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 : Helllo Solidity- Writting First Smart Contract

*Coding Phase: Pseudo Code / Flow Chart / Algorithm:

Algorithm:

- **Start** the process by opening the Remix Ethereum IDE in a web browser.
- **Create a new Solidity file** (e.g., Hello.sol) and define the license identifier and Solidity version.
- **Write the smart contract** code with a variable and a function (e.g., a function that returns the message “Hello, Solidity!”).
- **Compile** the smart contract using the Remix compiler to check for any syntax errors.
- **Deploy** the compiled contract on the Ethereum JavaScript VM or test network using MetaMask.
- **Execute the function** from the deployed contract to verify the output message.
- **End** the process after confirming successful deployment and correct output display.

*Software Used:

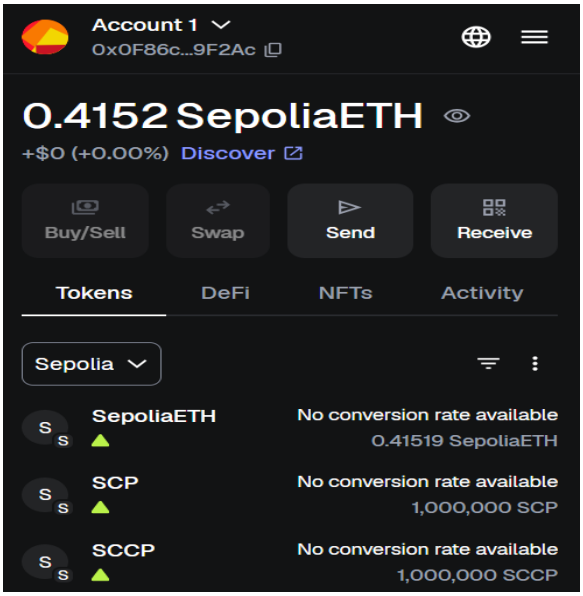
1. Remix IDE
2. MetaMask Wallet
3. Ethereum Sepolia Faucet
4. Chrome Web Browser

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

*Testing/Implementation Phase:

First we have to download the MetaMask and add the extension on the web browser where we are going to use Remix IDE. Then go to ethereum cloud to receive faucet.

Give you wallet address in ethereum sepolia faucet to receive 0.05 faucet.



Open Remix IDE on the same browser and create a solidity file named as SimpleStorage.sol and write the code which we have to deploy.

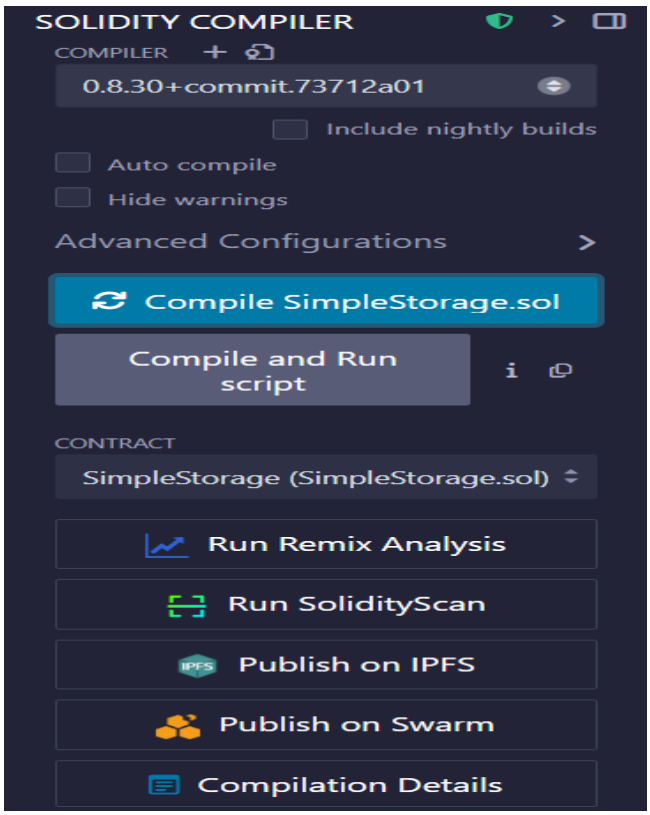
```
//SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;
contract SimpleStorage {
    uint public storedData;

    constructor(uint _data) {
        storedData = _data;
    }
    function set(uint x) public {
        storedData = x;
    }
    function get() public view returns (uint) {
        return storedData;
    }
}
```


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

*Testing/Implementation Phase:

In code we have to create two functions named as get() and set(). Get function to get the data and set function to set the data after writting thw smart contract go to the environment and choose Injected Provider-MetaMask and we can see there are autogenerate of your MetaMask Wallet address with the test balance.

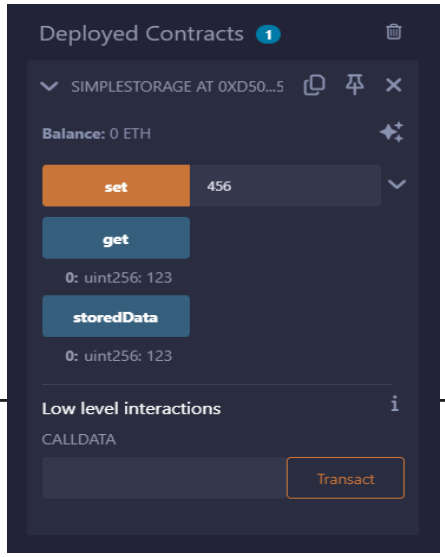
Then click on compile after clicking our file is successfully compiled.



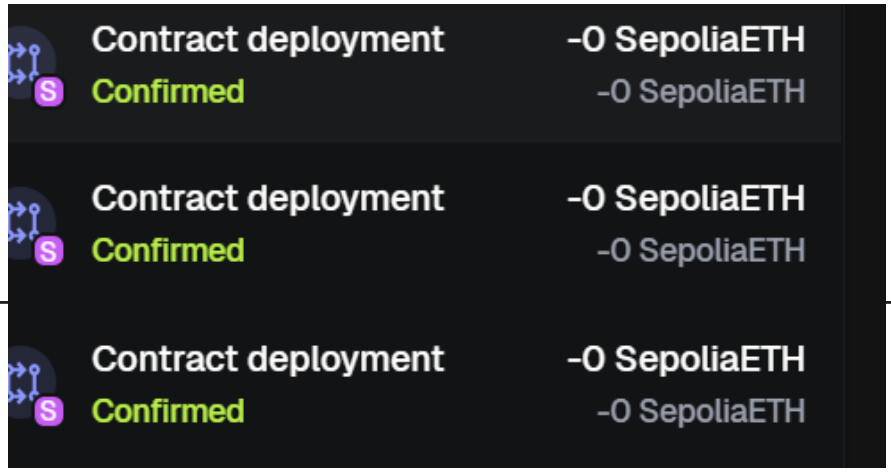
*As applicable according to the experiment.
Two sheets per experiment (10-20) to be used.

Implementation Phase: Final Output (no error)

Then click deploy



Now our smart contract is successfully connected to our wallet address.



Observation:

1. The smart contract was successfully compiled and deployed on the Ethereum testnet using Remix and MetaMask.
2. MetaMask handled the transaction and confirmed it on the blockchain.

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.*