Pune Institute of Computer Technology Dhankawadi, Pune

A MINI PROJECT REPORT ON

Blockchain-Based Decentralized Application (dApp) for E-Voting System

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE IN THE PARTIAL FULFILLMENT OF THE LP III

BE (COMPUTER ENGINEERING)

SUBMITTED BY

Roll No.	Name
41471	Prachi Said

Under the Guidance of

Prof. S. P. Shintre

DEPARTMENT OF COMPUTER ENGINEERING

Academic Year 2024-25

PUNE INSTITUTE OF COMPUTER TECHNOLOGY DEPARTMENT OF COMPUTER ENGINEERING

CERTIFICATE

This is to certify that the Mini Project report of LP III (BT) entitled

”Blockchain-Based Decentralized Application (dApp) for E-Voting System”

Submitted by

Roll No.	**Name**
41471	Prachi Said

has satisfactorily completed a micro project report under the guidance of Prof. S. P. Shintre towards the partial fulfillment of BE Computer Engineering, Academic Year 2024-25 of Savitribai Phule Pune University.

Prof. S. P. Shintre
(Lab Guide)

Dr. Geetanjali Kale
(H. O. D)

Date: / / Place: Pune

Title :

Blockchain-Based Decentralized Application (dApp) for E-Voting System

Problem Statement :

Traditional voting systems face challenges such as fraud, tampering, transparency issues, and trust. These systems rely heavily on centralized authorities, making them vulnerable to cyber-attacks, manipulation, and lack of accountability. A decentralized, transparent, and secure voting system is essential to ensure voter trust and integrity in elections. This project aims to develop an e-voting system using blockchain technology, which ensures transparency, immutability, and decentralization for secure voting.

Learning Objectives:

1. Understand the fundamentals of blockchain technology and how it provides decentralization and immutability.
2. Explore the architecture of decentralized applications (dApps) and their use in secure, transparent systems.
3. Develop a smart contract for voting, handling voter registration, and ensuring that each vote is recorded immutably on the blockchain.
4. Implement a blockchain-based e-voting system to prevent fraud, ensure transparency, and protect voter privacy.

Theory:

Blockchain is a decentralized, immutable ledger where data is stored in a distributed network of nodes. Each node holds a copy of the blockchain, ensuring transparency and trust across the system. A decentralized application (dApp) built on top of blockchain uses smart contracts — self-executing contracts with the agreement directly written into code — to automate processes such as voting. In this context, blockchain ensures that votes are immutable, preventing tampering, while the decentralized nature removes the need for trust in a central authority.

Key concepts include:

- **Blockchain**: A distributed ledger where all transactions are recorded across multiple nodes.
- **Smart Contracts:** Code that runs on the blockchain and automatically enforces rules.
- **Ethereum** (or other blockchain platforms): A platform for building dApps using smart contracts.

Algorithm Complexity

Voter Registration:

- Check if the user is eligible to vote (via identity proof).
- Register the voter's Ethereum address in the system.

Vote Casting:

- Each voter can vote only once.
- The vote is recorded as a transaction on the blockchain.
- The vote is associated with the voter's Ethereum address but does not reveal the identity of the voter.

Vote Counting:

- At the end of the voting period, the votes are automatically tallied.
- Results are publicly available and transparent to all.

Security Mechanism:

- Ensure only authorized users can vote.
- Ensure each user votes only once.
- Votes are immutable, once cast, they cannot be altered or deleted.

Methodology/Algorithms Details:

Blockchain Selection:

- Use Ethereum as the blockchain platform for development, given its support for smart
- contracts and dApps.

Smart Contract Design:

- Create a smart contract that governs the voting process, ensuring:
- Voter registration.
- Secure vote submission.
- Vote counting and result display.

Front-End Development:

- Develop a user-friendly interface using web technologies (React.js).
- Connect the front-end to the Ethereum blockchain using Web3.js or ethers.js library.

Testing and Deployment:

- Test the smart contract using a test blockchain environment like Ganache.
- Deploy the smart contract to Ethereum's Rinkeby or Kovan test network for public
- testing.

Code Implementation

```
pragma solidity ^0.8.0;
contract Voting {

    struct Candidate {
        uint id;
        string name;
        uint voteCount;
    }
    struct Voter {
        bool voted;
        uint vote;
    }
    address public owner;
    mapping(uint => Candidate) public candidates;
    mapping(address => Voter) public voters;
    uint public candidatesCount;
    uint public totalVotes;
    modifier onlyOwner() {
        require(msg.sender == owner, "Only owner can call this function");
        _;
    }
    constructor() {
        owner = msg.sender;
        addCandidate("Alice");
        addCandidate("Bob");}
        function addCandidate(string memory _name) public onlyOwner {
        candidatesCount++;
        candidates[candidatesCount] = Candidate(candidatesCount, _name, 0);
    }
    function vote(uint _candidateId) public {
        require(!voters[msg.sender].voted, "Already voted");
        require(_candidateId > 0 && _candidateId <= candidatesCount, "Invalid
        candidate");
        voters[msg.sender].voted = true;
        voters[msg.sender].vote = _candidateId;
        candidates[_candidateId].voteCount++;
        totalVotes++;
```

```
    }
    function getCandidate(uint _id) public view returns (string memory name, uint voteCount)
    {
        Candidate memory candidate = candidates[_id];
        return (candidate.name, candidate.voteCount);
    }
}
```

Testing:

- Deploy the smart contract using Truffle or Hardhat framework.
- Interact with the deployed contract using Ganache for local testing or Ethereum testnets (Rinkeby/Kovan).

Deployment:

- Once tested, deploy the smart contract to Ethereum's mainnet or other blockchain platforms like Polygon.

Example Output:

Voting contract deployed at address: 0x123...ABC
Voter with account: 0x456...DEF voted for candidate: Alice

Result:

Candidate: Alice, Votes: 120 Candidate: Bob, Votes: 85 Total Votes: 205

Analysis :

The blockchain-based e-voting system was successfully developed using smart contracts for secure and transparent voting. Key benefits include:

- Immutability: Once a vote is cast, it cannot be altered, ensuring the integrity of the
- election.
- Transparency: All votes are publicly available on the blockchain, allowing for real-
- time auditing.
- Decentralization: The system does not rely on a central authority, increasing voter
- trust.

The system could be further improved by integrating privacy-preserving techniques such as zero-knowledge proofs to ensure voter anonymity while maintaining transparency.

Conclusion :

This project demonstrates the power of blockchain in solving critical issues such as transparency, immutability, and trust in voting systems. The decentralized nature of the application, combined with smart contracts, ensures that no central authority can manipulate the outcome, making the e-voting system secure and reliable for large-scale elections.