

Practical 6

Aim: Develop a voting application using Hyperledger and Ethereum. Build a decentralised app that combines Ethereum's Web3 and Solidity smart contracts with Hyperledger hosting Fabric and Chaincode EVM.

Code:

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

contract Voting {
    // Declare variables to store the total votes for each team
    uint256 public votesForTeamA;
    uint256 public votesForTeamB;
    uint256 public votesForTeamC;
    address public owner;
    mapping(address => bool) authorizedVoters;

    constructor() {
        owner = msg.sender;
    }

    modifier onlyAuthorizedVoter() {
        require(
            authorizedVoters[msg.sender] == true,
            "You are not authorized to vote"
        );
        _;
    }

    modifier onlyOwner() {
        require(msg.sender == owner, "Only the owner can call this
function");
        _;
    }

    // Function to allow a voter to cast their vote for a team
    function vote(uint256 _team) public onlyAuthorizedVoter {
        // Check the value of _team and increment the corresponding team's
        vote count
        if (_team == 1) {
            votesForTeamA += 1;
        } else if (_team == 2) {
            votesForTeamB += 1;
        } else if (_team == 3) {
            votesForTeamC += 1;
        }
    }
}
```

```

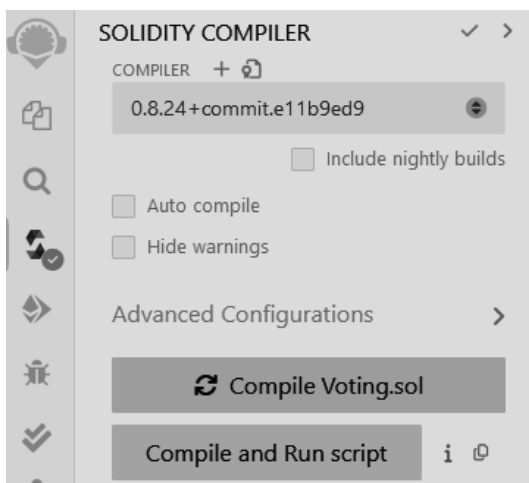
// Function to add a voter to the list of authorized voters
function addVoter(address _voter) public onlyOwner {
    require(msg.sender == owner, "Only the owner can add a voter");
    authorizedVoters[_voter] = true;
}

// Function to declare results of the vote
function getWinner() public view returns (string memory result) {
    // Check the vote counts and return the winner
    if (votesForTeamA > votesForTeamB && votesForTeamA > votesForTeamC)
    {
        result = "Team A is the winner!";
    } else if (
        votesForTeamB > votesForTeamA && votesForTeamB > votesForTeamC
    ) {
        result = "Team B is the winner!";
    } else if (
        votesForTeamC > votesForTeamA && votesForTeamC > votesForTeamB
    ) {
        result = "Team C is the winner!";
    } else {
        result = "There is a tie!";
    }
}

function getTotalVotes() public view returns (uint256) {
    return votesForTeamA + votesForTeamB + votesForTeamC;
}
}

```

Compile The Contract



Deploy The Contract

DEPLOY & RUN TRANSACTIONS ✓ >

ENVIRONMENT ▾
Remix VM (Shanghai) ⓘ

VM

ACCOUNT +
0x5B3...eddC4 (100 ether) ⓘ

GAS LIMIT
3000000 ▾

VALUE
0 ▾ Wei ▾

CONTRACT
Voting - contracts/Voting.sol ▾
evm version: shanghai

Deploy

Publish to IPFS

At Address Load contract from Address

Adding voters

(Note: you can get wallet address from above dropdown; change it to another and copy it)

addVoter cF9b849Ae677dD3315835cb2 ▾

✓ [vm] from: 0x5B3...eddC4 to: Voting.addVoter(address) 0xd91...39138 value: 0 wei data: 0xf4a...35cb2 logs: 0 hash: 0x585...2498b

Voting

vote 1 ▾

✓ [vm] from: 0xAb8...35cb2 to: Voting.vote(uint256) 0xd91...39138 value: 0 wei data: 0x012...00001 logs: 0 hash: 0xd19...53ff6

Checking the winner

getWinner

0: string: result Team A is the winner!