St. Francis Institute of Technology, Mumbai-400 103
Department Of Information Technology

A.Y. 2025-2026
Class: BE-ITA/B, Semester: VIII
Subject: BlockChain Lab

Experiment 4

1. **Aim:** To implement smart contract using Remix IDE and deploy it using Ganache and MetaMask.

2. **Objective:** To …
   - explore working of Ganache ethernet network.
   - test a smart contract by using Remix IDE

3. Lab outcome: After performing the experiment, the students will be able to **implement** smart contracts in Ethereum using different development frameworks (PO3, PSO2, BL3)

4. Prerequisite:
- Fundamental knowledge of blockchain
- Knowledge of the Ethereum platform and Remix IDE
- Familiarity with the Solidity programming language

5. Requirements: The following are the requirements –
Remix IDE, Ganache Provider, MetaMask wallet etc.

6. Pre-Experiment Theory:

What is Ganache?

Ganache is a private Ethereum blockchain development environment that allows to you emulate the Ethereum blockchain so that you can interact with smart contracts in your own private blockchain. You can use Ganache across the entire development cycle; enabling you to develop, deploy, and test your dApps (decentralized Applications) in a safe and deterministic environment.

7. Laboratory Exercise

A. Steps to be implemented.

To Follow the procedure given below to build smart contract in Remix IDE

- Download and install Ganache from https://trufflframework.com/ganache.
- Download and install the official MetaMask extension/ plugin.
- Open Remix IDE in Google Chrome.
- Write smart contract by creating new file under contract folder, with .sol extension.
- Write your contract code using solidity language.
- Click on compiler icon, choose compiler version or keep default and click on compile button.
- For deployment…
  Part A) First start Ganache provider and then in Remix IDE, choose environment Ganache Provider and then deploy the said contract.
  Part B) Set-up Ganache with MetaMask and then in Remix IDE choose environment Injected Provider- MetaMask and then deploy the said contract.
- Under deployed contract, get the output of your contract.

B. Program Code

Write a smart contract StudentMarks.sol to create student database with attributes, id, name, dept, sub1marks, sub2Marks etc. and deploy it using Ganache and MetaMask.

8. Post Experimental Exercise-

A. Questions:

1. List down the details of Ganache IDE
2. Create any smart contract that you have not done so far and deploy it.

B. Results/Observations/Program output:

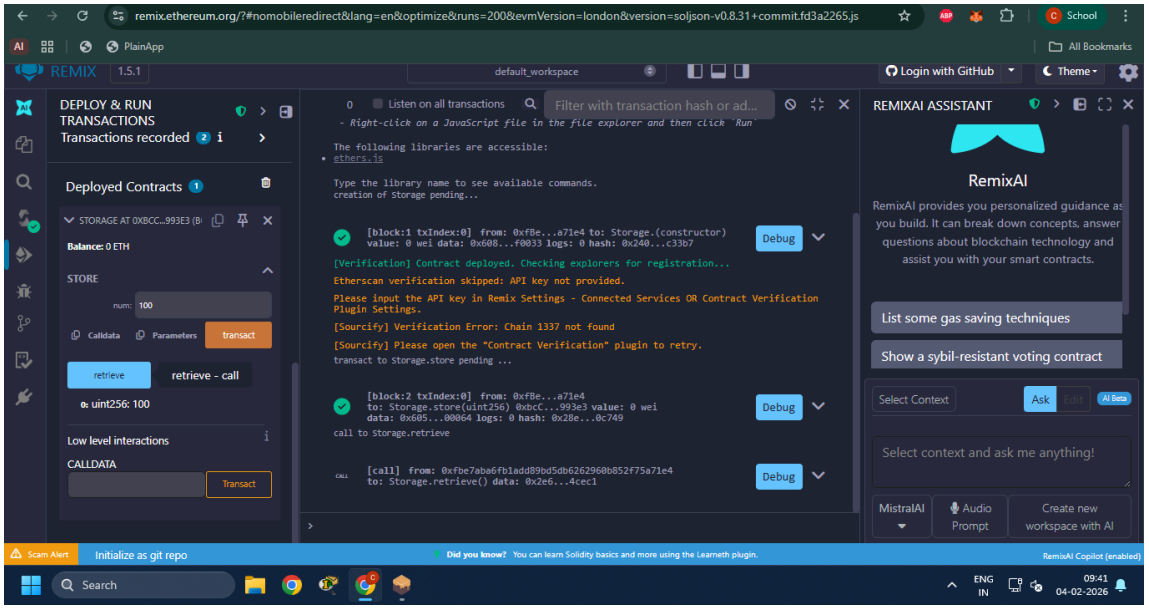
Present the program input/output results if any and comment on the same.

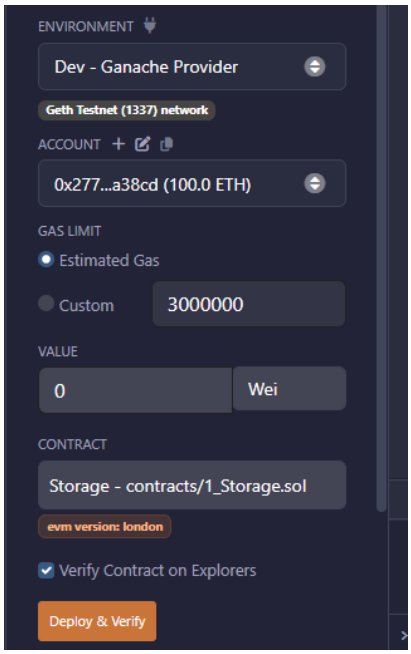
C. Conclusion:

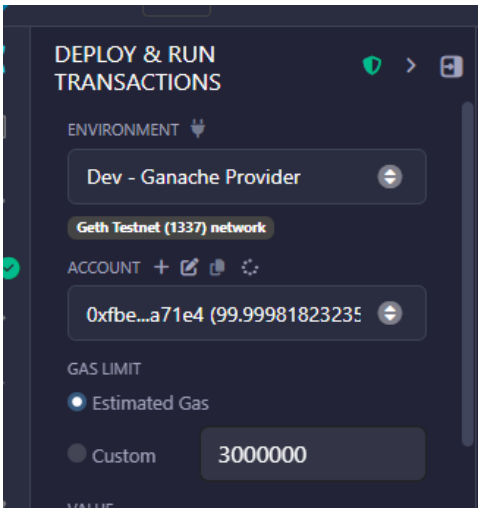
Write what was performed in the experiment.
Write which tools you used to perform the experiment
Write what you inferred from the output obtained.

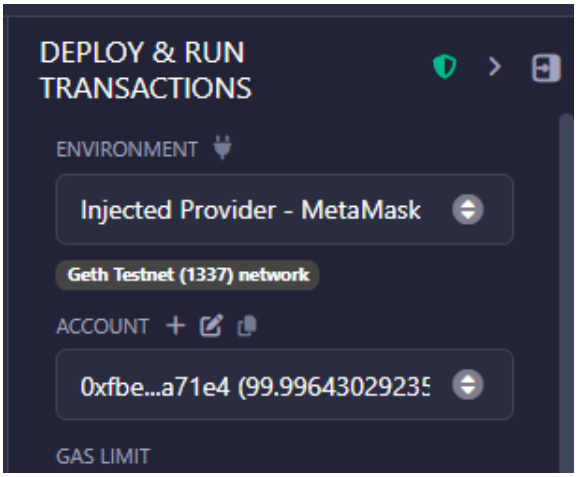
D. References:

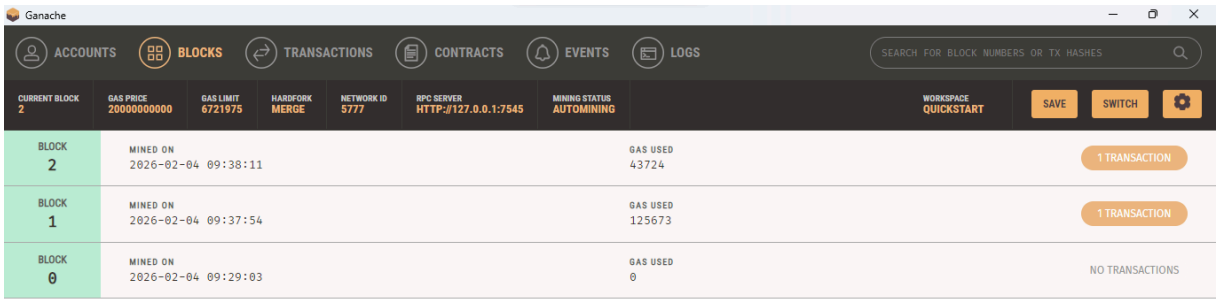
https://www.geeksforgeeks.org/how-to-use-metamask-to-deploy-a-smart-contract-in-solidity-blockchain/
https://www.geeksforgeeks.org/how-to-set-up-ganche-with-metamask/
Mastering Ethereum, Building Smart Contract and Dapps, Andreas M. Antonopoulos Dr. Gavin Wood, O'reilly

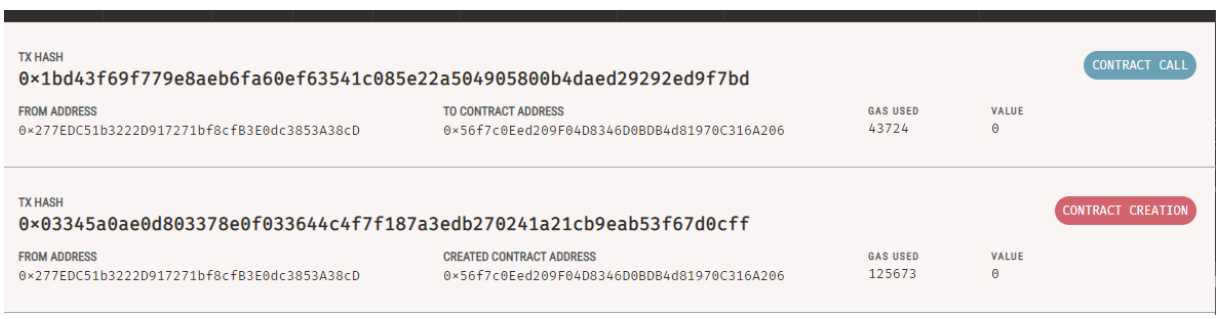

DEPLOY & RUN TRANSACTIONS
Transactions recorded 2
Deployed Contracts 1
STORE
num: 100
transact
retrieve
retrieve - call
0: uint256: 100
Low level interactions
CALLDATA
transact
REMIXAI ASSISTANT
RemixAI
RemixAI provides you personalized guidance as you build. It can break down concepts, answer questions about blockchain technology and assist you with your smart contracts.
List some gas saving techniques
Show a sybil-resistant voting contract
Select Context
Ask
Edit
Select context and ask me anything!
MistralAI
Audio Prompt
Create new workspace with AI
Initialize as git repo


ENVIRONMENT
Dev - Ganache Provider
Geth Testnet (1337) network
ACCOUNT
0x277...a38cd (100.0 ETH)
GAS LIMIT
Estimated Gas
Custom
3000000
VALUE
0
Wei
CONTRACT
Storage - contracts/1_Storage.sol
evm version: london
Verify Contract on Explorers
Deploy & Verify

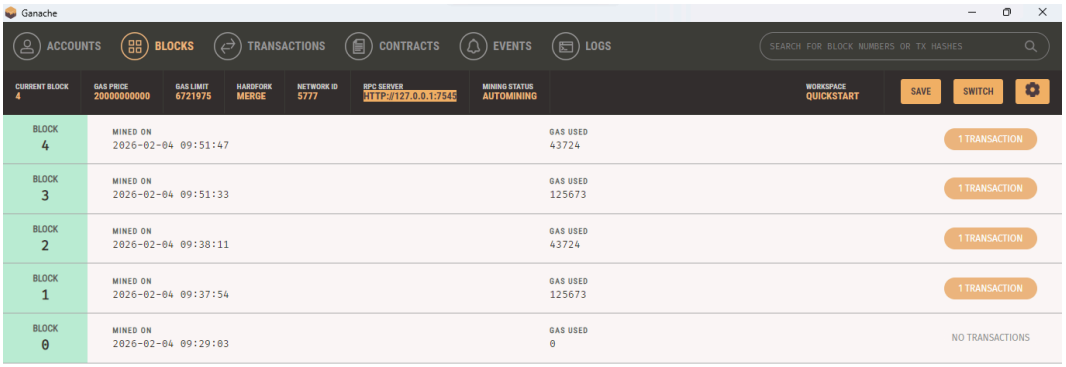

DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Dev - Ganache Provider
Geth Testnet (1337) network
ACCOUNT
0xfbe...a71e4 (99.99981823235
GAS LIMIT
Estimated Gas
Custom
3000000

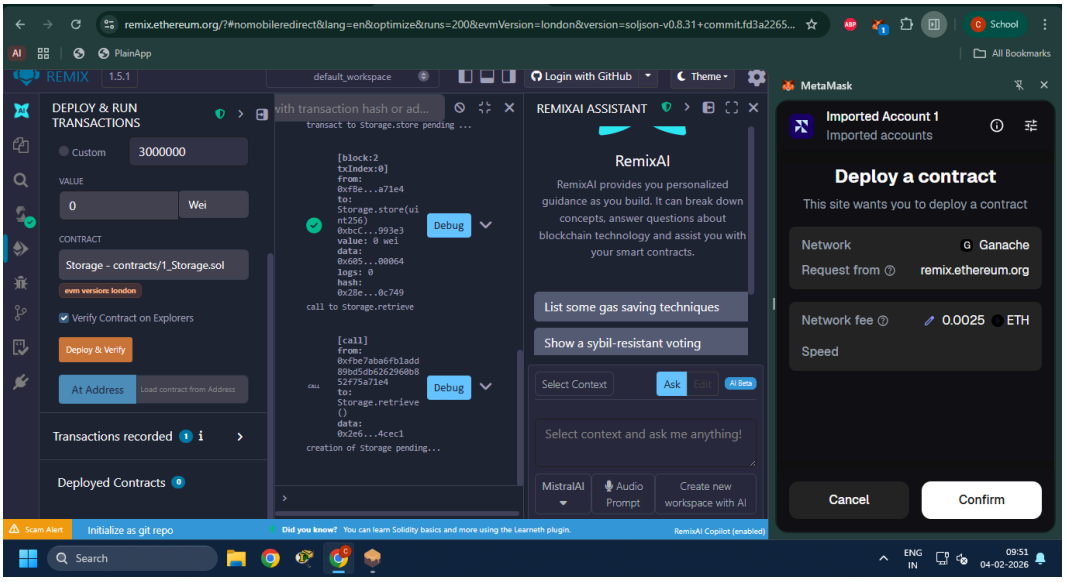

DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Provider - MetaMask
Geth Testnet (1337) network
ACCOUNT
0xfbe...a71e4 (99.9964302923
GAS LIMIT

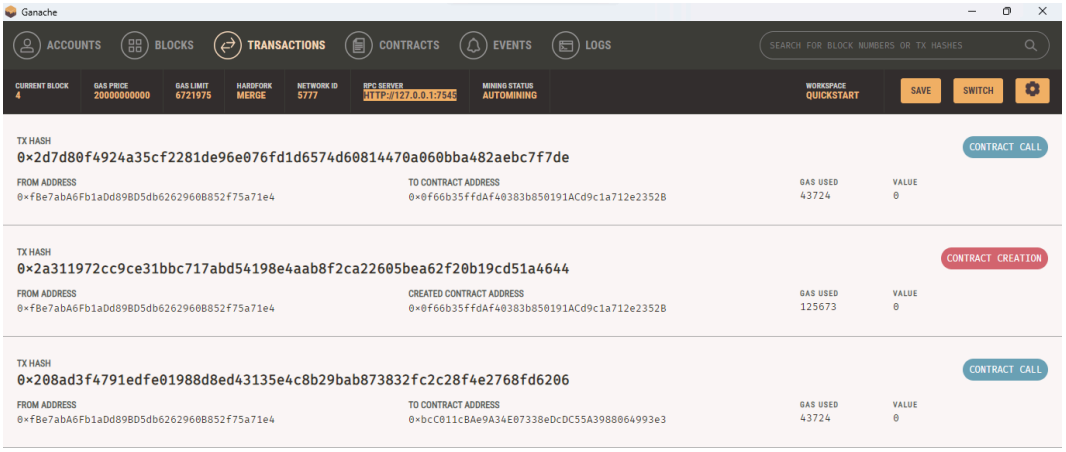

Ganache
ACCOUNTS
BLOCKS
TRANSACTIONS
CONTRACTS
EVENTS
LOGS
SEARCH FOR BLOCK NUMBERS OR TX HASHES
CURRENT BLOCK 2
GAS PRICE 20000000000
GAS LIMIT 6721975
HARDFORK MERGE
NETWORK ID 5777
RPC SERVER HTTP://127.0.0.1:7545
MINING STATUS AUTOMINING
WORKSPACE QUICKSTART
SAVE
SWITCH
BLOCK 2
MINED ON 2026-02-04 09:38:11
GAS USED 43724
1 TRANSACTION
BLOCK 1
MINED ON 2026-02-04 09:37:54
GAS USED 125673
1 TRANSACTION
BLOCK 0
MINED ON 2026-02-04 09:29:03
GAS USED 0
NO TRANSACTIONS


TX HASH
0x1bd43f69f779e8aeb6fa60ef63541c085e22a504905800b4daed29292ed9f7bd
CONTRACT CALL
FROM ADDRESS
0x277EDC51b3222D917271bf8cfB3E0dc3853A38cD
TO CONTRACT ADDRESS
0x56f7c0Eed209F04D8346D0BDB4d81970C316A206
GAS USED
43724
VALUE
0
TX HASH
0x03345a0ae0d803378e0f033644c4f7f187a3edb270241a21cb9eab53f67d0cff
CONTRACT CREATION
FROM ADDRESS
0x277EDC51b3222D917271bf8cfB3E0dc3853A38cD
CREATED CONTRACT ADDRESS
0x56f7c0Eed209F04D8346D0BDB4d81970C316A206
GAS USED
125673
VALUE
0

Metamask-ganashe setup







Program Code-

Write a smart contract StudentMarks.sol to create student database with attributes, id,

name, dept, sub1marks, sub2Marks etc

Code-

/ SPDX-License-Identifier: MIT	uint256 _sub1Marks,

```
pragma solidity ^0.8.0;
contract StudentMarks {
    struct Student {
        uint256 id;
        string name;
        string dept;
        uint256 sub1Marks;
        uint256 sub2Marks;
    }
    Student[] private students;
    mapping(uint256 => Student) private studentById;

    function addStudent(
        uint256 _id,
        string memory _name,
        string memory _dept,
        uint256 _sub1Marks,
        uint256 _sub2Marks
    ) public {
        require(_sub1Marks <= 100 && _sub2Marks <= 100, "Marks cannot exceed 100");
        Student memory newStudent = Student({
            id: _id,
            name: _name,
            dept: _dept,
            sub1Marks: _sub1Marks,
            sub2Marks: _sub2Marks
        });
        students.push(newStudent);
        studentById[_id] = newStudent;
    }
    function getStudent(uint256 _id) public view returns (Student memory) {
        return studentById[_id];
    }
}
```