**Blockchain Lab Exp 7**

**Shashwat Tripathi**

**D20A RollNo: 64**

**Aim:** To develop a simple crypto exchange and wallet system for users to trade and securely store cryptocurrencies.

**Theory:**

A cryptocurrency exchange and wallet system is a decentralized platform that enables users to trade digital currencies securely while maintaining full control of their assets. Built on blockchain technology, the system uses smart contracts to automate and facilitate peer-to-peer transactions without the need for intermediaries. The wallet system ensures that users' private keys are stored securely, allowing them to access, send, and receive cryptocurrencies. By integrating blockchain with a user-friendly interface and ensuring transaction security through encryption, the exchange and wallet system provides a transparent, immutable, and efficient trading environment.

#### **Key Points:**

* **Blockchain Technology**: Ensures transparency, immutability, and decentralized control.
* **Smart Contracts**: Automate trading transactions securely without intermediaries.
* **Cryptocurrency Wallet**: Safely stores digital assets and private keys for transaction authorization.
* **Decentralization**: Allows peer-to-peer transactions, reducing the risk of third-party control or manipulation.
* **Security**: Enhanced through cryptography and blockchain's inherent trustless environment.

**Code:**

**EthSwap.sol**

pragma solidity ^0.5.0;

import "./Token.sol";

contract EthSwap {

string public name = "EthSwap Instant Exchange";

Token public token;

uint public rate = 100;

event TokensPurchased(

address account,

address token,

uint amount,

uint rate

);

event TokensSold(

address account,

address token,

uint amount,

uint rate

);

constructor(Token \_token) public {

token = \_token;

}

function buyTokens() public payable {

// Calculate the number of tokens to buy

uint tokenAmount = msg.value \* rate;

// Require that EthSwap has enough tokens

require(token.balanceOf(address(this)) >= tokenAmount);

// Transfer tokens to the user

token.transfer(msg.sender, tokenAmount);

// Emit an event

emit TokensPurchased(msg.sender, address(token), tokenAmount, rate);

}

function sellTokens(uint \_amount) public {

// User can't sell more tokens than they have

require(token.balanceOf(msg.sender) >= \_amount);

// Calculate the amount of Ether to redeem

uint etherAmount = \_amount / rate;

// Require that EthSwap has enough Ether

require(address(this).balance >= etherAmount);

// Perform sale

token.transferFrom(msg.sender, address(this), \_amount);

msg.sender.transfer(etherAmount);

// Emit an event

emit TokensSold(msg.sender, address(token), \_amount, rate);

}

}

**Token.sol**

pragma solidity ^0.5.0;

contract Token {

string public name = "DApp Token";

string public symbol = "DAPP";

uint256 public totalSupply = 1000000000000000000000000; // 1 million tokens

uint8 public decimals = 18;

event Transfer(

address indexed \_from,

address indexed \_to,

uint256 \_value

);

event Approval(

address indexed \_owner,

address indexed \_spender,

uint256 \_value

);

mapping(address => uint256) public balanceOf;

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

constructor() public {

balanceOf[msg.sender] = totalSupply;

}

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

require(balanceOf[msg.sender] >= \_value);

balanceOf[msg.sender] -= \_value;

balanceOf[\_to] += \_value;

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

return true;

}

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

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

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

return true;

}

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

require(\_value <= balanceOf[\_from]);

require(\_value <= allowance[\_from][msg.sender]);

balanceOf[\_from] -= \_value;

balanceOf[\_to] += \_value;

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

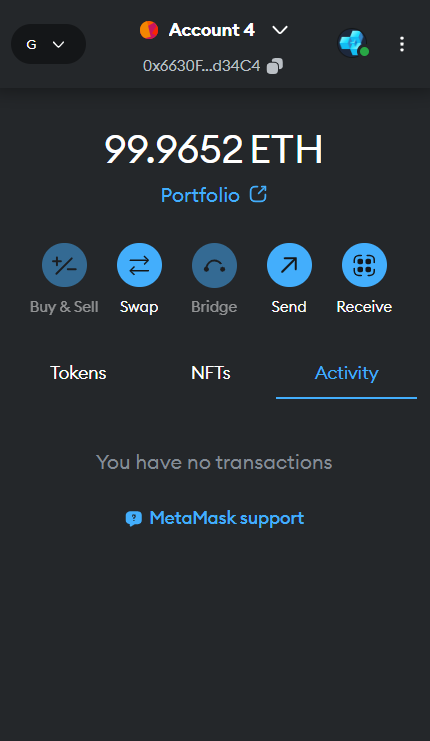
emit Transfer(\_from, \_to, \_value);

return true;

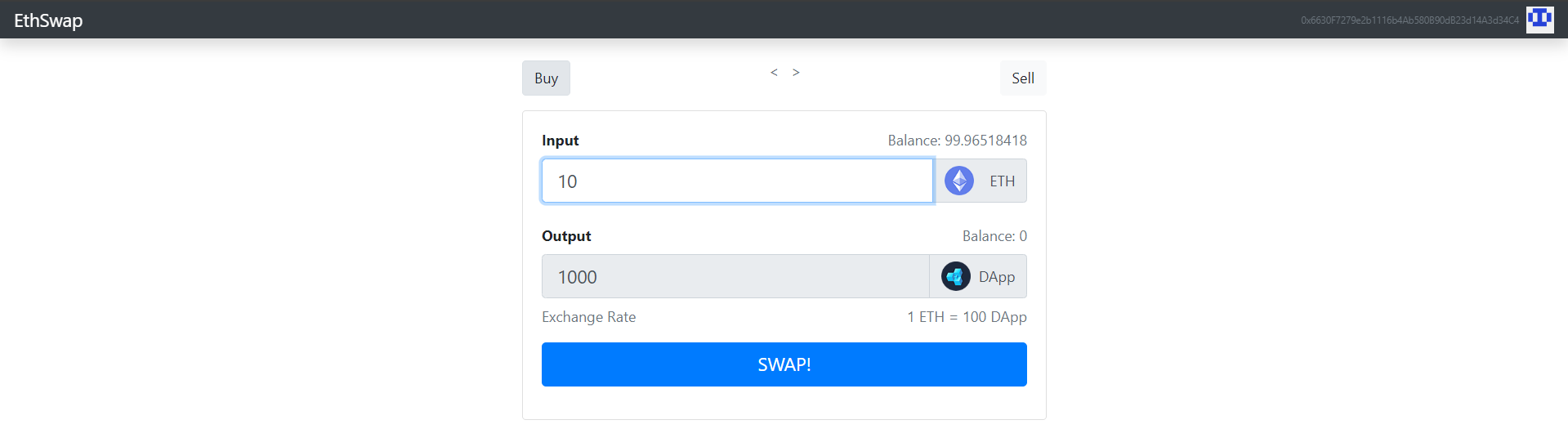
}

}

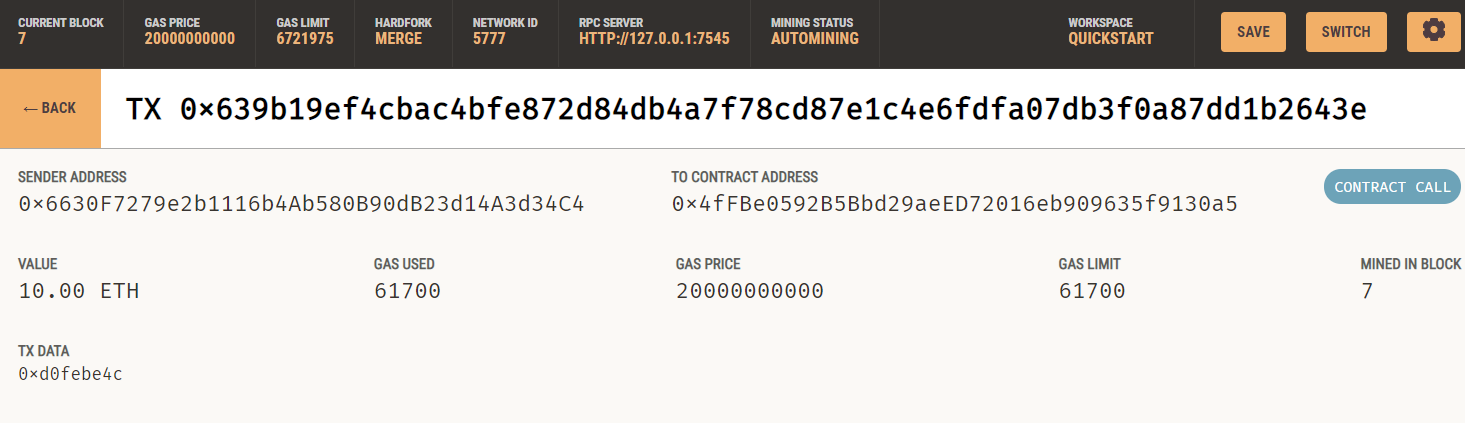
**Output:**



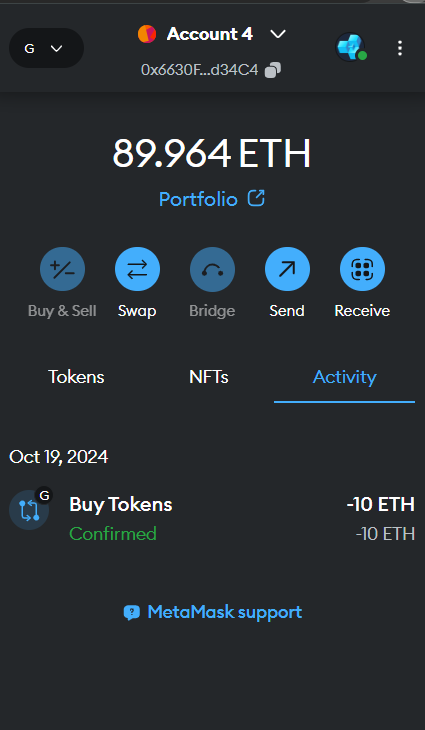
Account balance before any transaction



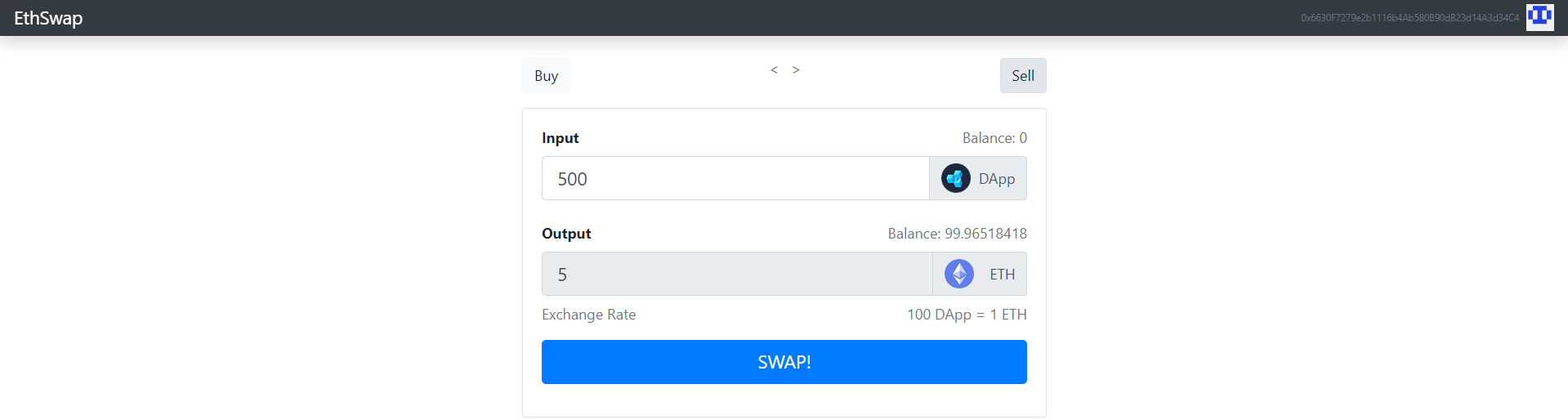
Buying 1000 DApp tokens for 10 ethers

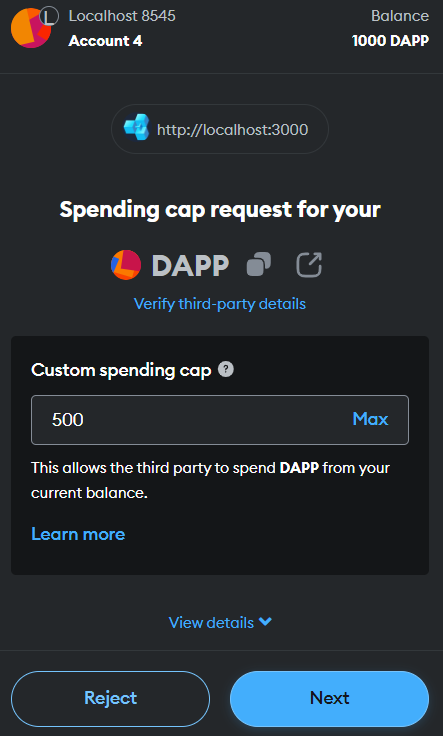
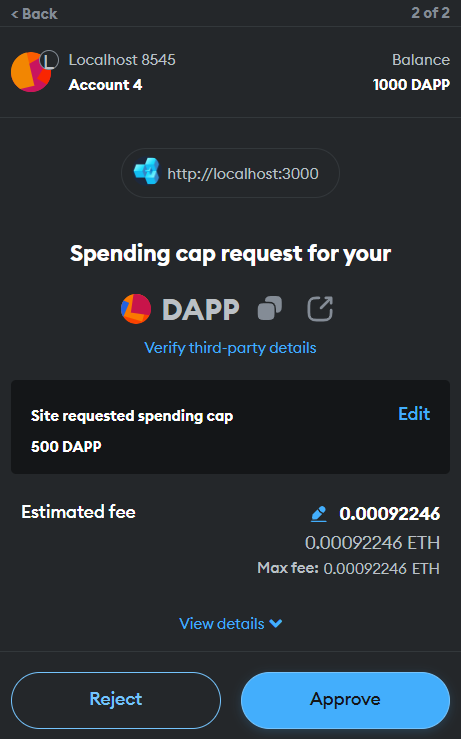


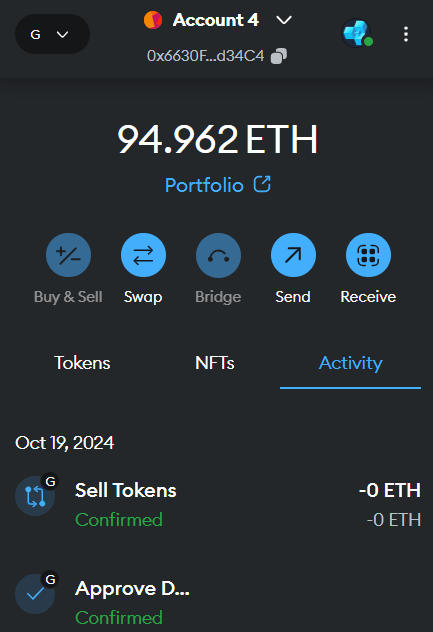
Transaction recorded on ganache



Updated wallet balance after purchasing DApp tokens

  
Spending 500 DApp tokens to get 5 Ethers



Account balance increased after selling 500 DApp tokens to get 5 Ethers

**Conclusion:** In conclusion, this experiment successfully demonstrates the development of a secure and efficient crypto exchange and wallet system that enables users to trade and manage cryptocurrencies while ensuring the safety of their assets through blockchain technology.