**Name: Jash Mavani**

**Roll No: 19BCE123**

**Batch: EL2**

**Course Name & Course Code: 2CSDE93 & Blockchain Technology**

**Practical-6**

## Aim: - To build, implement and test voting mechanism using Ethereum Blockchain. First, list the contestants on the screen and the vote they got. Whenever the user tries to vote a particular contestant, the count of the votes for the particular contestant should increase by 1. Also, the user who has already voted should be marked. Marked means “the user has already voted once and will not be allowed to vote again”.

**Code:-**

pragma solidity ^0.4.21;

contract Election {

    struct Candidate {

        string name;

        uint voteCount;

    }

    struct Voter {

        bool authorized;

        bool voted;

        uint vote;

    }

    address public owner;

    string public electionName;

    mapping(address => Voter) public voters;

    Candidate[] public candidates;

    uint public totalVotes;

    modifier ownerOnly() {

        require(msg.sender == owner);

        \_; //remaining body of addCandidate to be executed

    }

    function Election(string \_name) public {

        owner = msg.sender;

        electionName = \_name;

    }

    function addCandidate(string \_name) ownerOnly public {

        candidates.push(Candidate(\_name,0));

    }

    function getNumCandidate() public view returns(uint) {

        return candidates.length;

    }

    function authorize(address \_person) ownerOnly public {

        voters[\_person].authorized = true;

    }

    function vote(uint \_voteIndex) public {

        require(!voters[msg.sender].voted);

        require(voters[msg.sender].authorized);

        voters[msg.sender].vote = \_voteIndex;

        voters[msg.sender].voted = true;

        candidates[\_voteIndex].voteCount += 1;

        totalVotes += 1;

    }

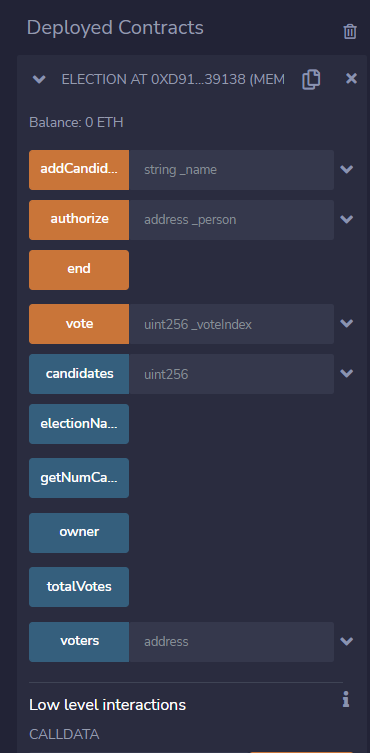
    function end() ownerOnly public {

        selfdestruct(owner);

    }

}

**Output :-**

****