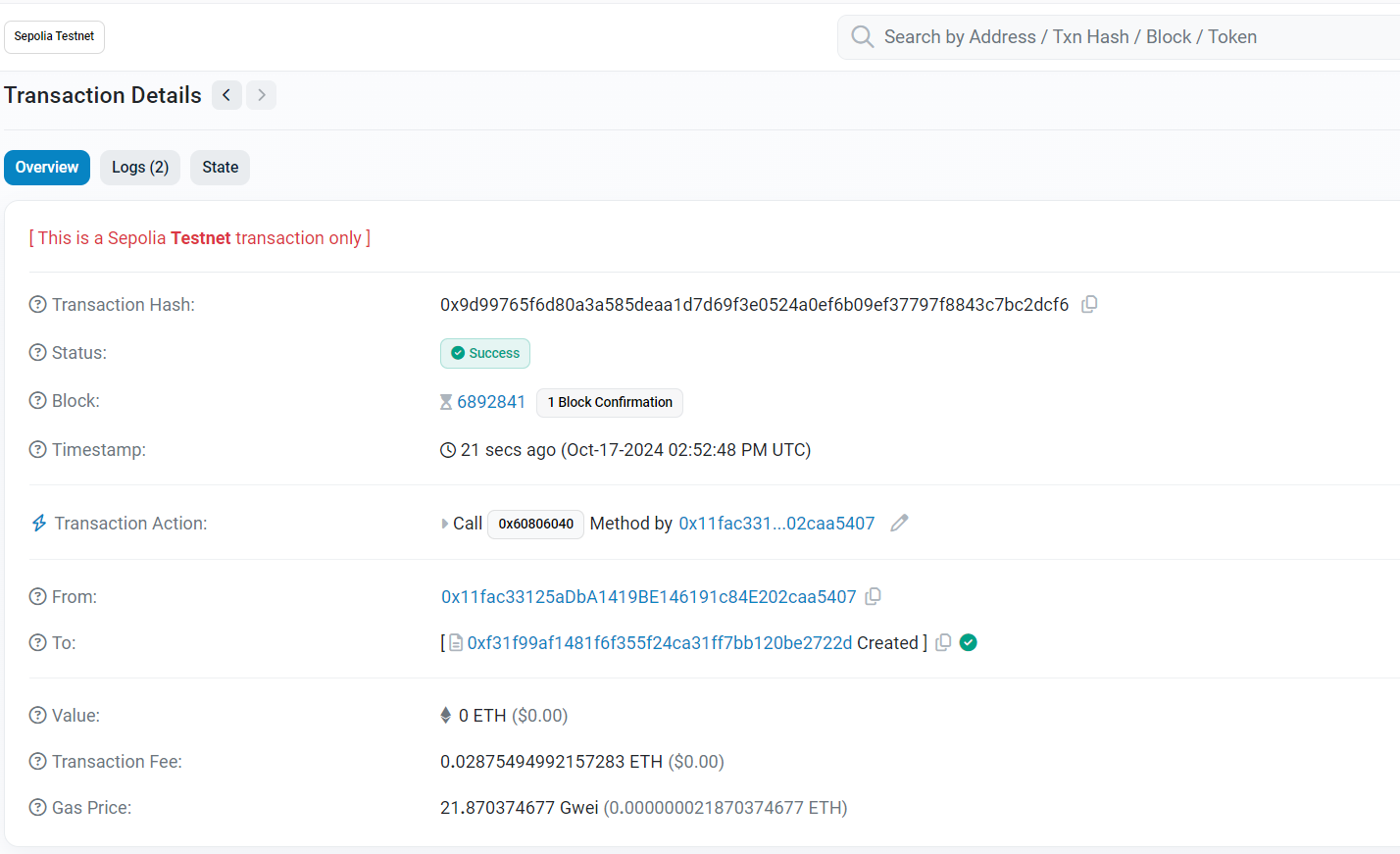
}

1. **Compile and Deploy the Smart Contract.**



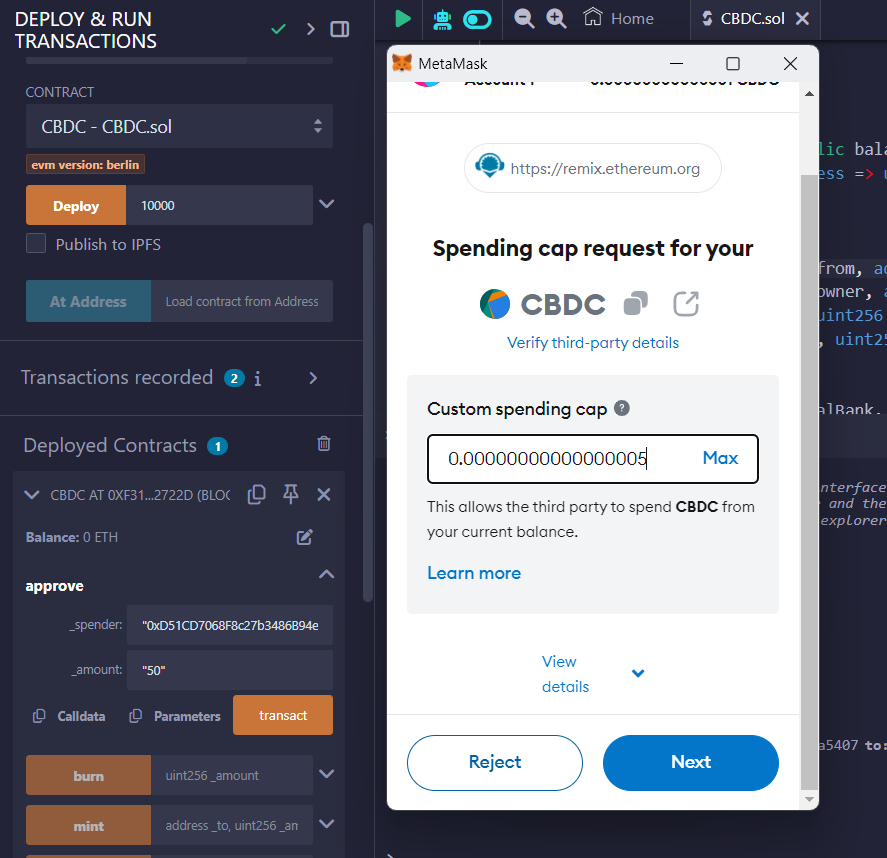
1. **Check the transaction details on the RSK Explorer**

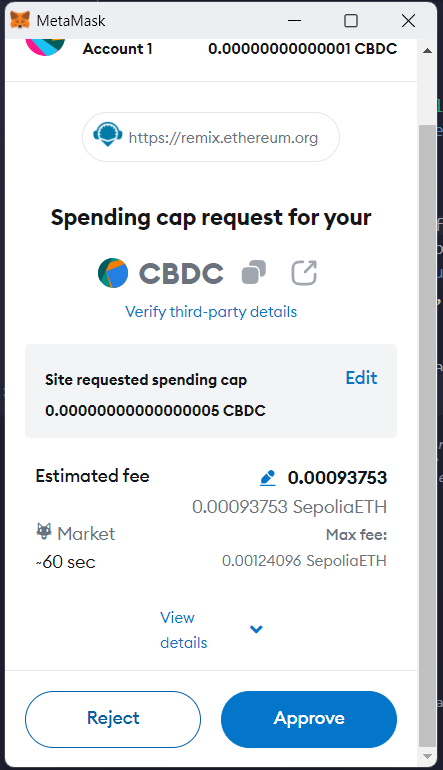
[Txn detail on Etherscan](https://sepolia.etherscan.io/tx/0x9d99765f6d80a3a585deaa1d7d69f3e0524a0ef6b09ef37797f8843c7bc2dcf6)



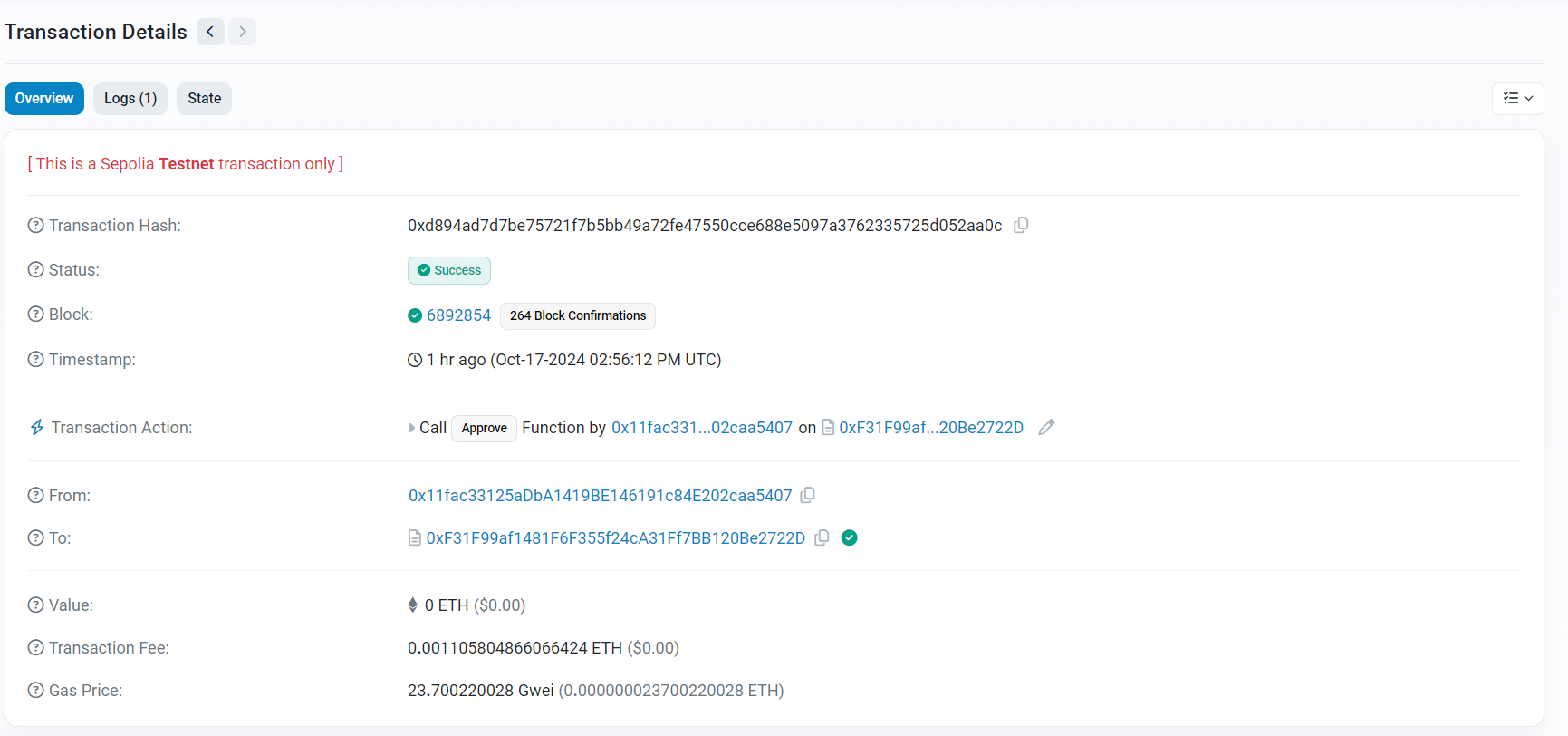
1. **Interact with the smart contract**

Account 1 is allowing account 2 to spend 50 coins on their behalf.





[Txn detail on Etherscan](https://sepolia.etherscan.io/tx/0xd894ad7d7be75721f7b5bb49a72fe47550cce688e5097a3762335725d052aa0c)



**Conclusion:** Thus, we have developed a blockchain-powered web application using Solidity programming language on Remix IDE and embedding Metamask Wallet and verified transactions on etherscan explorer.