**Project 1**

**Problem Statement:**

Design and implement a decentralized lottery system using Ethereum blockchain and smart contracts. The lottery should allow participants to enter by sending a minimum amount of Ether, and the manager (the creator of the smart contract) should be able to pick a random winner from the list of participants. The application should have a simple HTML front-end for users to interact with the smart contract, and JavaScript code to facilitate communication between the front-end and the Ethereum blockchain via the Web3 library. Ensure that the smart contract enforces necessary conditions such as a minimum entry fee and that only the manager can pick a winner. Additionally, display the list of participants on the front-end, and provide functionality to connect to a user's Ethereum wallet using MetaMask.

**Suggested Solution:**

**Smart contract:**

*// SPDX-License-Identifier: MIT*

*pragma solidity ^0.8.0;*

*contract Lottery {*

*address public manager;*

*address payable[] public participants;*

*constructor() {*

*manager = msg.sender;*

*}*

*function enter() public payable {*

*require(msg.value > .01 ether, "Minimum 0.01 ETH required to enter");*

*participants.push(payable(msg.sender));*

*}*

*function random() private view returns (uint) {*

*return uint(keccak256(abi.encodePacked(blockhash(block.number-1), block.timestamp, participants)));*

*}*

*function pickWinner() public restricted {*

*require(msg.sender == manager, "Only the manager can pick a winner");*

*require(participants.length > 3, "Not enough participants");*

*uint index = random() % participants.length;*

*participants[index].transfer(address(this).balance);*

*// Reset the lottery for the next round*

*participants = new address payable[](0);*

*}*

*modifier restricted() {*

*require(msg.sender == manager);*

*\_;*

*}*

*function getParticipants() public view returns (address payable[] memory) {*

*return participants;*

*}*

*}*

**HTML Frontend:**

*<!DOCTYPE html>*

*<html lang="en">*

*<head>*

*<meta charset="UTF-8">*

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

*<title>Lottery Smart Contract</title>*

*<script* [*src="https://cdn.jsdelivr.net/npm/web3@1.3.0/dist/web3.min.js"></script*](mailto:src=%22https://cdn.jsdelivr.net/npm/web3@1.3.0/dist/web3.min.js%22%3e%3c/script)*>*

*</head>*

*<body>*

*<h1>Lottery Contract</h1>*

*<button id="connectWallet">Connect to Wallet</button>*

*<div>*

*<h2>Enter Lottery</h2>*

*<button id="enterLottery">Enter</button>*

*<p>Cost to enter: 0.01 ETH</p>*

*</div>*

*<div>*

*<h2>Pick a Winner (Only Manager)</h2>*

*<button id="pickWinner">Pick Winner</button>*

*</div>*

*<div>*

*<h2>Participants</h2>*

*<ul id="participantsList"></ul>*

*</div>*

*<script src="app.js"></script>*

*</body>*

*</html>*

**JavaScript for Interacting with the Contract:**

*const contractAddress = "YOUR\_CONTRACT\_ADDRESS\_HERE";*

*const abi = YOUR\_CONTRACT\_ABI\_HERE;*

*let contract;*

*let web3;*

*let accounts;*

*window.addEventListener('load', async () => {*

*if(window.ethereum) {*

*web3 = new Web3(window.ethereum);*

*try {*

*await window.ethereum.enable();*

*initApp();*

*} catch (error) {*

*console.error("Access to your Ethereum account rejected.");*

*}*

*} else {*

*console.error("Please install MetaMask!");*

*}*

*});*

*function initApp() {*

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

*document.getElementById('connectWallet').addEventListener('click', async () => {*

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

*console.log("Connected account:", accounts[0]);*

*});*

*document.getElementById('enterLottery').addEventListener('click', () => {*

*contract.methods.enter().send({ from: accounts[0], value: web3.utils.toWei("0.01", "ether") })*

*.then(() => console.log("Entered the lottery!"))*

*.catch(console.error);*

*});*

*document.getElementById('pickWinner').addEventListener('click', () => {*

*contract.methods.pickWinner().send({ from: accounts[0] })*

*.then(() => console.log("Winner picked!"))*

*.catch(console.error);*

*});*

*// Fetch and display participants*

*contract.methods.getParticipants().call()*

*.then(displayParticipants)*

*.catch(console.error);*

*}*

*function displayParticipants(participants) {*

*const participantsList = document.getElementById('participantsList');*

*participantsList.innerHTML = participants.map(address => `<li>${address}</li>`).join('');}*