

Bhartiya Vidya Bhavan’s

Sardar Patel Institute of Technology, Mumbai-400058

Department of Computer Science and Engineering

**OEIT1:Blockchain Technology and Applications**

**Name : Adwait Purao**

**UCID: 2021300101**

**Division : Comps B**

**Lab-4A: Development of Smart Contract using Solidity**

**Objective:** Write a smart contract, compile, test, debug, and deploy using Solidity

**Outcomes:** After successful completion of the lab, students should be able to

1. Use Remix Ethereum IDE
2. Write a smart contract using Solidity, compile, debug, and deploy it.
3. Install node.js, truffle and ganache-cli
4. Create a truffle project and configure a development network
5. Create and deploy smart contracts
6. Interact with the smart contract from the Truffle console
7. Write tests for testing the main features offered by Solidity.

**System Requirements:**

PC (C2D, 4GB RAM, 100GB HDD space and NIC), Ubuntu Linux 14.04-22.04

Internet connectivity, node.js, truffle, Ganache-cli, Remix IDE, MS VS

**About Ethereum Blockchain:** Ethereum is a decentralized open-source platform based on the blockchain domain, used to run smart contracts i.e. applications that execute the program exactly as it was programmed without the possibility of any fraud, interference from a third party, censorship, or downtime. It serves as a platform for nearly 2,60,000 different cryptocurrencies. Ether is a cryptocurrency generated by Ethereum miners, used to reward for the computations performed to secure the blockchain. [3]

**Ethereum Virtual Machine(EVM)**

Ethereum Virtual Machine, abbreviated as EVM, is a runtime environment for executing smart contracts in ethereum. It focuses widely on providing security and execution of untrusted code using an international network of public nodes. EVM is specialized to prevent Denial-of-service attacks and confirms that the program does not have any access to each other’s state, it also ensures that the communication is established without any potential interference.

Ethereum is an open-source, public,blockchain-based distributed computing platform. It features smart contract (scripting) functionality, which facilitates online contractual agreements. The Ethereum elements include:

• Blocks and Blockchain

• Wallets and client software

• Nodes and miners

• APIs and tools

• Supporting protocols

• Programming languages

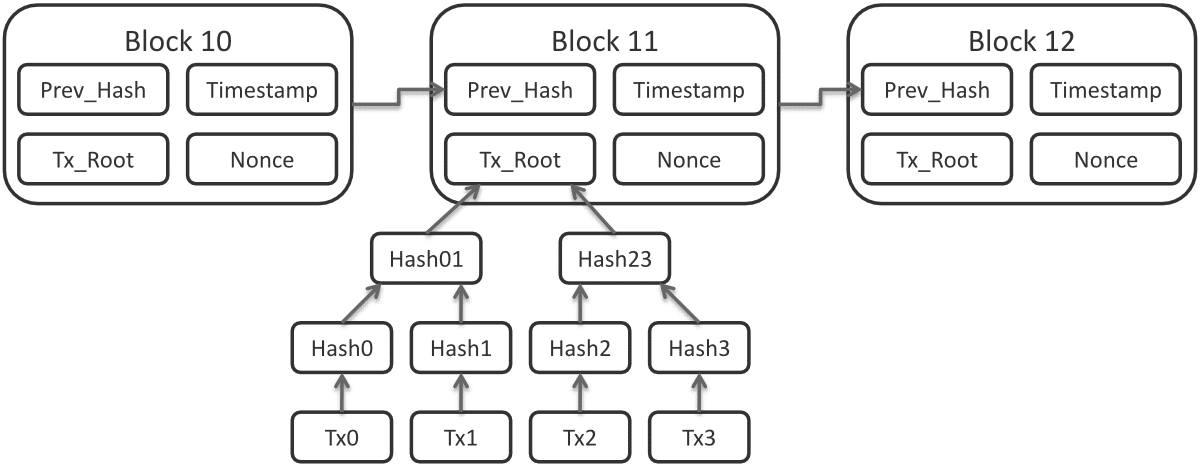


Figure-1: Blockchain Implementation

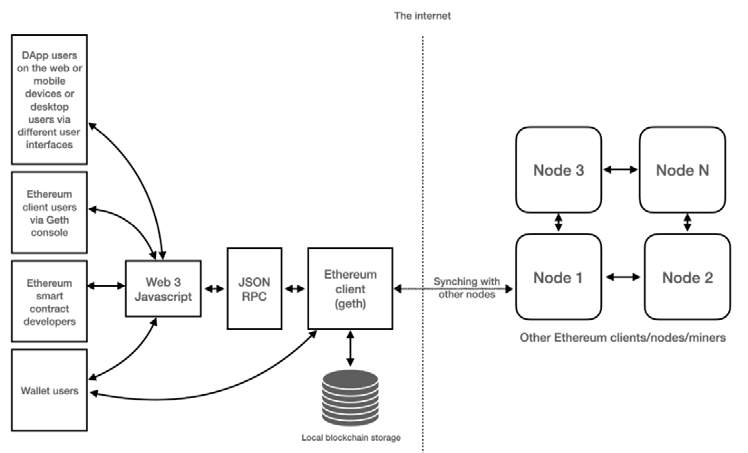


Figure-2: Ethereum high-level ecosystem

**Smart Contract:** Smart contracts are high-level program codes compiled into EVM byte code and deployed to the ethereum blockchain for further execution. It allows us to perform credible transactions without interference from a third party. These transactions are trackable and irreversible. Languages used to write smart contracts are Solidity (a language library with similarities to C and JavaScript)

**Solidity:** Solidity is a programming language created by Ethereum, which is the second-largest market of cryptocurrency by capitalization, released in the year 2015 and led by Christian Reitwiessner. Some key features of solidity are listed below:

* Solidity is a high-level programming language designed for implementing smart contracts.
* It is a statically typed object-oriented(contract-oriented) language.
* Solidity is highly influenced by Python, c++, and JavaScript, which run on the Ethereum Virtual Machine(EVM).
* Solidity supports complex user-defined programming, libraries, and inheritance.
* Solidity is the primary language for blockchains running platforms.
* Solidity can be used to create contracts like voting, blind auctions, crowdfunding, multi-signature wallets, etc.

**Truffle Suite:** The Truffle Suite is a collection of tools that is designed to ease the testing of your Ethereum Solidity smart contracts. Instead of using a testnet (such as Ropsten) and getting test ethers to deploy and use your contracts, you can now directly deploy your smart contracts locally on your computer. Best of all, there is no need to wait for transactions to be confirmed — your transactions are immediately confirmed after you have submitted them.

The Truffle Suite contains three components:

* Truffle — A world class development environment, testing framework and asset pipeline for blockchains using the Ethereum Virtual Machine (EVM).
* Ganache — A simulated Ethereum blockchain where you can use to deploy contracts, develop your applications, and run tests.
* Drizzle — A collection of front-end libraries that make writing dapp front-ends easier and more predictable.

**Procedure:**

***Part-I: Online Mode (Remix IDE)***

Steps to Execute Solidity Smart Contract using Remix IDE

Refer to [1],[3] online tutorials on Solidity programming using Remix IDE

Add the screenshots for every step with a brief description and conclusion towards the end.

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

contract Calculator {

// Function to add two numbers

function add(uint256 a, uint256 b) public pure returns (uint256) {

return a + b;

}

// Function to subtract two numbers

function subtract(uint256 a, uint256 b) public pure returns (uint256) {

require(a >= b, "Subtraction underflow");

return a - b;

}

// Function to multiply two numbers

function multiply(uint256 a, uint256 b) public pure returns (uint256) {

return a \* b;

}

// Function to divide two numbers

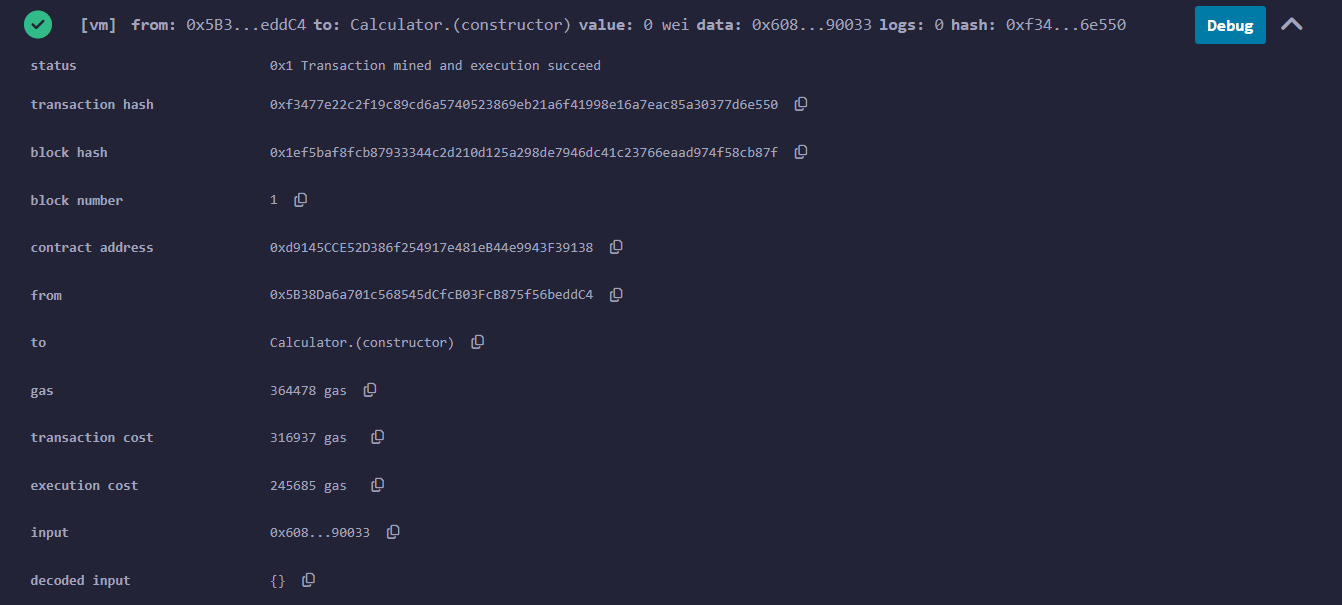
function divide(uint256 a, uint256 b) public pure returns (uint256) {

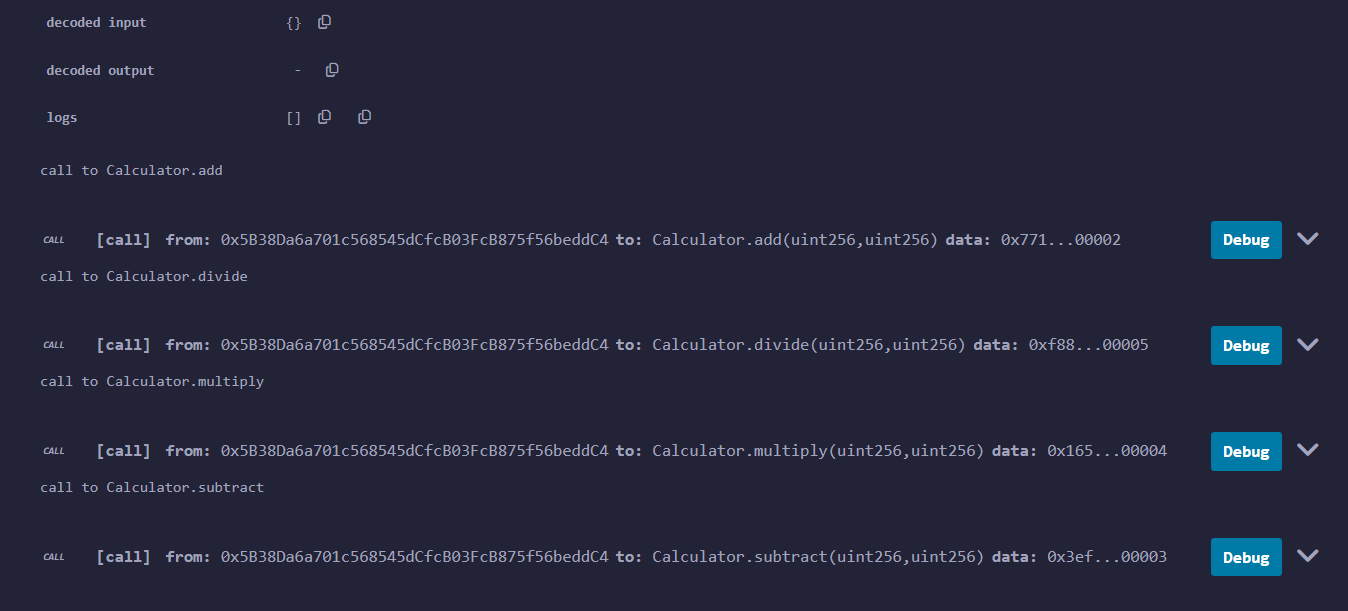
require(b != 0, "Division by zero");

return a / b;

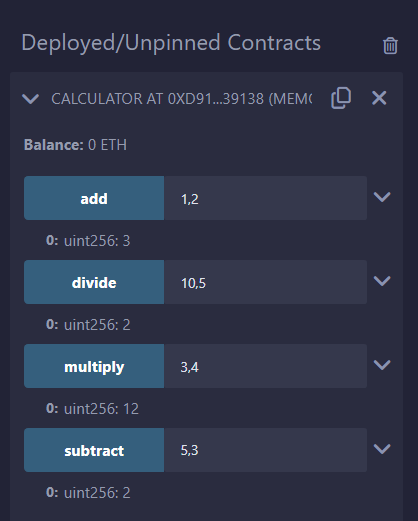
}

}









***Part-II: (Offline mode)***

Refer to [4] Deploying and Testing Ethereum Smart Contracts using Truffle and Ganache by Wei-Meng-Lee

[1] Download and install the node.js

[2] Install truffle (npm install -g truffle)

[3] Install ganache-cli

[4] Write a smart contract using solidity language

[5] Compile the code using solcjs

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

contract HelloWorld {

string public greeting;

constructor() {

greeting = "Hello, World!";

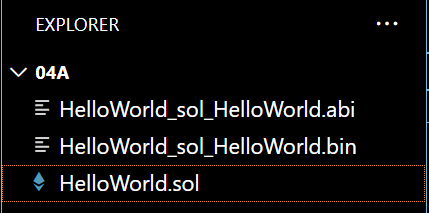
}

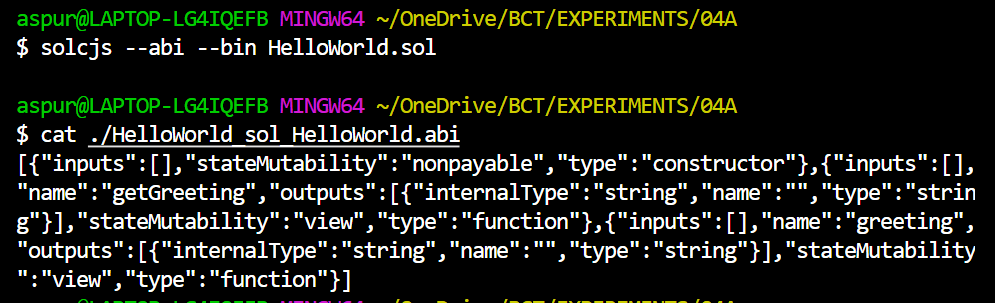
function getGreeting() public view returns (string memory) {

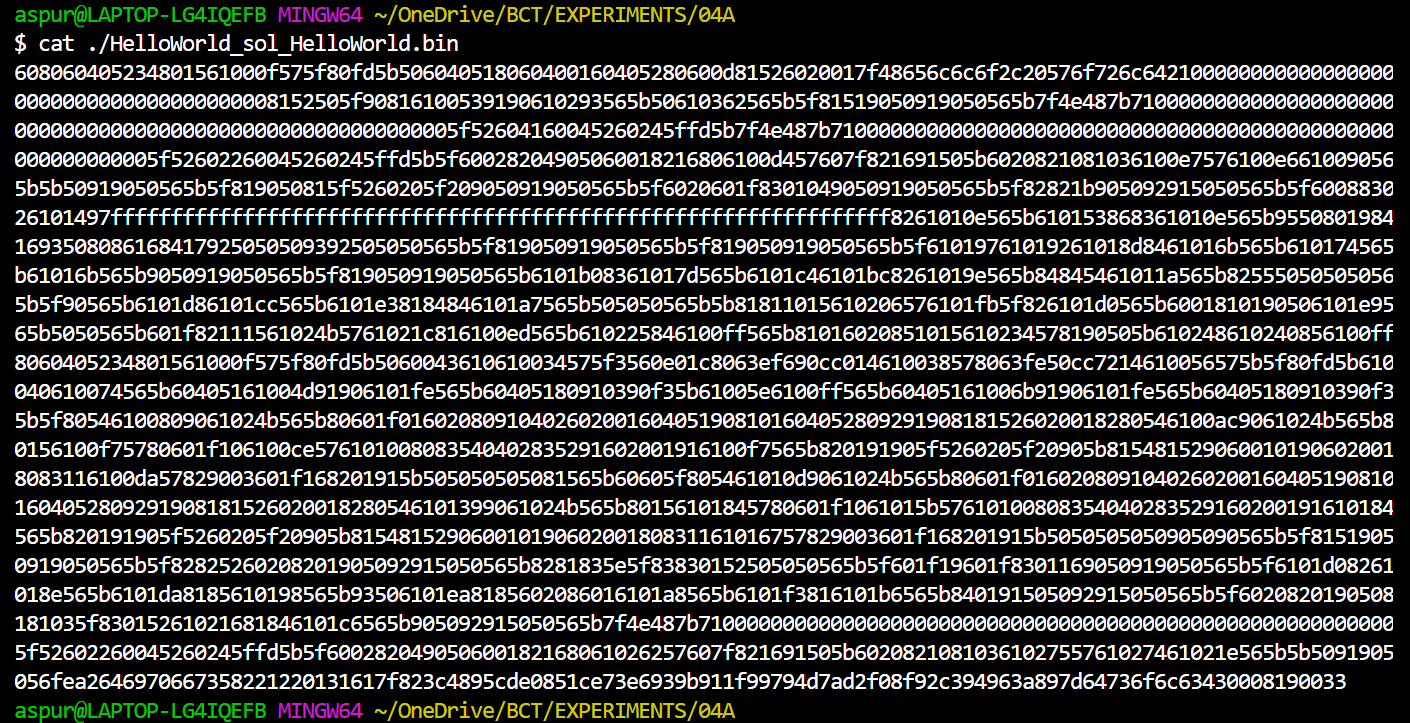
return greeting;

}

}







**Conclusion:**

Through the experimentation process, I've gained a comprehensive understanding of Solidity and Smart Contracts, including the intricacies involved in writing and deploying a basic smart contract onto the blockchain.

**References:**

[1] [Remix IDE](https://remix.ethereum.org) and [Remix’s documentation!](https://remix-ide.readthedocs.io/en/latest/)

[2] Building Ethereum Dapps: Decentralized applications on the Ethereum blockchain by Roberto Infante, Manning Publications

[3] [Steps to Execute Solidity Smart Contract using Remix IDE - GeeksforGeeks](https://www.geeksforgeeks.org/steps-to-execute-solidity-smart-contract-using-remix-ide/) by Jeetesh Gavande

<https://www.geeksforgeeks.org/steps-to-execute-solidity-smart-contract-using-remix-ide/>

[4] [Deploying and Testing Ethereum Smart Contracts using Truffle and Ganache | by Wei-Meng Lee | CryptoStars](https://blog.cryptostars.is/deploying-and-testing-ethereum-smart-contracts-using-truffle-and-ganache-a2b00828edbc)

<https://blog.cryptostars.is/deploying-and-testing-ethereum-smart-contracts-using-truffle-and-ganache-a2b00828edbc>