Trade Finance

Tradefinance.sol

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

contract TradeFinance {
    enum Status { Issued, Approved, Completed }

    struct LetterOfCredit {
        string lcId;
        string buyer;
        string seller;
        uint amount;
        Status status;
    }

    mapping(string => LetterOfCredit) private locRecords;
    address public bank;

    constructor() {
        bank = msg.sender; // Bank deploys the contract
    }

    // Issue a new Letter of Credit
    function issueLoC(string memory _lcId, string memory _buyer, string memory _seller,
uint _amount) public {
        require(bytes(locRecords[_lcId].lcId).length == 0, "LoC ID already exists");
        locRecords[_lcId] = LetterOfCredit(_lcId, _buyer, _seller, _amount,
Status.Issued);
    }

    // Approve LoC (only bank can approve)
    function approveLoC(string memory _lcId) public {
        require(msg.sender == bank, "Only the bank can approve LoC");
        require(locRecords[_lcId].status == Status.Issued, "LoC must be in Issued
status");
        locRecords[_lcId].status = Status.Approved;
    }

    // Mark transaction as completed
    function completeTransaction(string memory _lcId) public {
        require(locRecords[_lcId].status == Status.Approved, "LoC must be Approved
first");
        locRecords[_lcId].status = Status.Completed;
    }

    // Get LoC details
    function getLoC(string memory _lcId) public view returns (string memory, string
memory, uint, Status) {
```

```
        LetterOfCredit memory loc = locRecords[_lcId];
        return (loc.buyer, loc.seller, loc.amount, loc.status);
    }
}
```

Tradefinance.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Trade Finance</title>
    <script src="https://cdn.jsdelivr.net/npm/web3/dist/web3.min.js"></script>
    <style>
        body {
            font-family: Arial, sans-serif;
            text-align: center;
            background-color: #f4f4f4;
            margin: 0;
            padding: 0;
        }
        .container {
            width: 90%;
            max-width: 500px;
            margin: 50px auto;
            background: white;
            padding: 20px;
            border-radius: 10px;
            box-shadow: 0px 0px 10px rgba(0, 0, 0, 0.1);
        }
        h2 {
            color: #333;
        }
        input {
            width: 90%;
            padding: 10px;
            margin: 10px 0;
            border: 1px solid #ccc;
            border-radius: 5px;
            font-size: 16px;
        }
        button {
            width: 95%;
            padding: 12px;
            margin: 10px 0;
            border: none;
            border-radius: 5px;
            background-color: #007bff;
            color: white;
            font-size: 16px;
            cursor: pointer;
```

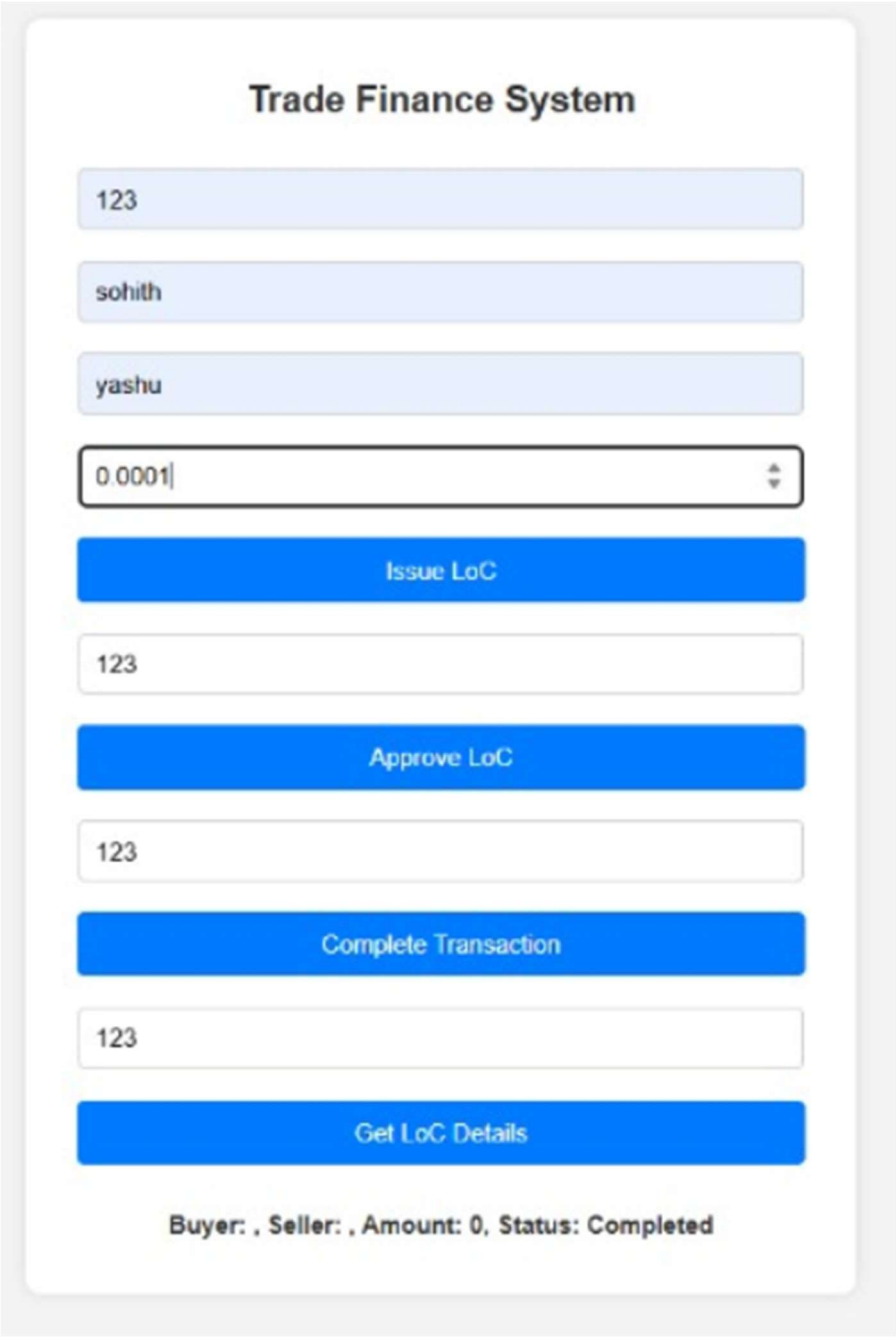
```
            transition: 0.3s;
        }
        button:hover {
            background-color: #0056b3;
        }
        #output {
            margin-top: 20px;
            font-size: 16px;
            font-weight: bold;
            color: #333;
        }
    </style>
</head>
<body>

    <div class="container">
        <h2>Trade Finance System</h2>

        <input type="text" id="lcId" placeholder="Enter LoC ID" />
        <input type="text" id="buyer" placeholder="Buyer Name" />
        <input type="text" id="seller" placeholder="Seller Name" />
        <input type="number" id="amount" placeholder="Amount" />
        <button onclick="issueLoC()">Issue LoC</button>

        <input type="text" id="approveLcId" placeholder="Enter LoC ID to Approve" />
        <button onclick="approveLoC()">Approve LoC</button>

        <input type="text" id="completeLcId" placeholder="Enter LoC ID to Complete" />
        <button onclick="completeTransaction()">Complete Transaction</button>

        <input type="text" id="getLcId" placeholder="Enter LoC ID to View" />
        <button onclick="getLoC()">Get LoC Details</button>

        <p id="output"></p>
    </div>

    <script>
        let web3;
        let contract;
        const contractAddress = "0x1ac011DA9E9626a079c41d9144e287A70960445a";  //
Replace with actual contract address
        const contractABI =[
    {
        "inputs": [
            {
                "internalType": "string",
                "name": "_lcId",
                "type": "string"
            }
        ],
        "name": "approveLoC",
        "outputs": [],
        "stateMutability": "nonpayable",
```

```
        "type": "function"
    },
    {
        "inputs": [
            {
                "internalType": "string",
                "name": "_lcId",
                "type": "string"
            }
        ],
        "name": "completeTransaction",
        "outputs": [],
        "stateMutability": "nonpayable",
        "type": "function"
    },
    {
        "inputs": [
            {
                "internalType": "string",
                "name": "_lcId",
                "type": "string"
            },
            {
                "internalType": "string",
                "name": "_buyer",
                "type": "string"
            },
            {
                "internalType": "string",
                "name": "_seller",
                "type": "string"
            },
            {
                "internalType": "uint256",
                "name": "_amount",
                "type": "uint256"
            }
        ],
        "name": "issueLoC",
        "outputs": [],
        "stateMutability": "nonpayable",
        "type": "function"
    },
    {
        "inputs": [],
        "stateMutability": "nonpayable",
        "type": "constructor"
    },
    {
        "inputs": [],
        "name": "bank",
        "outputs": [
            {
```

```
                "internalType": "address",
                "name": "",
                "type": "address"
            }
        ],
        "stateMutability": "view",
        "type": "function"
    },
    {
        "inputs": [
            {
                "internalType": "string",
                "name": "_lcId",
                "type": "string"
            }
        ],
        "name": "getLoC",
        "outputs": [
            {
                "internalType": "string",
                "name": "",
                "type": "string"
            },
            {
                "internalType": "string",
                "name": "",
                "type": "string"
            },
            {
                "internalType": "uint256",
                "name": "",
                "type": "uint256"
            },
            {
                "internalType": "enum TradeFinance.Status",
                "name": "",
                "type": "uint8"
            }
        ],
        "stateMutability": "view",
        "type": "function"
    }
];  // Replace with actual ABI JSON

        async function connectWeb3() {
            if (window.ethereum) {
                web3 = new Web3(window.ethereum);
                await window.ethereum.enable();
                contract = new web3.eth.Contract(contractABI, contractAddress);
                console.log("Connected to Web3");
            } else {
                alert("Please install MetaMask!");
            }
```

```
        }

        async function issueLoC() {
            const accounts = await web3.eth.getAccounts();
            const lcId = document.getElementById("lcId").value;
            const buyer = document.getElementById("buyer").value;
            const seller = document.getElementById("seller").value;
            const amount = document.getElementById("amount").value;

            await contract.methods.issueLoC(lcId, buyer, seller, amount).send({ from:
accounts[0] });
            document.getElementById("output").innerText = "LoC Issued Successfully!";
        }

        async function approveLoC() {
            const accounts = await web3.eth.getAccounts();
            const lcId = document.getElementById("approveLcId").value;

            await contract.methods.approveLoC(lcId).send({ from: accounts[0] });
            document.getElementById("output").innerText = "LoC Approved!";
        }

        async function completeTransaction() {
            const accounts = await web3.eth.getAccounts();
            const lcId = document.getElementById("completeLcId").value;

            await contract.methods.completeTransaction(lcId).send({ from: accounts[0]
});
            document.getElementById("output").innerText = "Transaction Completed!";
        }

        async function getLoC() {
            const lcId = document.getElementById("getLcId").value;
            const loc = await contract.methods.getLoC(lcId).call();
            document.getElementById("output").innerText =
                Buyer: ${loc[0]}, Seller: ${loc[1]}, Amount: ${loc[2]}, Status:
${["Issued", "Approved", "Completed"][loc[3]]};
        }

        connectWeb3();
    </script>

</body>
</html>
```

Output :


Trade Finance System
123
sohith
yashu
0.0001
Issue LoC
123
Approve LoC
123
Complete Transaction
123
Get LoC Details
Buyer: , Seller: , Amount: 0, Status: Completed