DECENTRALIZED MARKETPLACE SMART CONTRACT

Empowering Decentralized Buying and Selling

SUBMITTED BY:
CHINMAYEE SHARMA

Table of Contents

Overview of Problem Statement

Challenges in Traditional Marketplaces

- **Centralization Issues**:

Traditional marketplaces are often centralized, leading to a lack of transparency and increased vulnerability to fraud.

- **Intermediaries**:

The involvement of intermediaries may result in higher fees and slower transaction processes.

- **Limited Security**:

Security concerns arise due to centralized control, making data and transactions susceptible to hacks.

The Need for a Decentralized Solution

- **Reduced Fees**:

With no middlemen, users can transact directly, saving money on commissions.

- **Transparency**:

All transactions and agreements are recorded on the blockchain, making them transparent and tamper-proof.

- **Censorship Resistance**:

Users have full control over their transactions, reducing the risk of censorship.

- **Global Access**:

Anyone with an internet connection can participate in the marketplace.

Objectives

Clear Objectives of the Smart Contract

- **Decentralization**: Enable decentralized buying and selling to eliminate the need for centralized control.
- **Transparency:** Provide transparency in transactions by leveraging blockchain technology.
- **Security**: Enhance security by securing ownership and transaction details on the blockchain.
- **Efficiency**: Automate and streamline transactions through the use of smart contracts.
- **User Empowerment**: Empower users by reducing reliance on intermediaries and providing direct control over transactions.

Addressing Key Requirements.

- **Seller Listings**: Allow sellers to list items with names and prices.
- **Buyer Transactions**: Enable buyers to purchase items using Ether equal to the listed price.
- **Secure Ether Transfer**: Ensure secure transfer of Ether from buyers to sellers.
- **Prevent Further Sales**: Implement functionality to mark items as sold upon purchase, preventing further sales.

Enhancing Transparency and Security in Transactions

- **Smart Contracts**: Utilize smart contracts for automated and secure transactions.
- **Blockchain Technology**: Leverage blockchain to create a tamper-resistant and transparent ledger.

Home | Decentralised Marketplace.sol

```
/*
Problem Statement 1: Decentralized Marketplace

Objective: Develop a Decentralized Marketplace contract where users can list items for sale and others
can purchase them using Ether.

Sellers should list items with a name and price.
Buyers can purchase items by sending Ether equal to the listed price.
Ensure that the Ether sent by buyers is transferred to the seller.
Implement functionality to mark items as sold upon purchase to prevent further sales.
*/

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

contract DecentralizedMarketplace{

    struct Product{
        uint256 productId;
        string name;
        uint256 price;
        address seller;
        bool sold;
        string details;
    }

    address public owner;

    //mapping productId with Product
    mapping(uint256=>Product) public products;

    //total number of products in a list
    uint256 public productCount;

    // Mapping to keep track of authorized sellers
    mapping(address => bool) public authorizedSellers;

    modifier authOwner() {
        require(msg.sender == owner, "You are not the owner");
        _;
    }

    modifier isAuthorizedSeller() {
        require(authorizedSellers[msg.sender], "Real ID se aao seller bhai");
        _;
    }

    modifier productAvailable(uint256 productId) {
        require(productId < productCount, "Product is Out of Stock");
        _;
    }

    modifier notSold(uint256 productId) {
        require(!products[productId].sold, "Sold out");
        _;
    }
```

Home | Decentralised Marketplace.sol

```
    constructor() {
        owner=msg.sender;
        productCount=0; //initially no. of products set to 0
    }

    // Allowing the owner to authorize sellers
    function authorizeSeller(address _seller) external authOwner {
        authorizedSellers[_seller] = true;
    }

    // Allowing authorized sellers to add products to the list
    function addProduct(string memory _name, uint256 _price, string memory _details) external isAuthorizedSeller{

        products[productCount] = Product({
            productId: productCount,
            name: _name,
            price: _price,
            seller: msg.sender,
            sold: false,
            details: _details
        });

        productCount++;
    }


    //Buyers can purchase items by sending Ether equal to the listed price.
    function buyProduct(uint256 productId) external payable productAvailable(productId) notSold(productId) {

        Product storage product = products[productId];

        //the ether send by buyer is sufficient or not
        require(msg.value == product.price, "Pure pese de bhai");

        // Ensuring that the Ether sent by buyers is transferred to the seller.
        payable(product.seller).transfer(msg.value);
        product.sold = true;
    }

    // Implement functionality to mark items as sold upon purchase to prevent further sales.
    function isProductSold(uint256 productId) external view returns (bool) {
        return products[productId].sold;
    }

    // New function to get the product ID by providing the name
    function getProductIdByName(string memory productName) external view returns (uint256) {
        for (uint256 i = 0; i < productCount; i++) {
            if (keccak256(abi.encodePacked(products[i].name)) == keccak256(abi.encodePacked(productName))) {
                return i;
            }
        }
        revert("Product not found");
    }
}
```

Solution Overview

- **Decentralized Marketplace Contract**:

Empower users to buy and sell products without central authority.

- **Key Components:**

1. **Product Struct:** Defines the structure for product data, including name, price, seller, and transaction status.
2. **Functions:**

✓ **authorizeSeller:** Owner authorizes sellers

✓ **addProduct:** Sellers add products

✓ **buyProduct:** Buyers purchase products

✓ **isProductSold:** Checks product status

✓ **getProductIdByName:** Retrieves product ID by name

3. Modifiers:

❑ **authOwner:** Ensures owner authentication

❑ **isAuthorizedSeller:** Ensures authorized seller authentication

❑ **productAvailable:** Ensures product availability

❑ **notSold:** Ensures the product is not sold

- **Secure and Transparent Transactions**

Utilises blockchain technology and smart contracts for enhanced security and transparency.

- **Objective**

Provide a decentralized and secure platform for transparent buying and selling.

Testing of the decentralized marketplace contract

Step 1: Deploy the Contract

1. Open Remix at https://remix.ethereum.org/.

2. Paste the modified Solidity code into the editor.

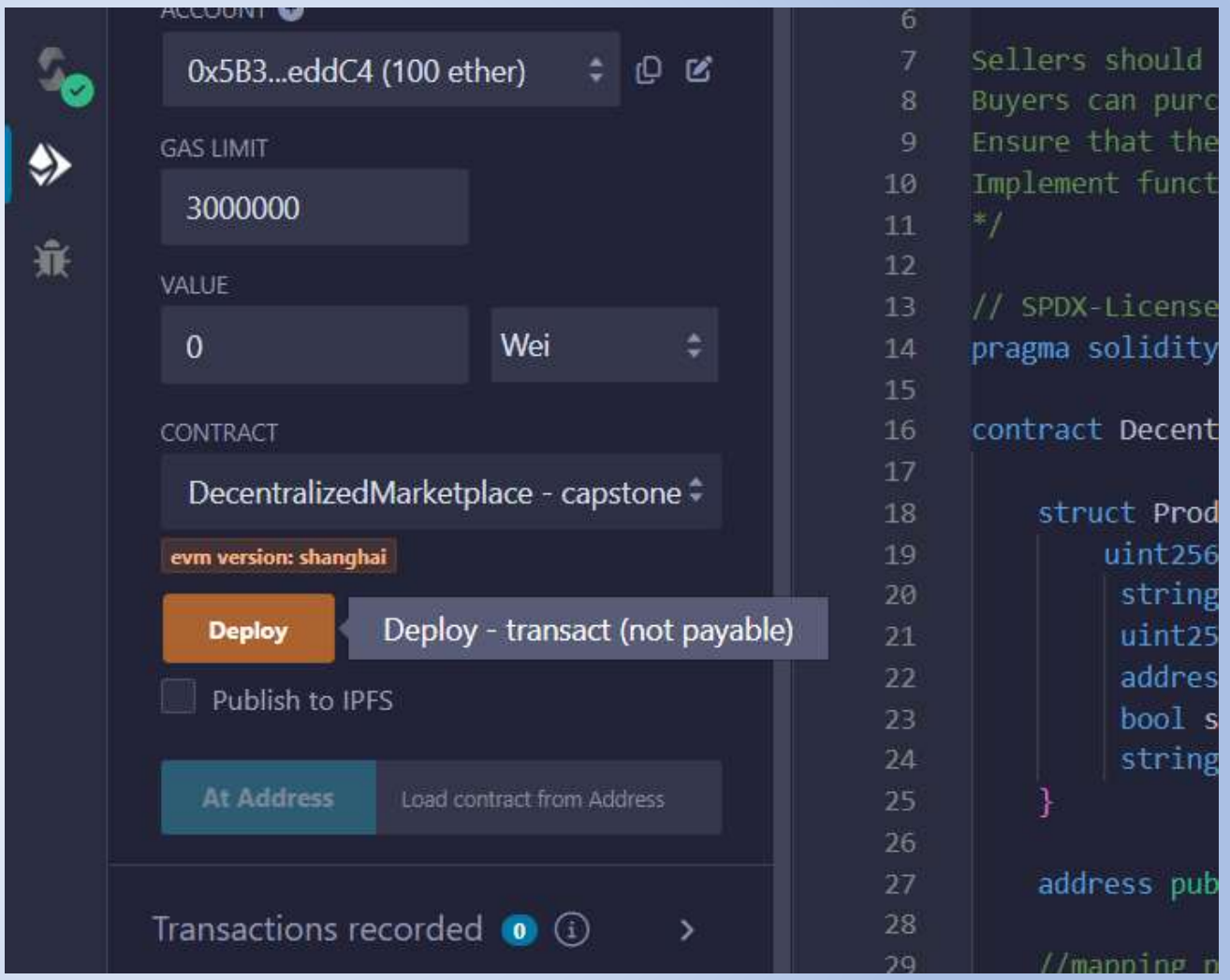
3. Choose a Solidity compiler version (e.g., 0.8.0) in the "Solidity" tab.

4. Click the "Compile" button to compile the contract.

5. Switch to the "Deploy & Run Transactions" tab.

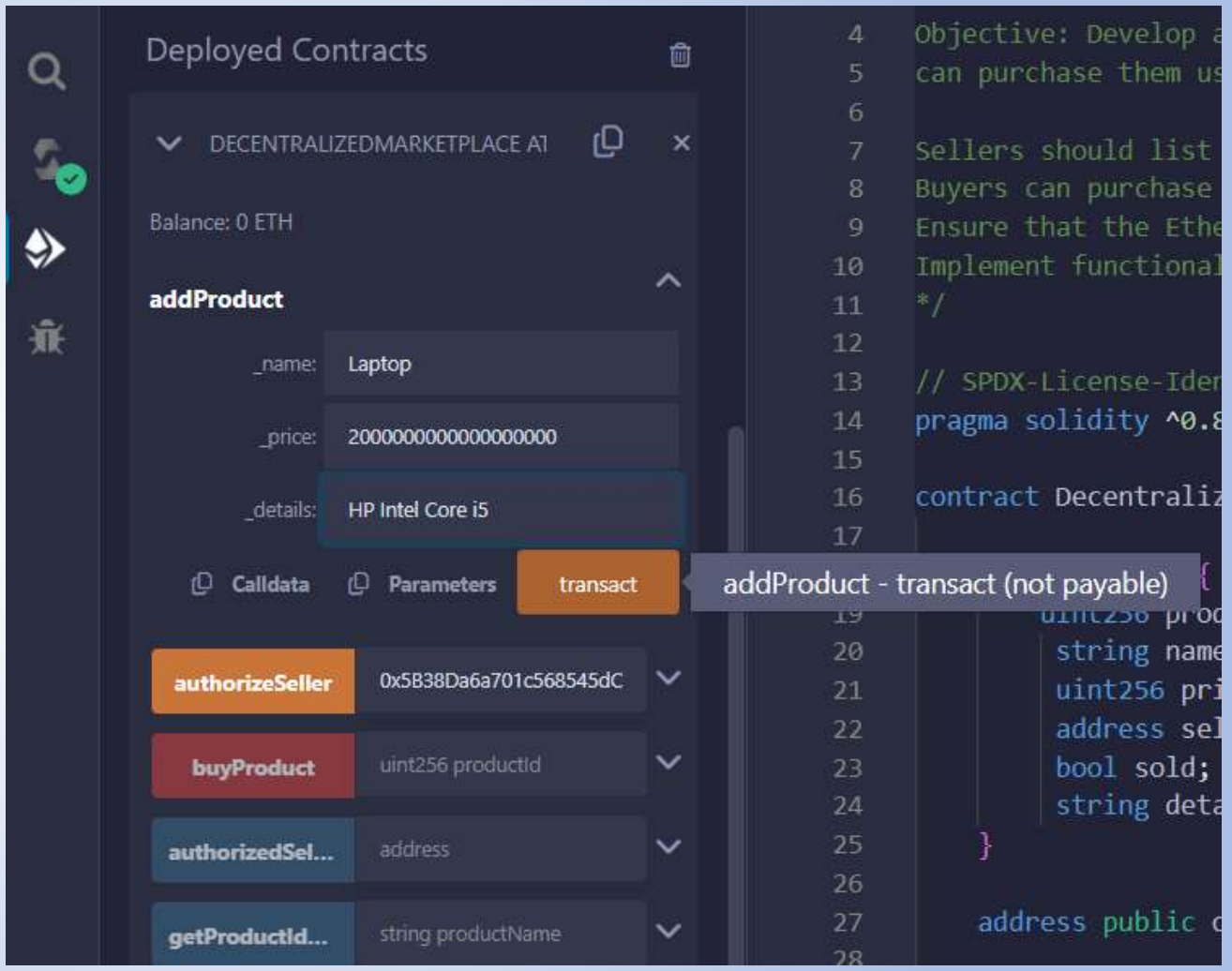
6. In the "Environment" section, select “Remix VM Shanghai" as the environment.

7.Click the "Deploy" button to deploy the contract.



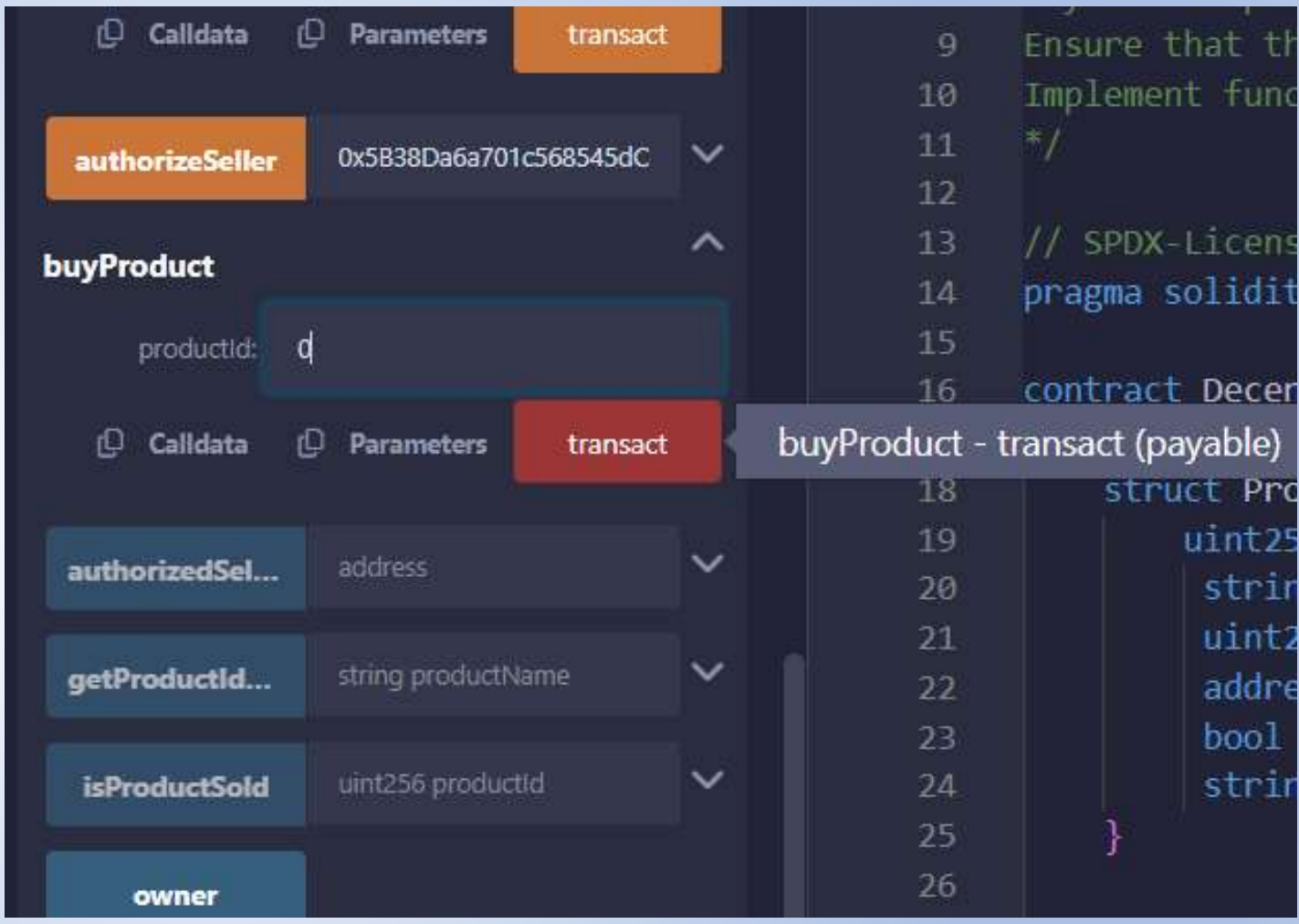
Step 2: Authorize Sellers (Owner's Action) and Add Products

1. Use the **authorizeSeller** function to authorize some addresses as sellers.
2. Use the **addProduct** function to add the products in the marketplace.
3. Example: Adding a product with the name Laptop, price 2 ether, and details “HP Intel Core i5”.
4. Add more product with different details.



Step 3: Buy Products

1. Use the **buyProduct** function to simulate the buyers purchasing products.
2. Example: buying the product with **productId** 0 by sending 2 ether.
3. Check if the product is marked as sold by calling the **isProductSold** function.



CONCLUSION

Summary of the Smart Contract's Capabilities

- **Decentralization:** Provides a decentralized platform for buying and selling.
- **Transparency:** Utilizes blockchain for transparent and tamper-resistant transactions.
- **Security:** Enhances security through smart contracts and secure ownership.
- **Efficiency:** Automates transactions, eliminating the need for intermediaries.
- **User Empowerment:** Direct control over transactions, reducing reliance on third parties.

Contributions to Decentralized Marketplaces

- **Elimination of Intermediaries**: Reduces fees and accelerates transaction processes.
- **Secure Transactions**: Ensures secure ownership and reduces fraud risk.
- **Efficient Marketplace**: Streamlines buying and selling processes.

Benefits of Transparency and Security

- **Tamper-Resistant Transactions**: Transactions recorded on the blockchain are resistant to tampering.
- **Trust and Confidence**: Builds trust and confidence among users through transparent processes.
- **Future of Decentralized Commerce**: Paves the way for the future of decentralized commerce.