Experiment No.8
To create a private ethereum blockchain using Ganache and Truffle
Date of Performance:5/10/23
Date of Submission:10/10/23

AIM: To create a private ethereum blockchain using Ganache and Truffle

Objective: To create private ethereum blockchain and deploy smart contract on it

Theory:

Smart contracts are programs stored on a blockchain. The term 'smart contract' was first coined by Nick Szabo in 1994. It's a collection of code (its functions) and data (its state) that resides at a specific address on the Ethereum blockchain. One of the main features is that they are immutable once deployed on the blockchain. Solidity and Vyper are the two most active programming languages used to write smart contracts on the Ethereum blockchain with Solidity being the top choice for most developers as it is an object-oriented, statically-typed language and is strongly influenced by more popular OOP languages like JavaScript and C++.

Ganache

Ganache is a personalized blockchain for Ethereum development. It can be used to run tests, execute commands, and inspect states while controlling how the chain operates. Ganache is an Ethereum simulator that makes developing Ethereum applications faster, easier, and safer. It is provided by Truffle Suite and can be downloaded from https://www.trufflesuite.com/ganache. The below image shows the view of Ganache

Fig.8.1 Ganache view

Truffle

Truffle is the most popular development framework for Ethereum. Truffle takes care of managing your contract artifacts so you don't have to. Includes support for custom deployments, library linking and complex Ethereum applications. Truffle is used to test contracts in both javascript and solidity.

Process:

Step 1. Install Ganache , Download Ganache from the website https://truffleframework.com/ganache
Step 2. Install Truffle by executing the following command at the command prompt: npm install –g truffle
To install Truffle you need to have Node and NPM along with Python setup on your machine.

Step 3. To verify if Truffle is installed successfully, execute the following command at the command prompt.
truffle version
Step 4. To start a project in Truffle, go into a directory and type the init command: truffle init
This will create a new project with the required directory: contracts, migrations, test
Step 5. Create the required contract file in solidity and save it in the contracts directory
Step 6. Create a migration file in javascript and save it in migrations directory

Step 7. Run the following command to deploy the contract on Ganache truffle Test Ganache should be running while executing this command. The contract deployed will be visible in the contracts tab of the Ganache.

Code:
Election.sol:

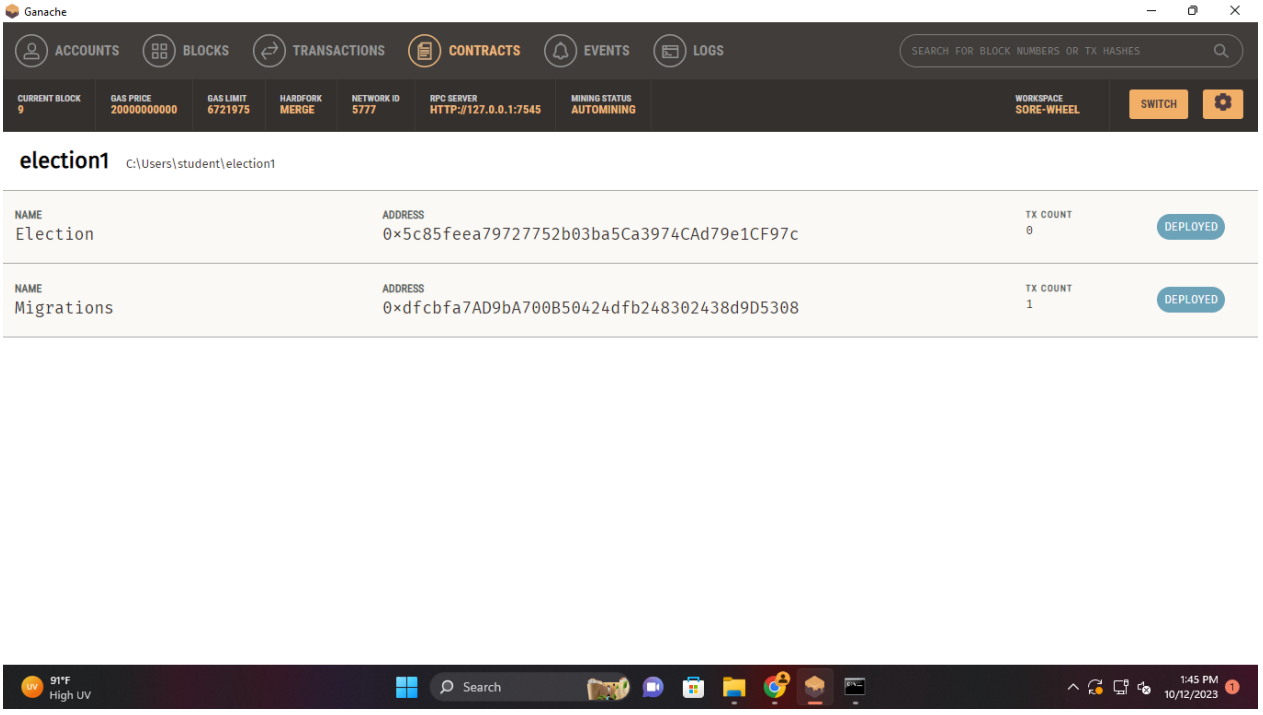
```
pragma solidity ^0.5.16;



contract Election {
    // Model a Candidate
    struct Candidate {
        uint id;
        string name;
        uint voteCount;
    }

    // Store accounts that have voted
    mapping(address => bool) public voters;
    // Read/write candidates
    mapping(uint => Candidate) public candidates;
    // Store Candidates Count
    uint public candidatesCount;

    constructor () public {
        addCandidate("Candidate 1");
        addCandidate("Candidate 2");
    }

    function addCandidate (string memory _name) private {
        candidatesCount ++;
        candidates[candidatesCount] = Candidate(candidatesCount, _name, 0);
    }

    function vote (uint _candidateId) public {
        // require that they haven't voted before
        require(!voters[msg.sender]);

        // require a valid candidate
        require(_candidateId > 0 && _candidateId <= candidatesCount);
```

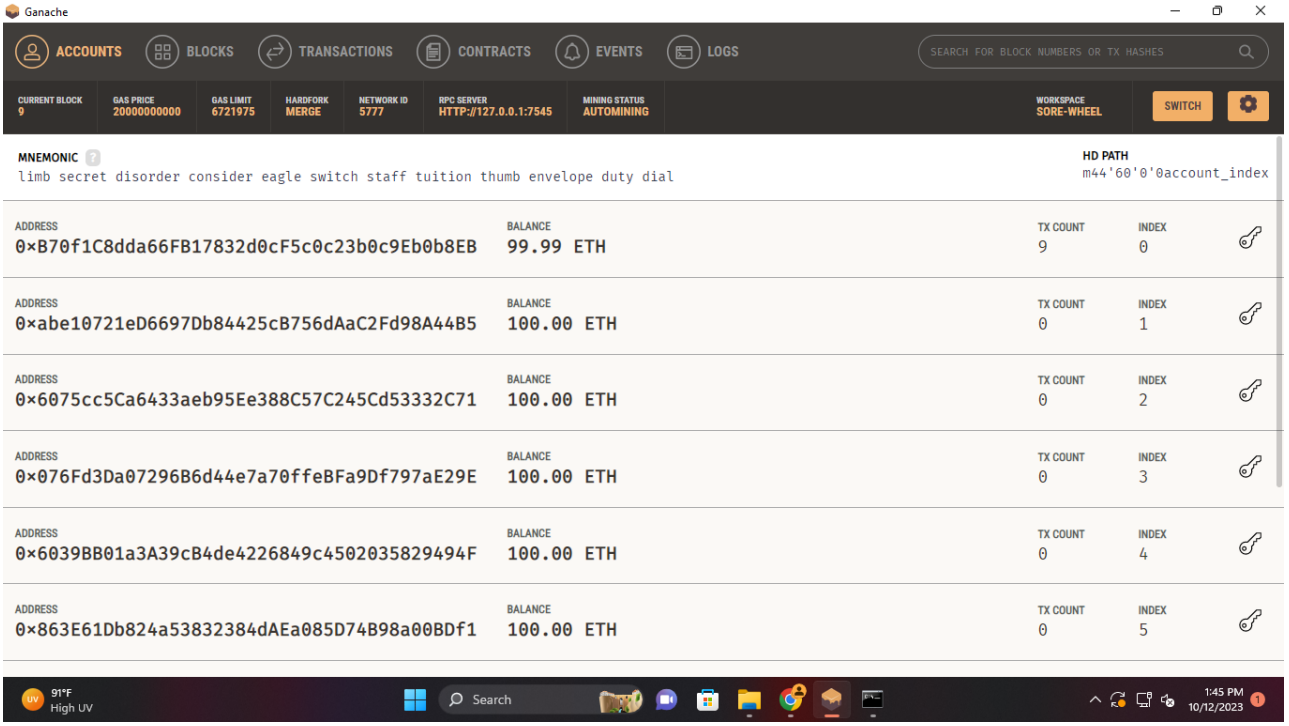
```
        // record that voter has voted
        voters[msg.sender] = true;

        // update candidate vote Count
        candidates[_candidateId].voteCount ++;
    }
}
```

2_deploy_contracts.js:

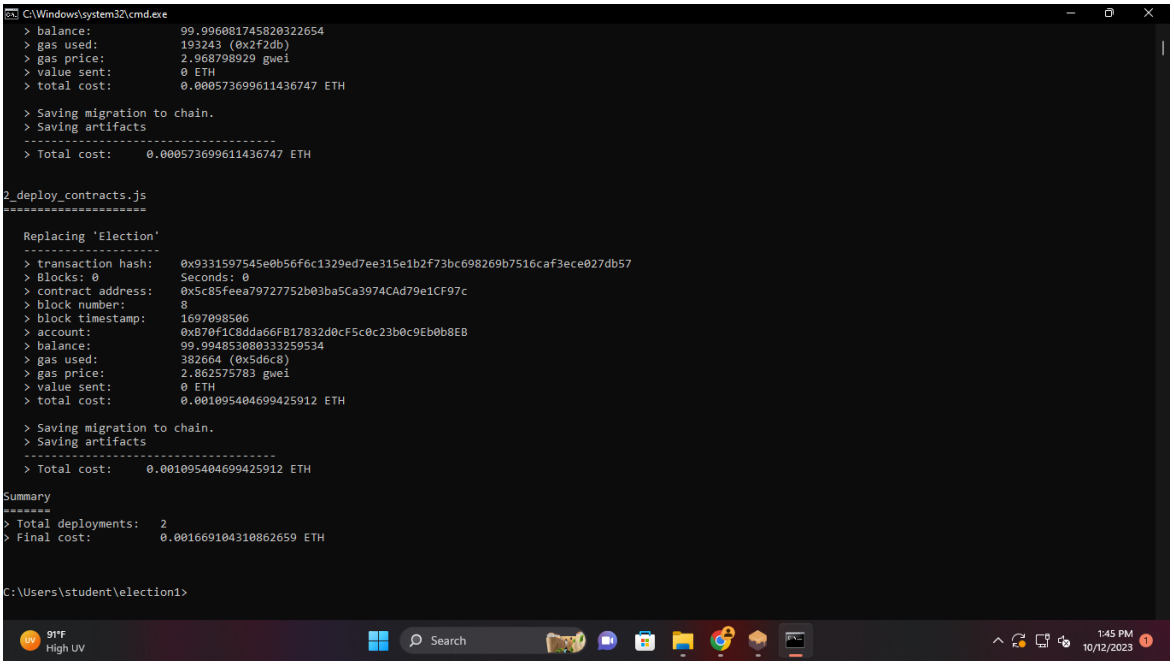
```
var Election = artifacts.require("./Election.sol");

module.exports = function(deployer) {
  deployer.deploy(Election);
};
```

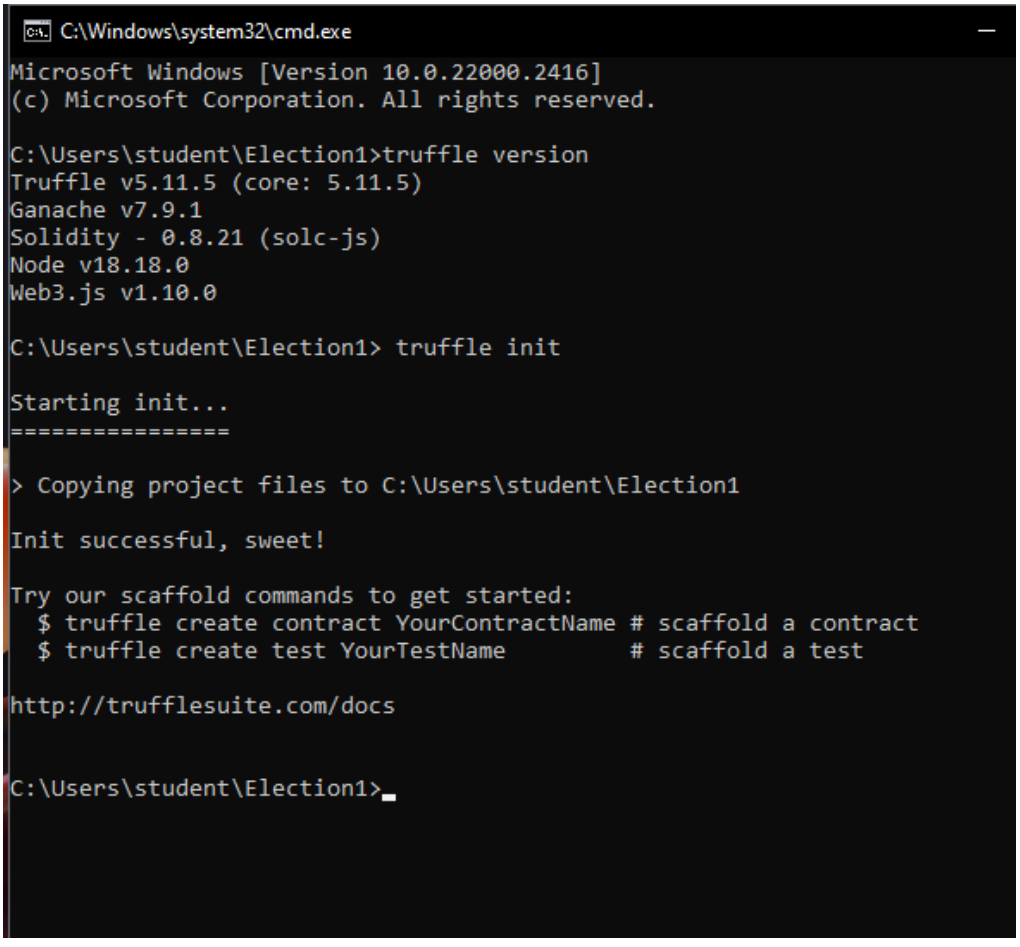
Output:






```
C:\Windows\system32\cmd.exe
  0 passing (0ms)


C:\Users\student\election1>truffle migrate

Compiling your contracts...
===========================
> Compiling .\contracts\Election.sol
> Compiling .\contracts\Migrations.sol
> Artifacts written to C:\Users\student\election1\build\contracts
> Compiled successfully using:
   - solc: 0.5.16+commit.9c3226ce.Emscripten.clang


Starting migrations...
======================
> Network name:    'development'
> Network id:      5777
> Block gas limit: 6721975 (0x6691b7)


1_initial_migration.js
======================

   Replacing 'Migrations'
   ----------------------
   > transaction hash:    0x125d44bdf21bd24a015f48ab4602cc58ecfdd23443399c846d36df6d47f7e619
   > Blocks: 0            Seconds: 0
   > contract address:    0xdfcbfa7AD9bA700B50424dfb248302438d9D5308
   > block number:        6
   > block timestamp:     1697098506
   > account:             0xB70f1C8dda66FB17832d0cF5c0c23b0c9Eb0b8EB
   > balance:             99.996081745820322654
   > gas used:            193243 (0x2f2db)
   > gas price:           2.968798929 gwei
   > value sent:          0 ETH
   > total cost:          0.000573699611436747 ETH

   > Saving migration to chain.
   > Saving artifacts
   -------------------------------------
   > Total cost:     0.000573699611436747 ETH
```


C:\Windows\system32\cmd.exe
> balance: 99.996081745820322654
> gas used: 193243 (0x2f2db)
> gas price: 2.968798929 gwei
> value sent: 0 ETH
> total cost: 0.000573699611436747 ETH
> Saving migration to chain.
> Saving artifacts
> Total cost: 0.000573699611436747 ETH
2_deploy_contracts.js
Replacing 'Election'
> transaction hash: 0x9331597545e0b56f6c1329ed7ee315e1b2f73bc698269b7516caf3ece027db57
> Blocks: 0 Seconds: 0
> contract address: 0x5c85feea79727752b03ba5Ca3974CAd79e1CF97c
> block number: 8
> block timestamp: 1697098506
> account: 0xB70f1C8dda66FB17832d0cF5c0c23b0c9Eb0b8EB
> balance: 99.994853080333259534
> gas used: 382664 (0x5d6c8)
> gas price: 2.862575783 gwei
> value sent: 0 ETH
> total cost: 0.001095404699425912 ETH
> Saving migration to chain.
> Saving artifacts
> Total cost: 0.001095404699425912 ETH
Summary
> Total deployments: 2
> Final cost: 0.001669104310862659 ETH
C:\Users\student\election1>
91°F
High UV
Search
1:45 PM
10/12/2023


C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.22000.2416]
(c) Microsoft Corporation. All rights reserved.
C:\Users\student\Election1>truffle version
Truffle v5.11.5 (core: 5.11.5)
Ganache v7.9.1
Solidity - 0.8.21 (solc-js)
Node v18.18.0
Web3.js v1.10.0
C:\Users\student\Election1> truffle init
Starting init...
> Copying project files to C:\Users\student\Election1
Init successful, sweet!
Try our scaffold commands to get started:
$ truffle create contract YourContractName # scaffold a contract
$ truffle create test YourTestName # scaffold a test
http://trufflesuite.com/docs
C:\Users\student\Election1>

Conclusion: Justify the importance of the Ganache and Truffle in creating a personal ethererum blockchain

Ganache and Truffle are indispensable tools for creating a personal Ethereum blockchain. Ganache provides a local and isolated development environment for rapid smart contract testing, while Truffle offers a comprehensive development framework that streamlines the development, testing, and deployment processes. Together, they enhance the efficiency and developer-friendliness of Ethereum application development, making them essential for Ethereum developers.