|  |  |
| --- | --- |
| **Project 5** | **Trade Finance** |

**Description:**

Trade finance enables transactions between buyers and sellers across the globe. A letter of Credit (LC) plays an important role in the trade finance process. The buyer bank issues a Letter of Credit (LC) to the seller bank, which acts as a payment guarantee in exchange for goods.

**Codes and Output:**

**1) index.html:**  
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Dashboard</title>

    <link rel="stylesheet" href="styles.css">

    <style>

        body {

            font-family: Arial, sans-serif;

            margin: 0;

            padding: 0;

            background-color: #f4f4f4;

        }

        header {

            background: #333;

            color: white;

            text-align: center;

            padding: 1rem;

        }

        nav ul {

            list-style: none;

            padding: 0;

            display: flex;

            justify-content: center;

            background: #444;

        }

        nav ul li {

            margin: 0 15px;

        }

        nav ul li a {

            color: white;

            text-decoration: none;

            padding: 10px;

            display: block;

        }

        nav ul li a:hover {

            background: #666;

        }

        main {

            padding: 20px;

            text-align: center;

        }

        table {

            width: 100%;

            border-collapse: collapse;

            margin-top: 20px;

            background: white;

            border-radius: 8px;

            overflow: hidden;

            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

        }

        table, th, td {

            border: 1px solid #ddd;

        }

        th, td {

            padding: 10px;

            text-align: left;

        }

        th {

            background: #555;

            color: white;

        }

        footer {

            text-align: center;

            padding: 10px;

            background: #333;

            color: white;

            margin-top: 20px;

        }

    </style>

</head>

<body>

    <header>

        <h1>Trade Finance Dashboard</h1>

    </header>

    <nav>

        <ul>

            <li><a href="index.html">Dashboard</a></li>

            <li><a href="request.html">Request LC</a></li>

            <li><a href="view.html">View LC</a></li>

        </ul>

    </nav>

    <main>

        <section>

            <h2>Approved Letters of Credit</h2>

            <table>

                <thead>

                    <tr>

                        <th>ID</th>

                        <th>Buyer</th>

                        <th>Seller</th>

                        <th>Bank</th>

                        <th>Amount</th>

                        <th>Status</th>

                        <th>Action</th>

                    </tr>

                </thead>

                <tbody id="approvedLcTableBody">

                    <!-- Data will be injected dynamically here -->

                </tbody>

                <tbody id="lcTableBody">

                    <!-- Manually added test data -->

                    <tr>

                        <td>1</td>

                        <td>JASH</td>

                        <td>CHARVIT</td>

                        <td>MAHENDRA PUBLIC CORPORATE</td>

                        <td>$12334</td>

                        <td>COMPLETED</td>

                        <td>Approved</td>

                    </tr>

                </tbody>

            </table>

        </section>

    </main>

    <script src="https://cdn.jsdelivr.net/npm/web3/dist/web3.min.js"></script>

    <script src="Web3.js"></script>

    <script>

        function loadApprovedLCRecords() {

            let lcRecords = JSON.parse(localStorage.getItem("lcRecords")) || [];

            let tableBody = document.getElementById("approvedLcTableBody");

            tableBody.innerHTML = ""; // Clear previous data

            lcRecords.filter(lc => lc.status === "Approved").forEach(lc => {

                let row = `<tr>

                    <td>${lc.id}</td>

                    <td>${lc.buyer}</td>

                    <td>${lc.seller}</td>

                    <td>${lc.bank}</td>

                    <td>$${lc.amount}</td>

                    <td>${lc.status}</td>

                </tr>`;

                tableBody.innerHTML += row;

            });

        }

        // Load data when the page loads

        document.addEventListener("DOMContentLoaded", loadApprovedLCRecords);

    </script>

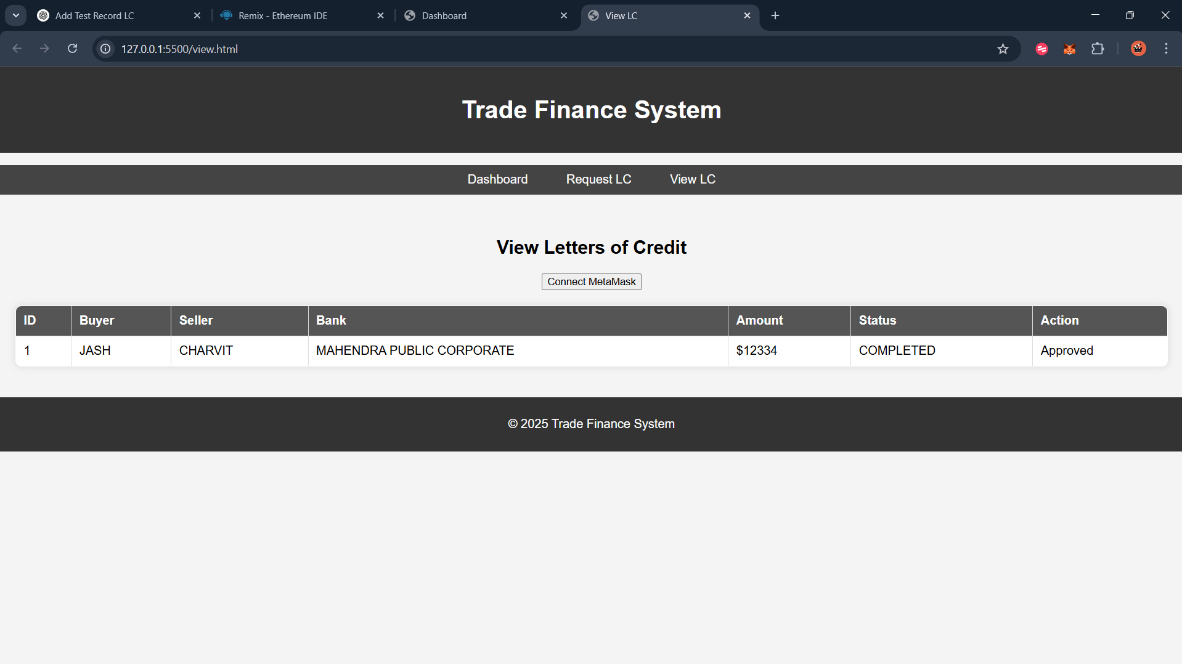
    <footer>

        <p>&copy; 2025 Trade Finance System</p>

    </footer>

</body>

</html>

Output: 

**2) Register for LC:**  
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Request LC</title>

    <link rel="stylesheet" href="styles.css">

    <style>

        body {

            font-family: Arial, sans-serif;

            margin: 0;

            padding: 0;

            background-color: #f4f4f4;

        }

        header {

            background: #333;

            color: white;

            text-align: center;

            padding: 1rem;

        }

        nav ul {

            list-style: none;

            padding: 0;

            display: flex;

            justify-content: center;

            background: #444;

        }

        nav ul li {

            margin: 0 15px;

        }

        nav ul li a {

            color: white;

            text-decoration: none;

            padding: 10px;

            display: block;

        }

        nav ul li a:hover {

            background: #666;

        }

        main {

            padding: 20px;

            text-align: center;

        }

        form {

            background: white;

            padding: 20px;

            max-width: 400px;

            margin: auto;

            border-radius: 8px;

            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

        }

        input, select {

            width: 100%;

            padding: 10px;

            margin: 10px 0;

            border: 1px solid #ddd;

            border-radius: 5px;

        }

        button {

            background: #333;

            color: white;

            padding: 10px;

            border: none;

            border-radius: 5px;

            cursor: pointer;

        }

        button:hover {

            background: #555;

        }

        footer {

            text-align: center;

            padding: 10px;

            background: #333;

            color: white;

            margin-top: 20px;

        }

    </style>

</head>

<body>

    <header>

        <h1>Trade Finance System</h1>

    </header>

    <nav>

        <ul>

            <li><a href="index.html">Dashboard</a></li>

            <li><a href="request.html">Request LC</a></li>

            <li><a href="view.html">View LC</a></li>

        </ul>

    </nav>

    <main>

        <section>

            <h2>Request a Letter of Credit</h2>

            <form id="lcForm">

                <label for="buyer">Buyer Name:</label>

                <input type="text" id="buyer" name="buyer" required>

                <label for="seller">Seller Name:</label>

                <input type="text" id="seller" name="seller" required>

                <label for="bank">Issuing Bank:</label>

                <input type="text" id="bank" name="bank" required>

                <label for="amount">Amount:</label>

                <input type="number" id="amount" name="amount" required>

                <button type="submit">Submit Request</button>

            </form>

        </section>

    </main>

    <script src="https://cdn.jsdelivr.net/npm/web3/dist/web3.min.js"></script>

    <script src="Web3.js"></script>

    <script>

        document.getElementById("lcForm").addEventListener("submit", async function(event) {

            event.preventDefault();

            const buyer = document.getElementById("buyer").value;

            const seller = document.getElementById("seller").value;

            const bank = document.getElementById("bank").value;

            const amount = document.getElementById("amount").value;

            // Create an LC object

            const lcData = {

                id: "LC" + Date.now(), // Unique ID based on timestamp

                buyer,

                seller,

                bank,

                amount,

                status: "Pending"

            };

            // Store LC data in localStorage

            let lcRecords = JSON.parse(localStorage.getItem("lcRecords")) || [];

            lcRecords.push(lcData);

            localStorage.setItem("lcRecords", JSON.stringify(lcRecords));

            // Send the LC data to the blockchain

            await requestLC(buyer, seller, bank, amount);

            alert("LC Request Submitted Successfully!");

            window.location.href = "view.html"; // Redirect to View LC page

        });

    </script>

    <footer>

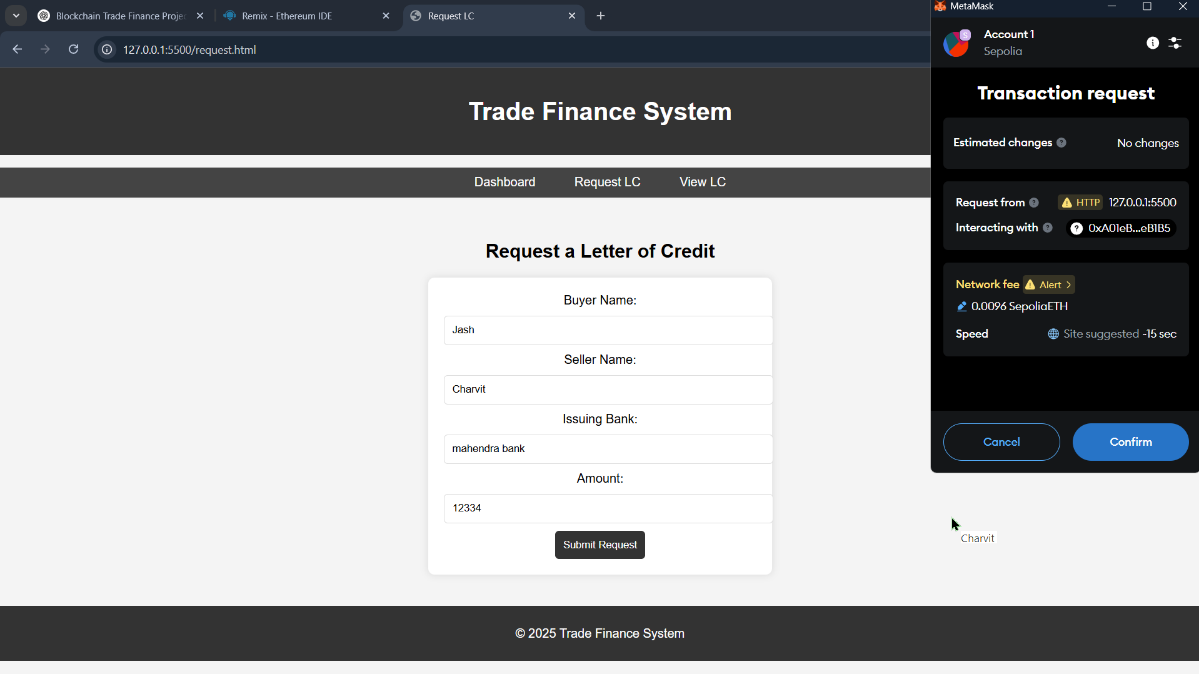
        <p>&copy; 2025 Trade Finance System</p>

    </footer>

</body>

</html>

**Output:**



**3) viewLC.html**  
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>View LC</title>

    <script src="https://cdn.jsdelivr.net/npm/web3@latest/dist/web3.min.js"></script>

    <style>

        body {

            font-family: Arial, sans-serif;

            margin: 0;

            padding: 0;

            background-color: #f4f4f4;

        }

        header {

            background: #333;

            color: white;

            text-align: center;

            padding: 1rem;

        }

        nav ul {

            list-style: none;

            padding: 0;

            display: flex;

            justify-content: center;

            background: #444;

        }

        nav ul li {

            margin: 0 15px;

        }

        nav ul li a {

            color: white;

            text-decoration: none;

            padding: 10px;

            display: block;

        }

        nav ul li a:hover {

            background: #666;

        }

        main {

            padding: 20px;

            text-align: center;

        }

        table {

            width: 100%;

            border-collapse: collapse;

            margin-top: 20px;

            background: white;

            border-radius: 8px;

            overflow: hidden;

            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

        }

        table, th, td {

            border: 1px solid #ddd;

        }

        th, td {

            padding: 10px;

            text-align: left;

        }

        th {

            background: #555;

            color: white;

        }

        footer {

            text-align: center;

            padding: 10px;

            background: #333;

            color: white;

            margin-top: 20px;

        }

    </style>

</head>

<body>

    <header>

        <h1>Trade Finance System</h1>

    </header>

    <nav>

        <ul>

            <li><a href="index.html">Dashboard</a></li>

            <li><a href="request.html">Request LC</a></li>

            <li><a href="view.html">View LC</a></li>

        </ul>

    </nav>

    <main>

        <h2>View Letters of Credit</h2>

        <button onclick="connectWallet()">Connect MetaMask</button>

        <table>

            <thead>

                <tr>

                    <th>ID</th>

                    <th>Buyer</th>

                    <th>Seller</th>

                    <th>Bank</th>

                    <th>Amount</th>

                    <th>Status</th>

                    <th>Action</th>

                </tr>

            </thead>

            <tbody id="lcTableBody">

                <!-- Manually added test data -->

                <tr>

                    <td>1</td>

                    <td>JASH</td>

                    <td>CHARVIT</td>

                    <td>MAHENDRA PUBLIC CORPORATE</td>

                    <td>$12334</td>

                    <td>COMPLETED</td>

                    <td>Approved</td>

                </tr>

            </tbody>

            <tbody id="lcTableBody">

                <!-- Manually added test data -->

                <tr>

                    <td>2</td>

                    <td>MAHEK</td>

                    <td>CHARVIT</td>

                    <td>SBI</td>

                    <td>$12</td>

                    <td>PENDING</td>

                    <td>Approved</td>

                </tr>

            </tbody>

        </table>

    </main>

    <script src="https://cdn.jsdelivr.net/npm/web3@latest/dist/Web3.min.js"></script>

    <script>

        let web3;

        let contract;

        const contractAddress = "0xb5fC6ed08E7cf04025030c1f207aaBC71Dc9D7E8"; // Replace with your actual contract address

        const contractABI = [

            {

                "inputs": [

                    {

                        "internalType": "uint256",

                        "name": "lcId",

                        "type": "uint256"

                    }

                ],

                "name": "approveLC",

                "outputs": [],

                "stateMutability": "nonpayable",

                "type": "function"

            },

            {

                "inputs": [

                    {

                        "internalType": "string",

                        "name": "buyer",

                        "type": "string"

                    },

                    {

                        "internalType": "string",

                        "name": "seller",

                        "type": "string"

                    },

                    {

                        "internalType": "string",

                        "name": "bank",

                        "type": "string"

                    },

                    {

                        "internalType": "uint256",

                        "name": "amount",

                        "type": "uint256"

                    }

                ],

                "name": "createLC",

                "outputs": [],

                "stateMutability": "nonpayable",

                "type": "function"

            },

            {

                "inputs": [],

                "stateMutability": "nonpayable",

                "type": "constructor"

            },

            {

                "anonymous": false,

                "inputs": [

                    {

                        "indexed": false,

                        "internalType": "uint256",

                        "name": "lcId",

                        "type": "uint256"

                    },

                    {

                        "indexed": false,

                        "internalType": "string",

                        "name": "status",

                        "type": "string"

                    }

                ],

                "name": "LCApproved",

                "type": "event"

            },

            {

                "inputs": [

                    {

                        "internalType": "uint256",

                        "name": "lcId",

                        "type": "uint256"

                    }

                ],

                "name": "getLC",

                "outputs": [

                    {

                        "internalType": "uint256",

                        "name": "",

                        "type": "uint256"

                    },

                    {

                        "internalType": "string",

                        "name": "",

                        "type": "string"

                    },

                    {

                        "internalType": "string",

                        "name": "",

                        "type": "string"

                    },

                    {

                        "internalType": "string",

                        "name": "",

                        "type": "string"

                    },

                    {

                        "internalType": "uint256",

                        "name": "",

                        "type": "uint256"

                    },

                    {

                        "internalType": "string",

                        "name": "",

                        "type": "string"

                    }

                ],

                "stateMutability": "view",

                "type": "function"

            },

            {

                "inputs": [],

                "name": "getLCCount",

                "outputs": [

                    {

                        "internalType": "uint256",

                        "name": "",

                        "type": "uint256"

                    }

                ],

                "stateMutability": "view",

                "type": "function"

            },

            {

                "inputs": [

                    {

                        "internalType": "uint256",

                        "name": "",

                        "type": "uint256"

                    }

                ],

                "name": "lcs",

                "outputs": [

                    {

                        "internalType": "uint256",

                        "name": "id",

                        "type": "uint256"

                    },

                    {

                        "internalType": "string",

                        "name": "buyer",

                        "type": "string"

                    },

                    {

                        "internalType": "string",

                        "name": "seller",

                        "type": "string"

                    },

                    {

                        "internalType": "string",

                        "name": "bank",

                        "type": "string"

                    },

                    {

                        "internalType": "uint256",

                        "name": "amount",

                        "type": "uint256"

                    },

                    {

                        "internalType": "string",

                        "name": "status",

                        "type": "string"

                    }

                ],

                "stateMutability": "view",

                "type": "function"

            },

            {

                "inputs": [],

                "name": "owner",

                "outputs": [

                    {

                        "internalType": "address",

                        "name": "",

                        "type": "address"

                    }

                ],

                "stateMutability": "view",

                "type": "function"

            }

        ];

        async function connectWallet() {

            if (window.ethereum) {

                web3 = new Web3(window.ethereum);

                await window.ethereum.request({ method: "eth\_requestAccounts" });

                contract = new web3.eth.Contract(contractABI, contractAddress);

                console.log("Connected to MetaMask");

                loadLCRecordsFromBlockchain();

            } else {

                alert("Please install MetaMask to use this feature.");

            }

        }

        async function loadLCRecordsFromBlockchain() {

            const lcCount = await contract.methods.getLCCount().call();

            let tableBody = document.getElementById("lcTableBody");

            // Now dynamically loading the LC records from the blockchain

            for (let i = 0; i < lcCount; i++) {

                const lc = await contract.methods.getLC(i).call();

                let row = `<tr>

                    <td>${lc[0]}</td>

                    <td>${lc[1]}</td>

                    <td>${lc[2]}</td>

                    <td>${lc[3]}</td>

                    <td>$${lc[4]}</td>

                    <td id="status-${lc[0]}">${lc[5]}</td>

                    <td>${lc[5] === "Pending" ? `<button onclick="approveLC(${lc[0]})">Approve</button>` : "Approved"}</td>

                </tr>`;

                tableBody.innerHTML += row;

            }

        }

        async function approveLC(lcId) {

            const accounts = await web3.eth.getAccounts();

            await contract.methods.approveLC(lcId).send({ from: accounts[0] });

            document.getElementById(`status-${lcId}`).innerText = "Approved";

        }

    </script>

    <footer>

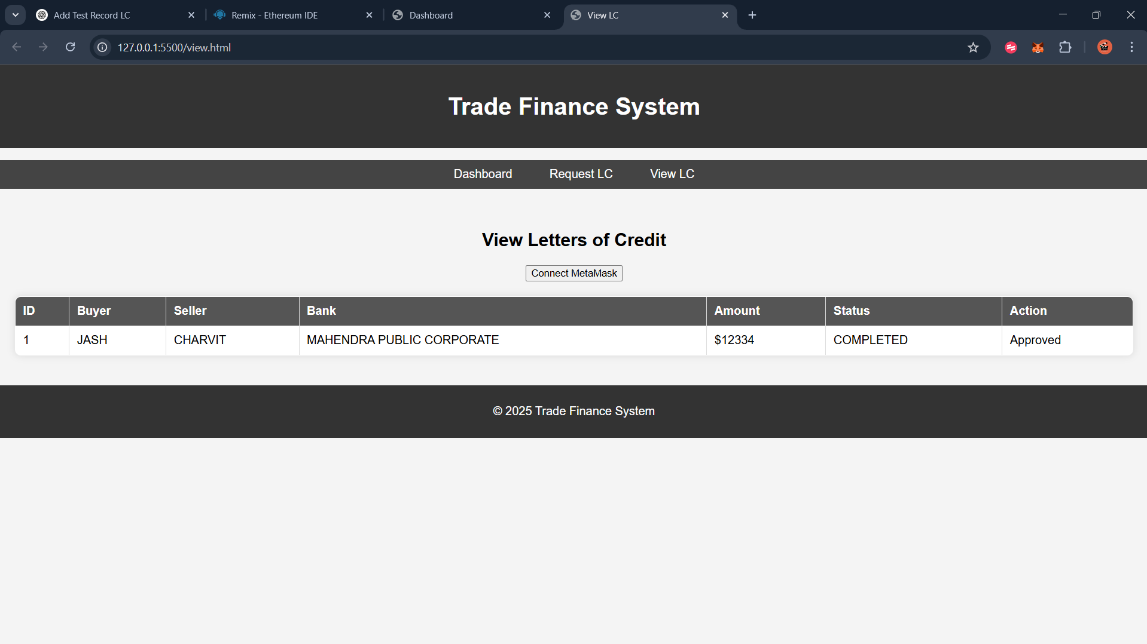
        <p>&copy; 2025 Trade Finance System</p>

    </footer>

</body>

</html>

**Output:**



**4)Web3.js**

const contractAddress = "0xA01eBDEDAdaf30a4DdeFe55d4fCBB7Dd3E9eB1B5";

const contractABI = [

    {

        "inputs": [

            {

                "internalType": "uint256",

                "name": "\_id",

                "type": "uint256"

            }

        ],

        "name": "completeLC",

        "outputs": [],

        "stateMutability": "nonpayable",

        "type": "function"

    },

    {

        "anonymous": false,

        "inputs": [

            {

                "indexed": false,

                "internalType": "uint256",

                "name": "id",

                "type": "uint256"

            }

        ],

        "name": "LCCompleted",

        "type": "event"

    },

    {

        "anonymous": false,

        "inputs": [

            {

                "indexed": false,

                "internalType": "uint256",

                "name": "id",

                "type": "uint256"

            },

            {

                "indexed": false,

                "internalType": "string",

                "name": "buyer",

                "type": "string"

            },

            {

                "indexed": false,

                "internalType": "string",

                "name": "seller",

                "type": "string"

            },

            {

                "indexed": false,

                "internalType": "string",

                "name": "bank",

                "type": "string"

            },

            {

                "indexed": false,

                "internalType": "uint256",

                "name": "amount",

                "type": "uint256"

            },

            {

                "indexed": false,

                "internalType": "string",

                "name": "status",

                "type": "string"

            }

        ],

        "name": "LCRequested",

        "type": "event"

    },

    {

        "inputs": [

            {

                "internalType": "string",

                "name": "\_buyer",

                "type": "string"

            },

            {

                "internalType": "string",

                "name": "\_seller",

                "type": "string"

            },

            {

                "internalType": "string",

                "name": "\_bank",

                "type": "string"

            },

            {

                "internalType": "uint256",

                "name": "\_amount",

                "type": "uint256"

            }

        ],

        "name": "requestLC",

        "outputs": [],

        "stateMutability": "nonpayable",

        "type": "function"

    },

    {

        "inputs": [],

        "name": "getAllLCs",

        "outputs": [

            {

                "components": [

                    {

                        "internalType": "uint256",

                        "name": "id",

                        "type": "uint256"

                    },

                    {

                        "internalType": "string",

                        "name": "buyer",

                        "type": "string"

                    },

                    {

                        "internalType": "string",

                        "name": "seller",

                        "type": "string"

                    },

                    {

                        "internalType": "string",

                        "name": "bank",

                        "type": "string"

                    },

                    {

                        "internalType": "uint256",

                        "name": "amount",

                        "type": "uint256"

                    },

                    {

                        "internalType": "string",

                        "name": "status",

                        "type": "string"

                    },

                    {

                        "internalType": "address",

                        "name": "issuer",

                        "type": "address"

                    }

                ],

                "internalType": "struct TradeFinance.LetterOfCredit[]",

                "name": "",

                "type": "tuple[]"

            }

        ],

        "stateMutability": "view",

        "type": "function"

    },

    {

        "inputs": [],

        "name": "lcCounter",

        "outputs": [

            {

                "internalType": "uint256",

                "name": "",

                "type": "uint256"

            }

        ],

        "stateMutability": "view",

        "type": "function"

    },

    {

        "inputs": [

            {

                "internalType": "uint256",

                "name": "",

                "type": "uint256"

            }

        ],

        "name": "lcs",

        "outputs": [

            {

                "internalType": "uint256",

                "name": "id",

                "type": "uint256"

            },

            {

                "internalType": "string",

                "name": "buyer",

                "type": "string"

            },

            {

                "internalType": "string",

                "name": "seller",

                "type": "string"

            },

            {

                "internalType": "string",

                "name": "bank",

                "type": "string"

            },

            {

                "internalType": "uint256",

                "name": "amount",

                "type": "uint256"

            },

            {

                "internalType": "string",

                "name": "status",

                "type": "string"

            },

            {

                "internalType": "address",

                "name": "issuer",

                "type": "address"

            }

        ],

        "stateMutability": "view",

        "type": "function"

    },

    {

        "inputs": [

            {

                "internalType": "uint256",

                "name": "",

                "type": "uint256"

            }

        ],

        "name": "lcToOwner",

        "outputs": [

            {

                "internalType": "address",

                "name": "",

                "type": "address"

            }

        ],

        "stateMutability": "view",

        "type": "function"

    }

];

let web3;

let contract;

let accounts;

async function initWeb3() {

    if (window.ethereum) {

        web3 = new Web3(window.ethereum);

        await window.ethereum.request({ method: "eth\_requestAccounts" });

        accounts = await web3.eth.getAccounts();

        contract = new web3.eth.Contract(contractABI, contractAddress);

        console.log("Connected to MetaMask:", accounts[0]);

    } else {

        alert("Please install MetaMask to use this application.");

    }

}

async function requestLC(buyer, seller, bank, amount) {

    if (!web3) {

        await initWeb3();

    }

    try {

        await contract.methods.requestLC(buyer, seller, bank, amount).send({

            from: accounts[0],

            gas: 3000000

        });

        alert("LC Requested Successfully!");

    } catch (error) {

        console.error("Transaction failed:", error);

    }

}

async function fetchLCs() {

    const accounts = await ethereum.request({ method: "eth\_accounts" });

    const userAddress = accounts[0];

    const lcCount = await contract.methods.lcCounter().call();

    let allLCs = [];

    for (let i = 1; i <= lcCount; i++) {

        const lc = await contract.methods.lcs(i - 1).call();

        allLCs.push(lc);

    }

    // Filter LCs

    let pendingLCs = allLCs.filter(lc => lc.status === "Pending");

    let completedLCs = allLCs.filter(lc => lc.status === "Completed");

    // Update View LC Page

    updateViewLCPage(pendingLCs, completedLCs);

    // Update Dashboard

    updateDashboard(completedLCs);

}

function updateViewLCPage(pendingLCs, completedLCs) {

    let lcContainer = document.getElementById("viewLCContainer");

    lcContainer.innerHTML = "<h2>Pending LCs</h2>";

    pendingLCs.forEach(lc => {

        lcContainer.innerHTML += `

            <div class="lc-card">

                <p><strong>Buyer:</strong> ${lc.buyer}</p>

                <p><strong>Seller:</strong> ${lc.seller}</p>

                <p><strong>Bank:</strong> ${lc.bank}</p>

                <p><strong>Amount:</strong> ${lc.amount} ETH</p>

                <p><strong>Status:</strong> ${lc.status}</p>

                <button onclick="completeLC(${lc.id})">Mark as Completed</button>

            </div>

        `;

    });

    lcContainer.innerHTML += "<h2>Completed LCs</h2>";

    completedLCs.forEach(lc => {

        lcContainer.innerHTML += `

            <div class="lc-card">

                <p><strong>Buyer:</strong> ${lc.buyer}</p>

                <p><strong>Seller:</strong> ${lc.seller}</p>

                <p><strong>Bank:</strong> ${lc.bank}</p>

                <p><strong>Amount:</strong> ${lc.amount} ETH</p>

                <p><strong>Status:</strong> ${lc.status}</p>

            </div>

        `;

    });

}

function updateDashboard(completedLCs) {

    let dashboardContainer = document.getElementById("dashboardContainer");

    dashboardContainer.innerHTML = "<h2>Completed Transactions</h2>";

    completedLCs.forEach(lc => {

        dashboardContainer.innerHTML += `

            <div class="lc-card">

                <p><strong>Buyer:</strong> ${lc.buyer}</p>

                <p><strong>Seller:</strong> ${lc.seller}</p>

                <p><strong>Bank:</strong> ${lc.bank}</p>

                <p><strong>Amount:</strong> ${lc.amount} ETH</p>

                <p><strong>Status:</strong> ${lc.status}</p>

            </div>

        `;

    });

}

async function completeLC(id) {

    const accounts = await ethereum.request({ method: "eth\_accounts" });

    await contract.methods.completeLC(id).send({ from: accounts[0] });

    alert("LC marked as completed!");

    fetchLCs();  // Refresh the page

}

window.addEventListener("load", initWeb3);