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: Frontend Connect – Web3.js Integration

*Coding Phase: Pseudo Code / Flow Chart / Algorithm

- Open Remix IDE and write the SimpleStorage.sol smart contract.
- Compile the 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 details to a .env file in the React project.
- Install web3.js to enable interaction with the blockchain.
- Connect the frontend to the smart contract using the ABI and contract address.
- Design the user interface in App.js and use web3.js functions to store and retrieve data from the blockchain.

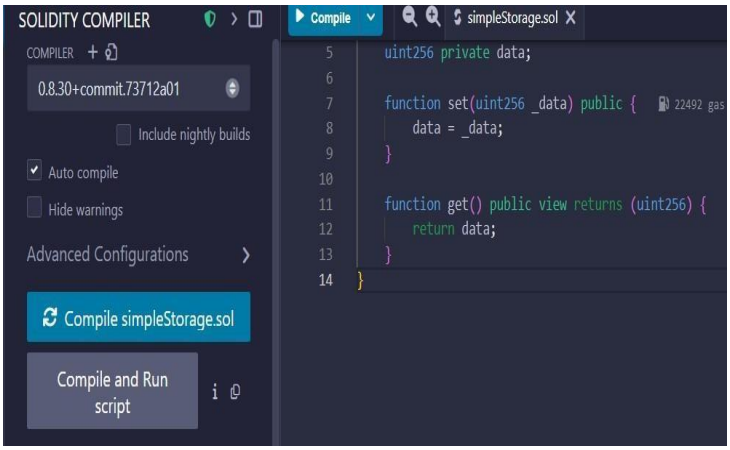
* Software used:

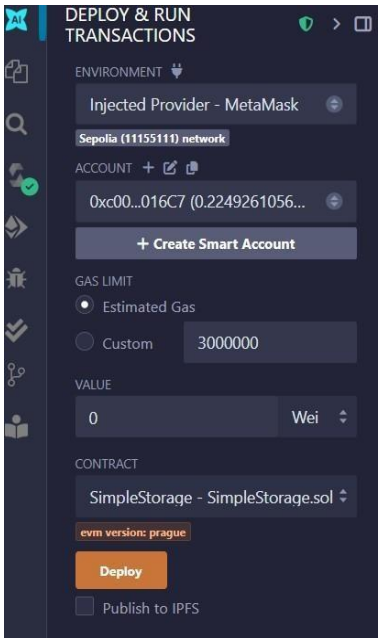
- Laptop
- Visual Studio Code (code editor)
- MetaMask wallet (browser extension)
- Remix IDE (web-based smart contract development tool)
- Node.js
- React (created using create-react-app)
- Web3.js (Ethereum JavaScript interaction library)
- dotenv (for managing environment variables)

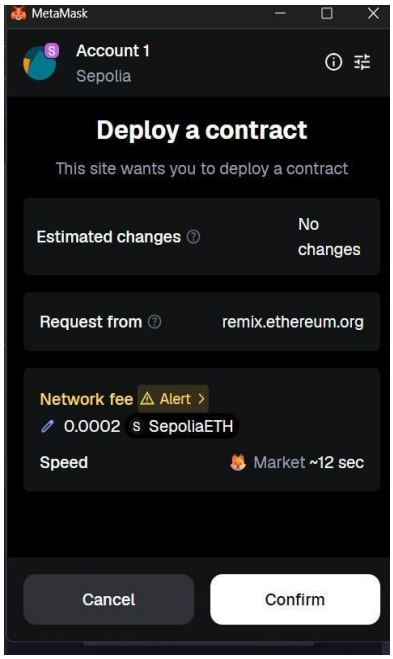
* *As applicable according to the experiment. Two sheets per experiment (10-20) to be used.*

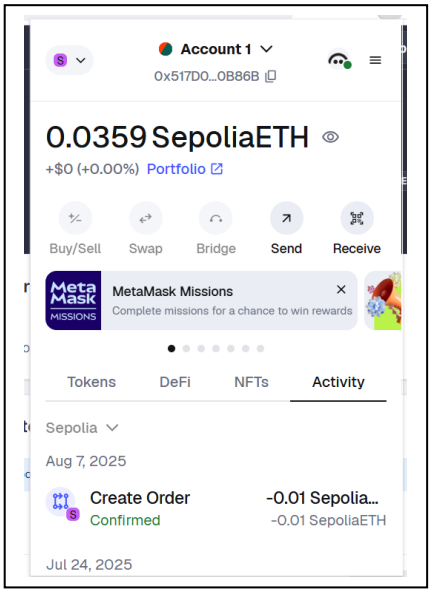
* Testing Phase: Compilation of Code (error detection)

- Open Remix IDE and create a `.sol` file named `SimpleStorage.sol`, then write the smart contract.
- Compile the smart contract and copy the generated ABI.
- After successful compilation, deploy the contract and select **Injected Provider – MetaMask** as the environment.
- Once deployed, copy the contract address from the **Deployed Contracts** section for later use.
- Use the **web3.js** library to build a frontend interface and interact with the deployed contract through your MetaMask wallet.







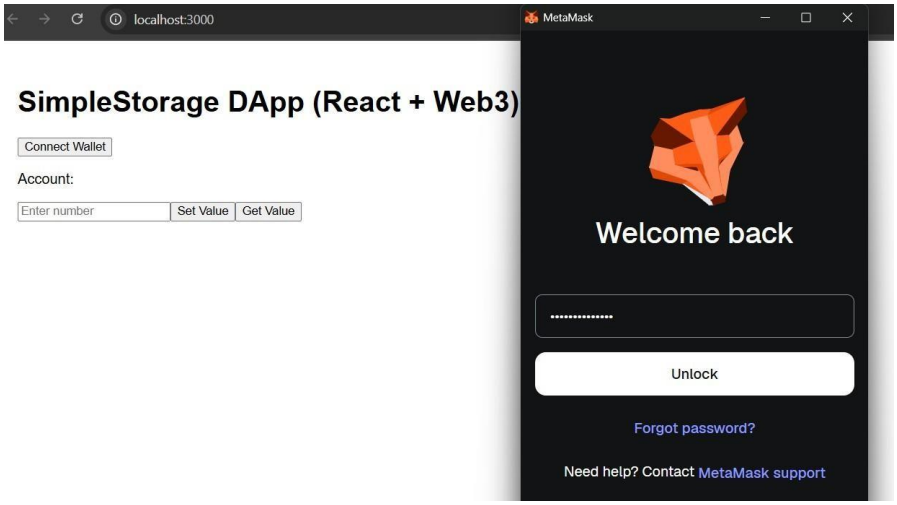


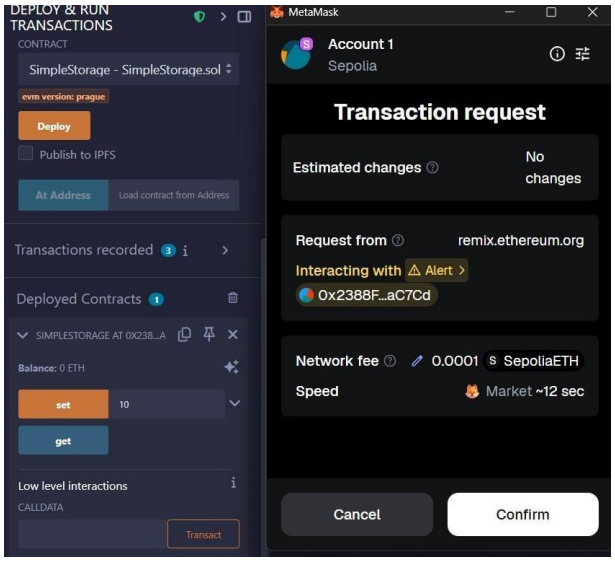
* Implementation Phase: Final Output (no error)

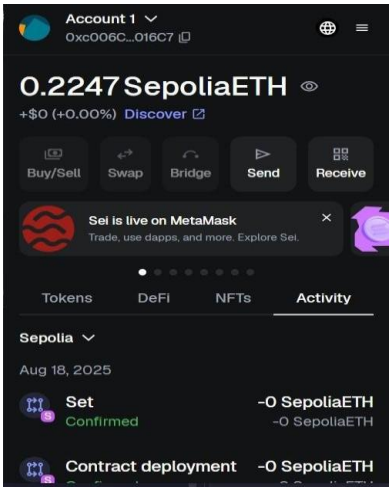
- Create a folder named **frontend**, open the terminal, and navigate to the frontend directory.
- Inside the frontend folder, create a **.env** file to store the contract address.
- In the **frontend/src/** directory, create an **ABI.json** file to store the contract ABI.
- Open the **App.js** file and write the frontend code, including the wallet connection function.
- In the terminal, install all required packages using the Node Package Manager (npm).

- Run the project using the command **npm start**.
- Once the app starts, interact with the user interface to connect the wallet and use the **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***