Centurion University
Shaping Lives Empowering Communities...

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 Experiement : DAO in Action – Governance Simulation

* Coding Phase: Pseudo Code / Flow Chart / Algorithm

ALGORITHM:

1. Start
2. Open Remix IDE and create a new Solidity file named `DAOGovernance.sol`.
3. Import OpenZeppelin libraries for ownership and governance functionality like Ownable, ERC20, etc.
4. Define the DAO structure with the following main functions:
    - createProposal() – to allow members to submit proposals.
    - voteOnProposal() – to enable token holders to vote.
    - executeProposal() – to execute proposals after voting ends.
5. Compile the smart contract using the Solidity compiler in Remix IDE. Connect MetaMask Wallet with Remix and select the Injected Provider (MetaMask) environment.
6. Deploy the contract on the Sepolia Testnet network. Copy the deployed Contract Address and ABI.
7. Create a React frontend using create-react-app.
8. Install Ethers.js to connect the frontend with the blockchain.
9. Integrate smart contract functions in the frontend to: 1) Display proposals. 2) Allow users to vote using their connected wallet. 3) Execute successful proposals.
10. Run the DApp using npm start and interact with the DAO simulation interface.
11. End

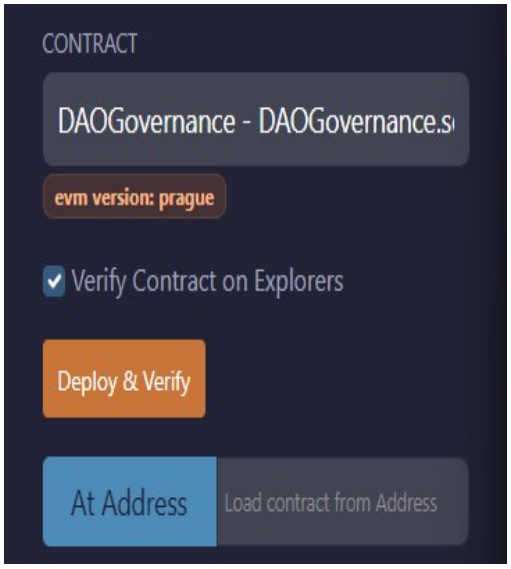
* Software Used:

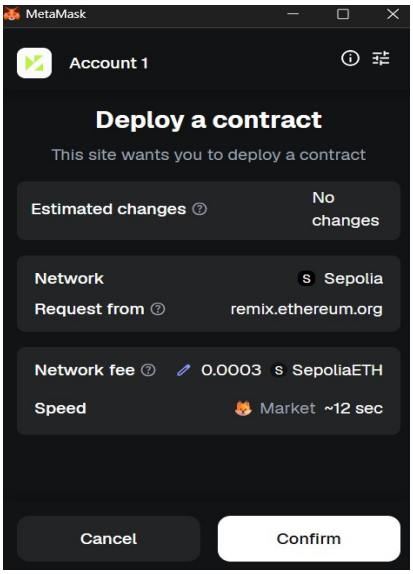
1. Remix IDE : For smart contract development and compilation.
2. MetaMask Wallet
3. React.js and Ethers.js
4. Sepolia Testnet (Ethereum)
5. Brave/Chrome Browser

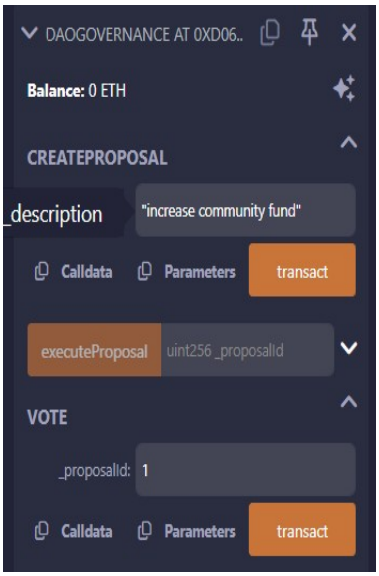
*As applicable according to the experiment.
Two sheets per experiment (10-20) to be used.

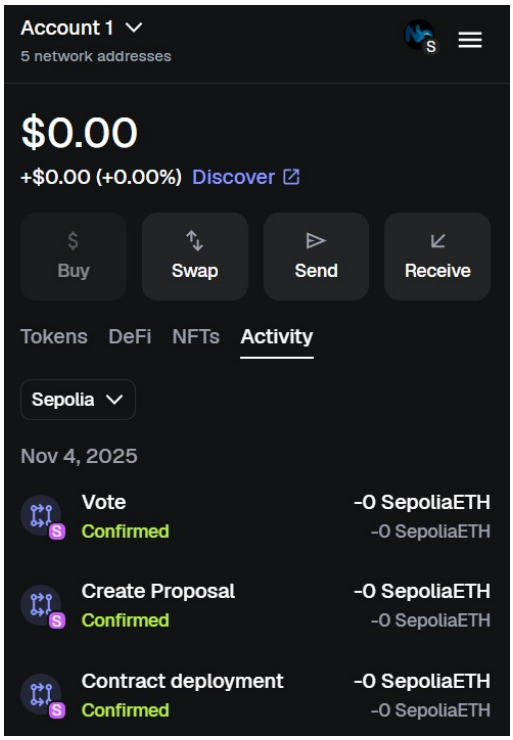
* Implementation Phase: Final Output (no error)

1. Open **Remix IDE** and paste the DAO contract code (DAOGovernance.sol). and Select the correct **Solidity compiler version (0.8.x)** and click **Compile**. Resolve all syntax or import errors (especially OpenZeppelin imports).
2. After successful compilation, verify the **ABI** and **Bytecode** are generated.
3. Connect MetaMask, switch to **Sepolia Testnet**, and ensure you have test ETH.
4. Choose **Injected Provider – MetaMask** in Remix and deploy the contract.
5. Verify that deployment succeeded without errors and the **Contract Address** is visible.









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

* Implementation Phase: Final Output (no error)

- The DAO smart contract was successfully deployed on Sepolia Testnet.
- The React frontend connected seamlessly to the contract using `ethers.js`.
- Users could create proposals by entering a title and description.
- Other wallet users could vote on active proposals (For/Against).
- After voting ended, the proposal status was updated to Executed once quorum was reached.
- The final output displayed the list of proposals, vote counts, and results dynamically on the frontend.
- All smart contract functions executed successfully without runtime or deployment errors.

Observation:

1. The DAO governance model allowed community-based decision making through smart contracts.
2. The integration of Ethers.js simplified the interaction between React frontend and Ethereum blockchain.
3. MetaMask provided secure authentication and transaction approval for voting.
4. Remix IDE was effective for writing, compiling, and deploying the DAO contract quickly.
5. The simulation demonstrated real-world DAO functionality**,** where proposals can be created, voted on, and executed transparently.
6. This experiment helped understand how blockchain enables decentralized governance without a central authority.

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:

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