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: Web3 Connect – Contract Calls via Frontend

*Coding Phase: Pseudo Code / Flow Chart / Algorithm

- Open Remix IDE and write the SimpleStorage.sol smart contract.
- Compile the smart contract using the Solidity compiler in Remix.
- Copy the generated ABI (Application Binary Interface) after successful compilation.
- Deploy the contract to the Sepolia Testnet using MetaMask.
- Copy the deployed contract address from Remix.
- Create a React frontend project using create-react-app.
- Add the contract address and network information to a .env file in the React project.
- Install Ethers.js to enable blockchain interaction
- Connect the frontend with the smart contract using the ABI and contract address.
- Design the UI in App.js, using web3.js to store and retrieve data from the blockchain.

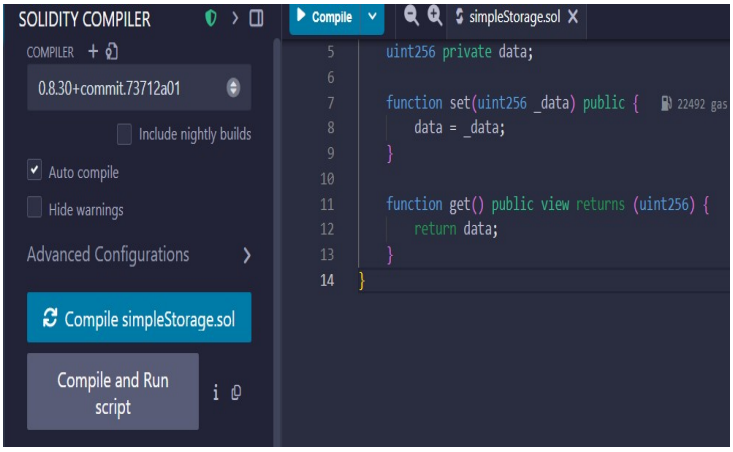
* Software used:

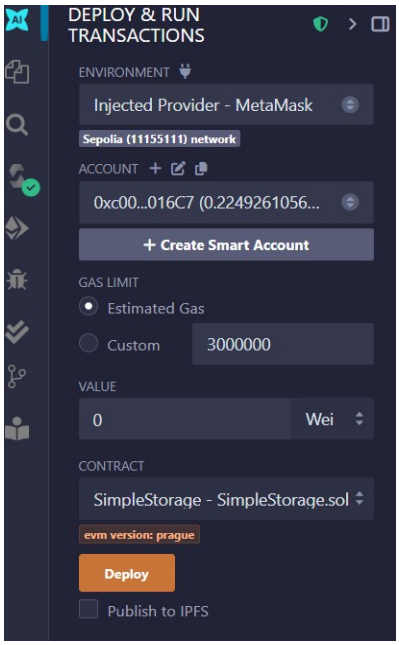
- Laptop
- Visual Studio Code (code editor)
- MetaMask Wallet (browser extension)
- Remix IDE (web-based smart contract IDE)
- Node.js
- React (via create-react-app)
- Ether.js (Ethereum JavaScript library)
- dotenv (for environment variables)

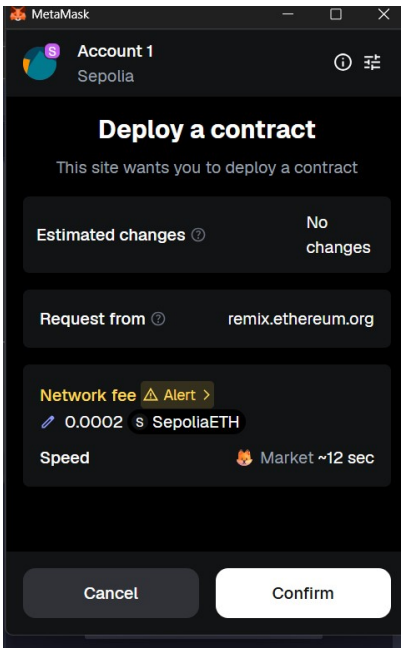
**** As applicable according to the experiment.***
Two sheets per experiment (10-20) to be used.

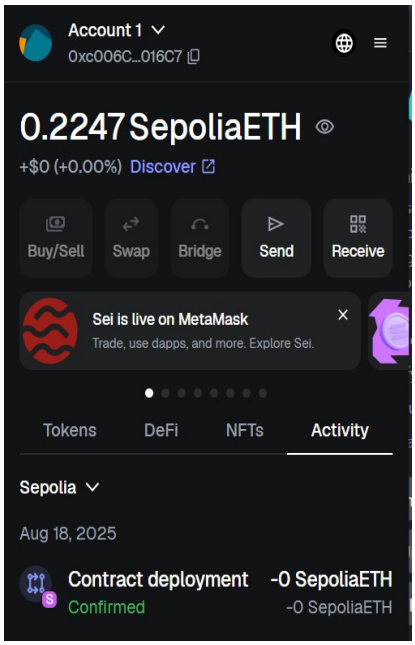
* Testing Phase: Compilation of Code (error detection)

- First we have to go Remix IDE and create a .sol file named as simpleStorage.sol and write our smart contract.
- Then we need to compile our smart contract and copy the generated ABI
- After successful compilation deploy the smart contract and choose the environment to Injected Provider - MetaMask
- After deployment under Deployed Contracts section copy the contract address for future use.
- Then using web3.js library we create frontend and interact with our wallet.





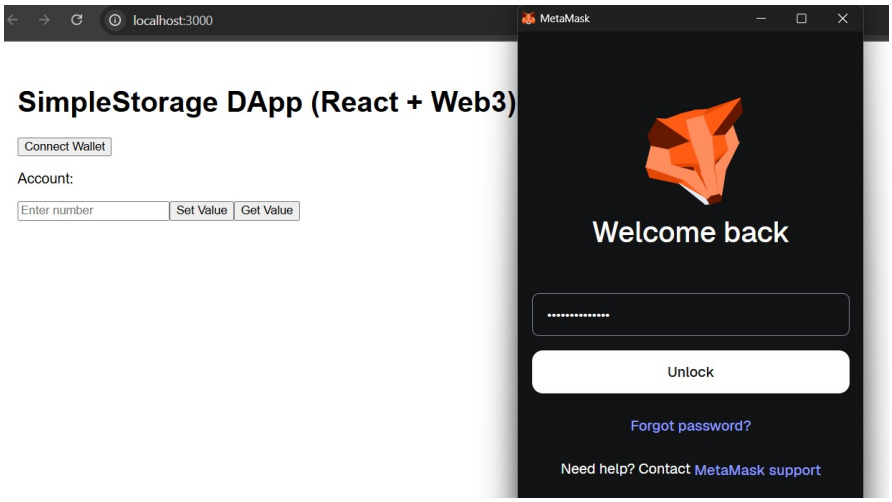


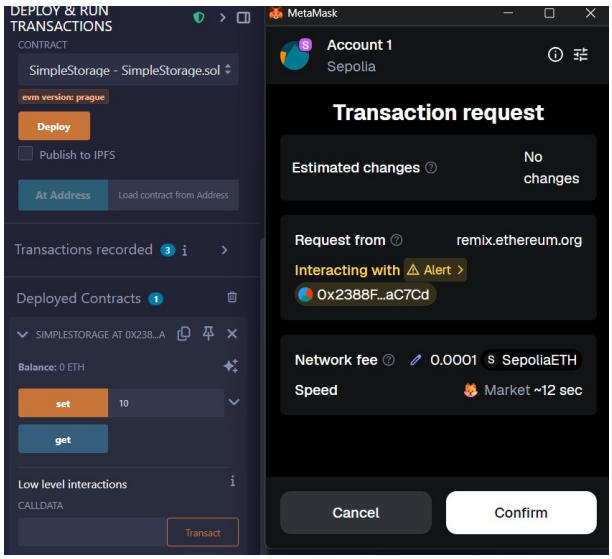


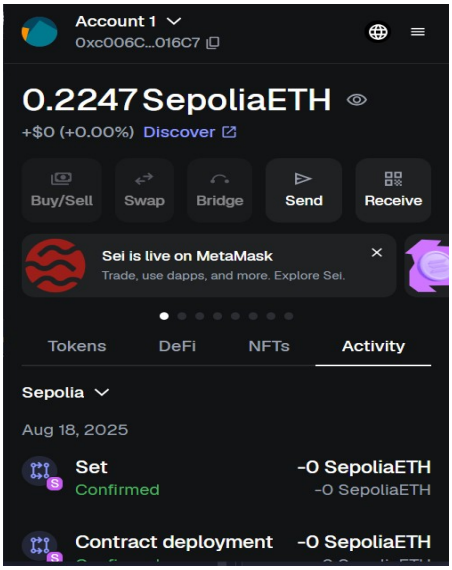
* Implementation Phase: Final Output (no error)

- Now we have to create a folder named as “frontend” and open the terminal and move to the current frontend directory.
- Inside frontend we have to create a ‘.env’ file where we will store our contract address.
- In the frontend/src/ folder we have to create a ABI.json file to store our contract ABI.
- Now in the App.js file we have to write our frontend code and wallet connection function.
- In the terminal install and the required packages from node package manager.
- Then run the terminal with the command ‘npm start’.
- Then we can interact with the UI such as connecting to wallet and set and get functions.

* Implementation Phase: Final Output (no error)







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

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 :

Signature of the Faculty :

Regn. No. :

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