



School: Campus:
Academic Year: Subject Name: Subject Code:
Semester: Program: Branch: Specialization:
Date:

Applied and Action Learning

(Learning by Doing and Discovery)

Name of the Experiment :

* Coding Phase: Pseudo Code / Flow Chart / Algorithm

Algorithm:

1. Create a governance smart contract with functions:
 - Create Proposal
 - Vote on Proposal
 - View Proposal Status
2. Deploy the contract using Remix and connect using MetaMask.
3. Cast votes from multiple Ethereum accounts to simulate DAO interaction.
4. Verify proposal acceptance or rejection based on votes.

* Softwares used

1. Remix IDE
2. Solidity ^0.8.x
3. MetaMask Wallet
4. Sepolia Testnet (for multiple voting accounts).

* Testing Phase: Compilation of Code (error detection)

1. Contract deployed using Remix in JavaScript VM.
2. Created 2 proposals using the createProposal() function.
3. Simulated multiple votes from different accounts using vote() function.
4. Verified vote counts increased correctly using getProposal().
5. No double-voting was possible due to voters mapping checks.

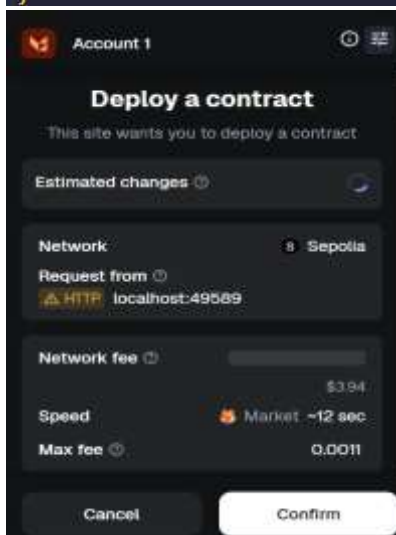
```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.7;

contract SimpleDAO {
    struct Proposal {
        string description;
        uint voteCount;
        bool executed;
    }
    Proposal[] public proposals;
    mapping(address => bool) public voters;

    function createProposal(string memory _description) public {
        proposals.push(Proposal({
            description: _description,
            voteCount: 0,
            executed: false
        }));
    }

    // Vote on a proposal by index
    function vote(uint proposalIndex) public {
        require(!voters[msg.sender], "You have already voted.");
        proposals[proposalIndex].voteCount++;
        voters[msg.sender] = true;
    }

    // Get proposal details
    function getProposal(uint proposalIndex) public view returns (string memory desc, uint voteCount, bool executed) {
        Proposal memory p = proposals[proposalIndex];
        return (p.description, p.voteCount, p.executed);
    }
}
```



* Implementation Phase: Final Output (no error)

Applied and Action Learning



* Observations

1. DAO smart contracts enable decentralized and transparent decision-making.
2. Voting process is handled via blockchain, making it immutable and verifiable.
3. Basic DAO contracts can be extended with token-weighted voting and execution logic.
4. Testing in Remix with multiple accounts is useful for simulating real DAOs.

ASSESSMENT

Rubrics	Full Mark	Marks Obtained	Remarks
Concept	10		
Planning and Execution/ Practical Simulation/ Programming	10		
Result and Interpretation	10		
Record of Applied and Action Learning	10		
Viva	10		
Total	50		

Signature of the Student:

Name :

Regn. No. :

Signature of the Faculty:

Page No.....

*As applicable according to the experiment.
Two sheets per experiment (10-20) to be used.