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 : Web3 Connect – Contract Calls via Frontend

* Coding Phase: Pseudo Code / Flow Chart / Algorithm

- Open **Remix IDE** and create the SimpleStorage.sol smart contract.
- Compile the smart contract using the **Solidity compiler**.
- Copy the **ABI** generated after successful compilation.
- Deploy the contract on the **Sepolia Testnet** using **MetaMask (Injected Provider)**.
- Copy and save the **deployed contract address**.
- Create a new **React frontend project** using create-react-app.
- Configure the .env file with the **contract address** and **network details**.
- Install necessary packages:
  - ethers.js (for blockchain interaction)
  - web3.js (if required for compatibility)
- Import ABI and contract address to connect the frontend with the smart contract.
- Design the **UI in App.js** to allow storing and retrieving data through wallet connection.

Software used

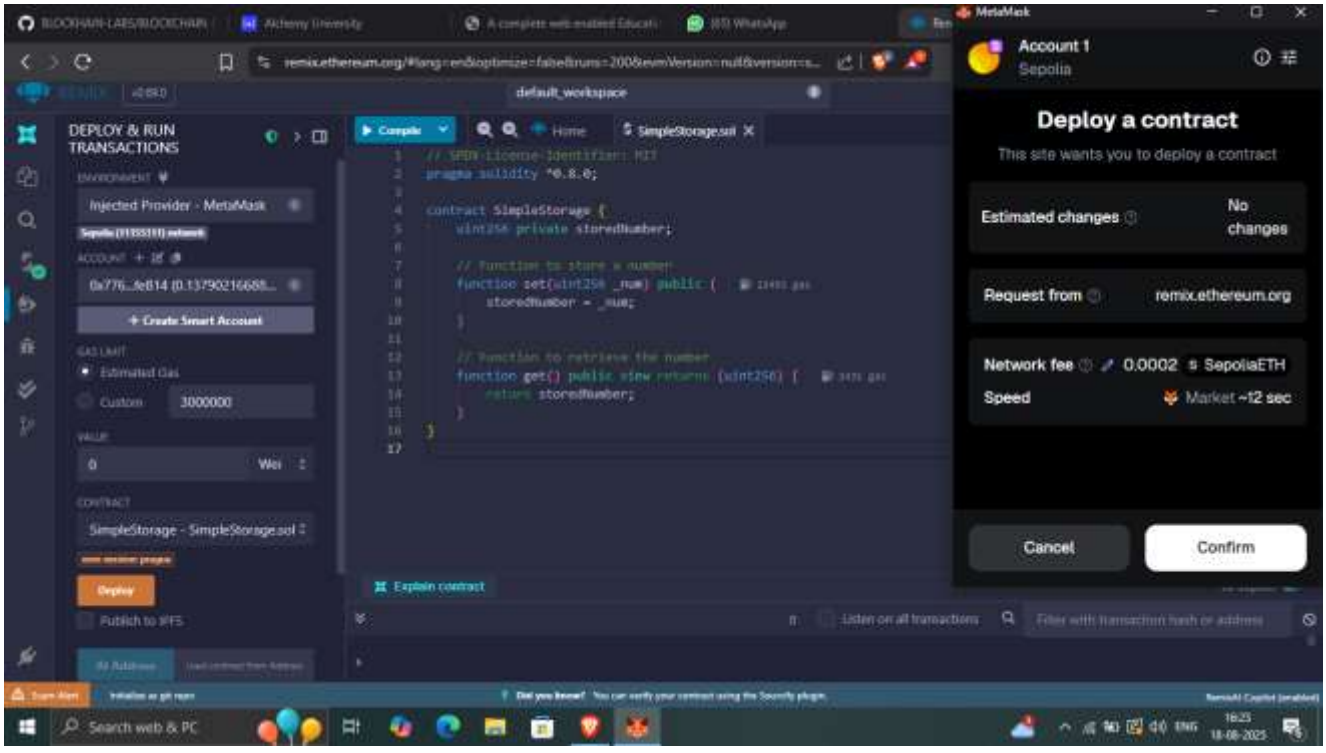
1. MetaMask Wallet
2. Remix IDE.
3. MS Word.
4. Brave for researching.

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

* Implementation Phase: Final Output (no error)

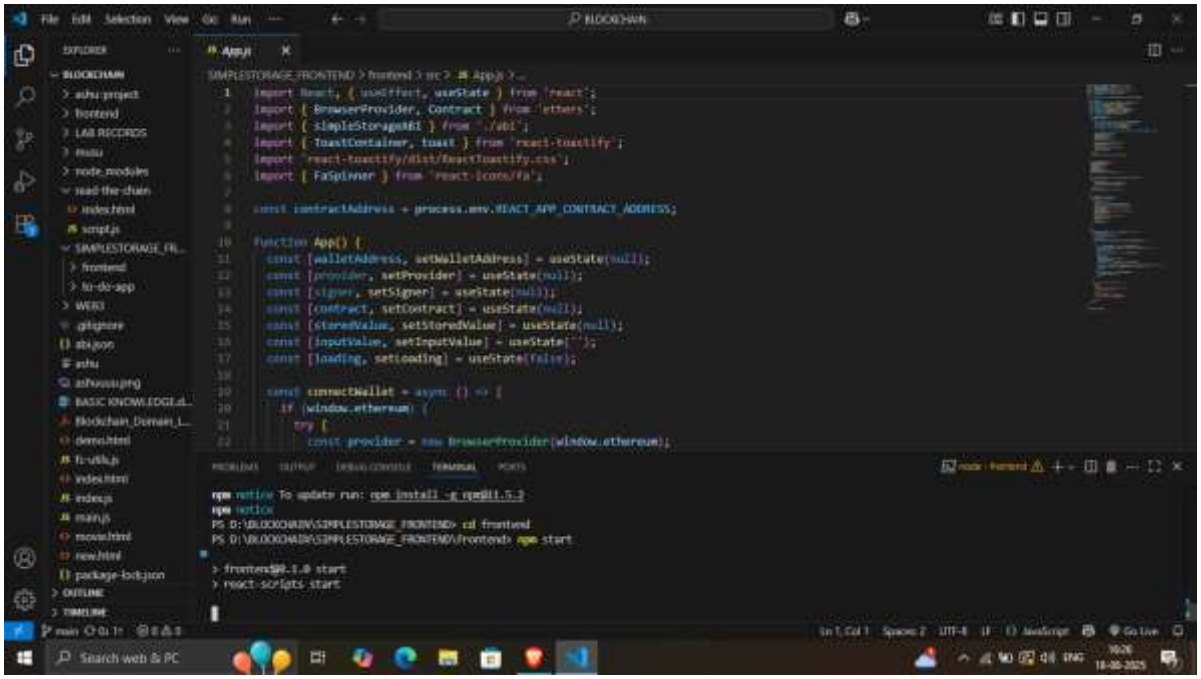
1. Firstly go to remix ide and write a smart contract on simplestorage.sol and compile it.
2. After compilation ABI will generated.
3. Then go to deploy and run transactions section and choose environment as injected provider-metamask, then simply deploy it.
4. Now we have to work on frontend first create a folder for your frontend then open terminal to install the react modules . Then create a ABI.js file inside your src folder where we have to store the abi of our smart contract and then create a .env file in the root of the project folder to dtore contract address and tectnet network.
5. Now in app.js write the frontend code and wallet connection code also.
6. Now simply move forward terminal of V.S Code just run the project with command npm start.

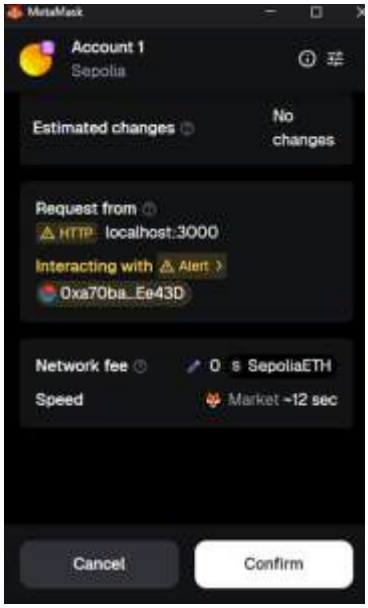
```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract SimpleStorage {
    uint256 private storedNumber;

    // Function to store a number
    function set(uint256 _num) public {    22492 gas
        storedNumber = _num;
    }

    // Function to retrieve the number
    function get() public view returns (uint256) {    2431 gas
        return storedNumber;
    }
}
```



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

* Implementation Phase: Final Output (no error)





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

* Observations:

- The smart contract was successfully deployed on the Sepolia Testnet.
- The frontend connected with the contract using ethers.js.
- Users were able to store and retrieve values through the UI.
- ethers.js provided a simpler and more secure interface compared to web3.js.
- The project demonstrated the complete flow of smart contract deployment, frontend integration, and blockchain interaction.

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.