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

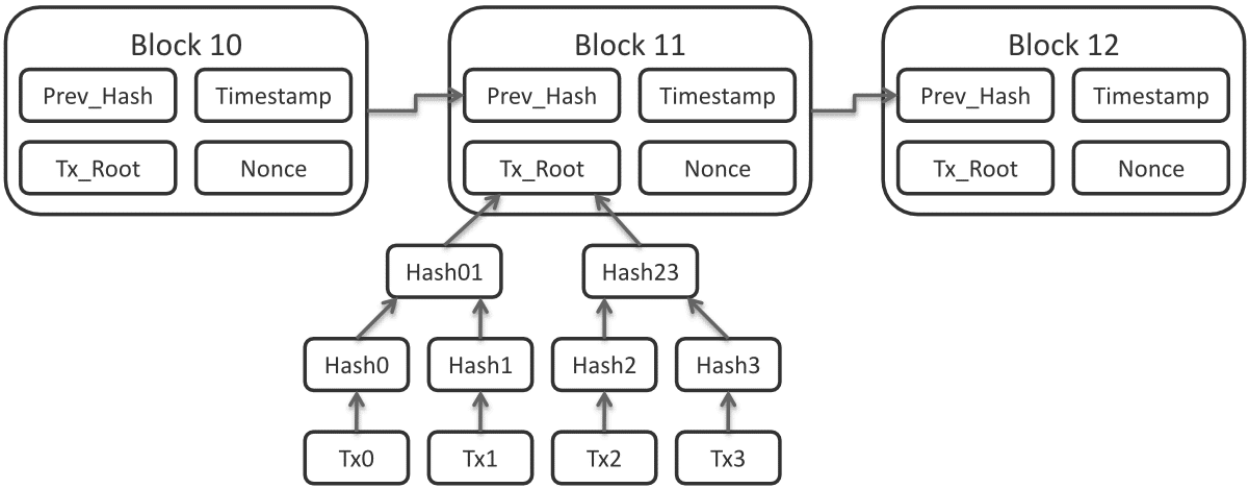


Figure-1: Blockchain Implementation

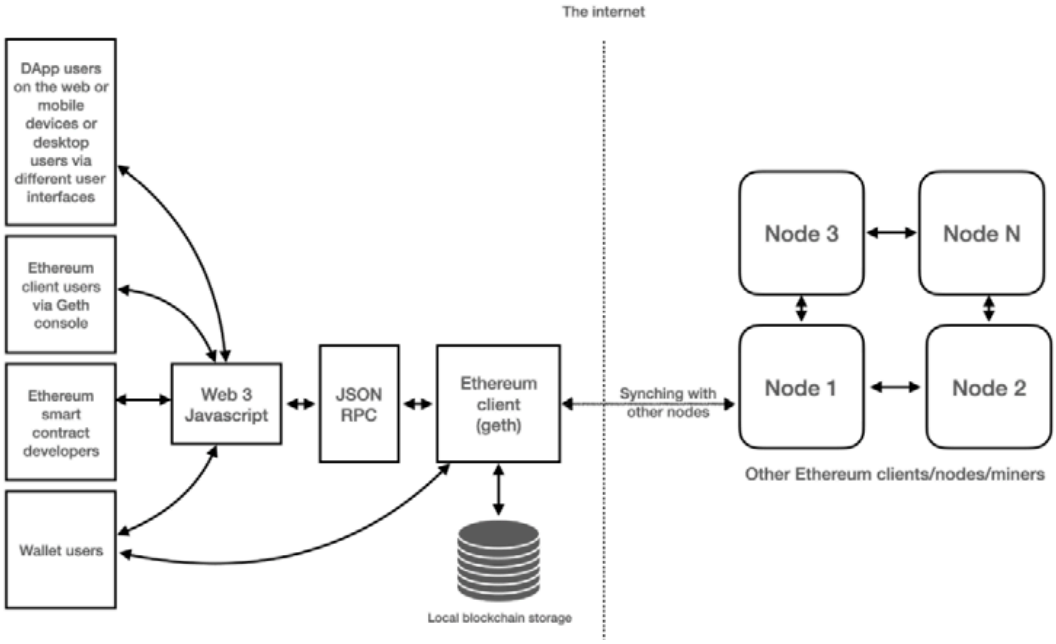


Figure-2: Ethereum high-level ecosystem

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:

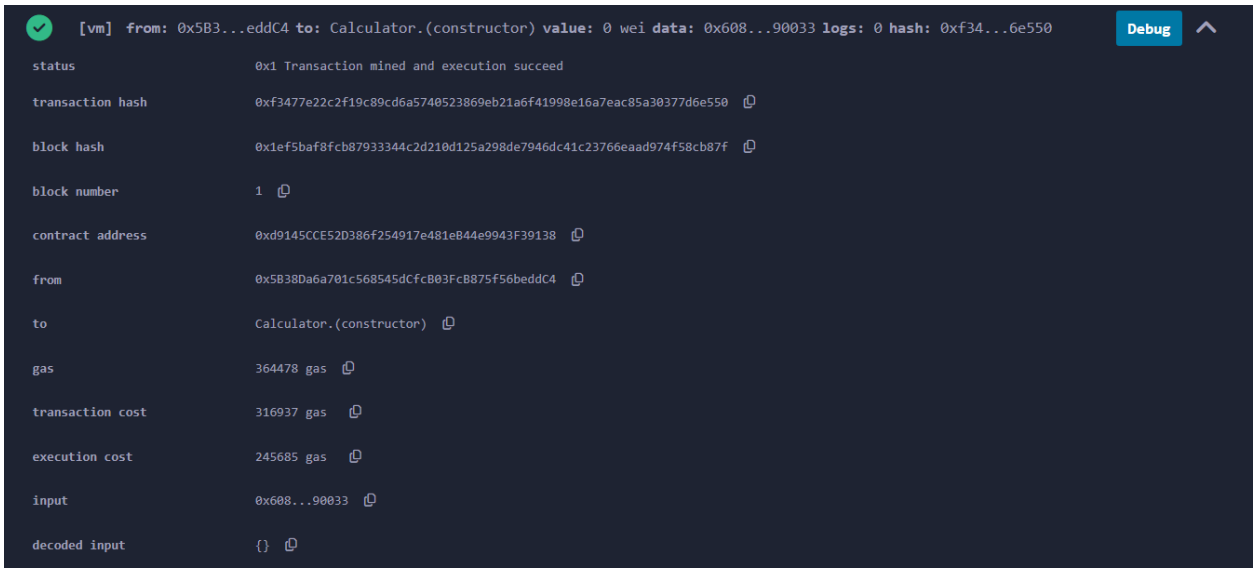
Part-I: Online Mode (Remix IDE)

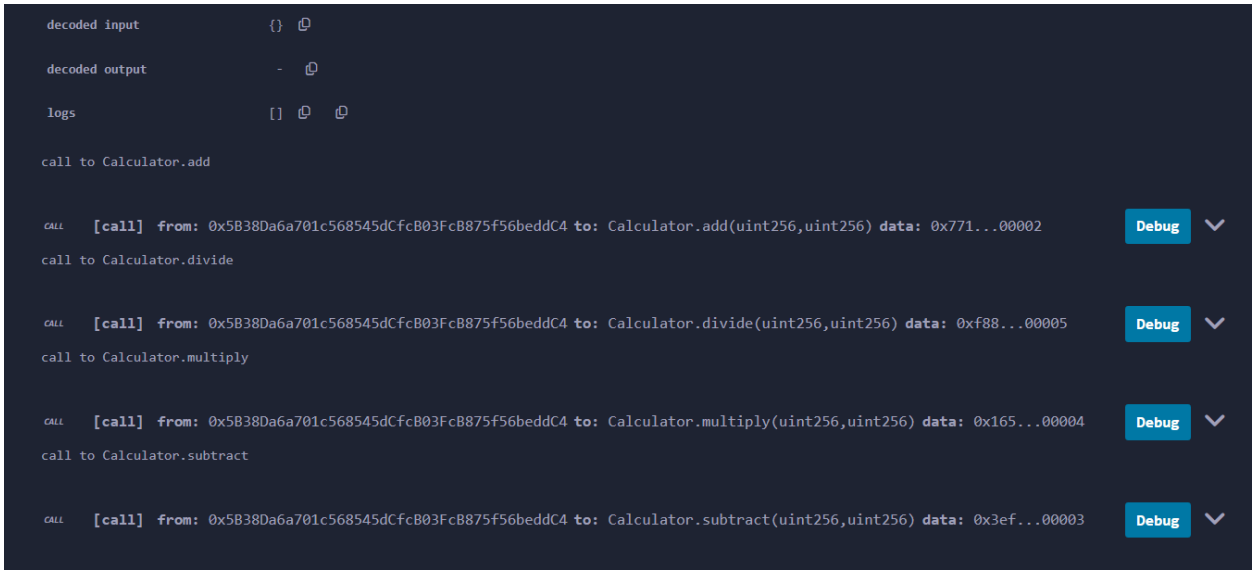
Steps to Execute Solidity Smart Contract using Remix IDE
Refer to [1],[3] online tutorials on Solidity programming using Remix IDE

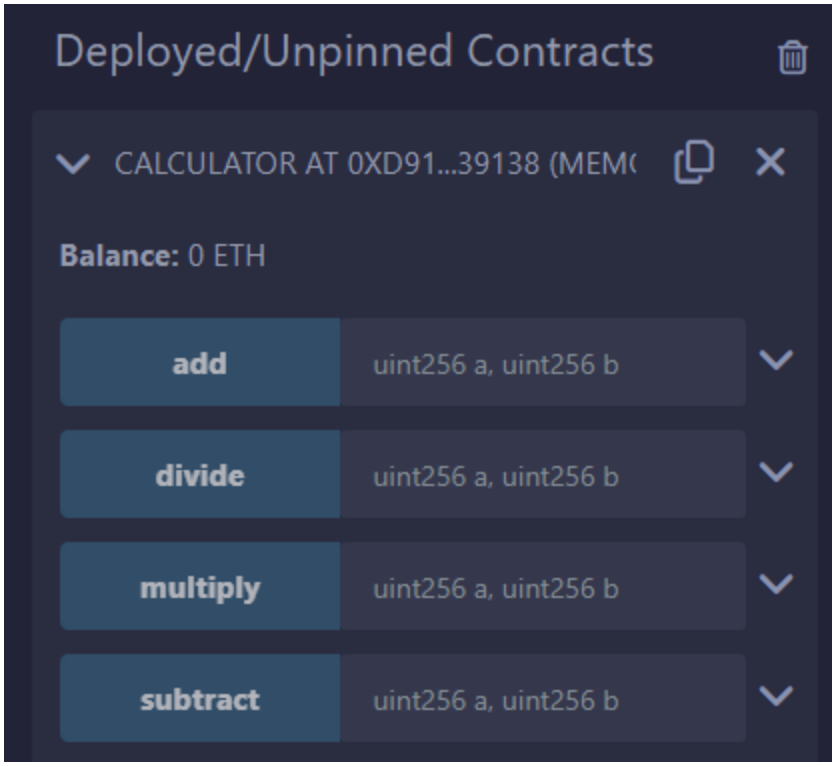
Add the screenshots for every step with a brief description and conclusion towards the end.

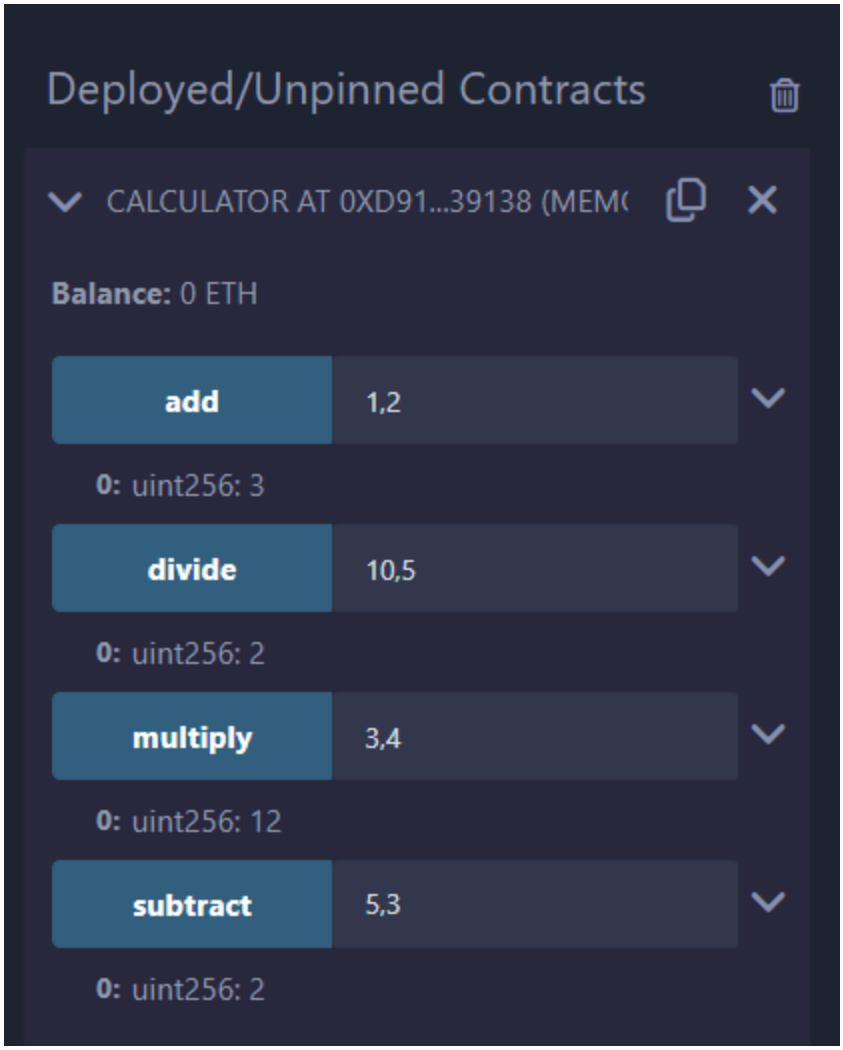
```
// SPDX-License-Identifier: MIT

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;
    }
}
```


[vm] from: 0x5B3...eddC4 to: Calculator.(constructor) value: 0 wei data: 0x608...90033 logs: 0 hash: 0xf34...6e550
Debug
status 0x1 Transaction mined and execution succeed
transaction hash 0xf3477e22c2f19c89cd6a5740523869eb21a6f41998e16a7eac85a30377d6e550
block hash 0x1ef5baf8fcb87933344c2d210d125a298de7946dc41c23766eaad974f58cb87f
block number 1
contract address 0xd9145CCE52D386f254917e481eB44e9943F39138
from 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to Calculator.(constructor)
gas 364478 gas
transaction cost 316937 gas
execution cost 245685 gas
input 0x608...90033
decoded input {}


decoded input {}
decoded output -
logs []
call to Calculator.add
CALL [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: Calculator.add(uint256,uint256) data: 0x771...00002
Debug
call to Calculator.divide
CALL [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: Calculator.divide(uint256,uint256) data: 0xf88...00005
Debug
call to Calculator.multiply
CALL [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: Calculator.multiply(uint256,uint256) data: 0x165...00004
Debug
call to Calculator.subtract
CALL [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: Calculator.subtract(uint256,uint256) data: 0x3ef...00003
Debug


Deployed/Unpinned Contracts
CALCULATOR AT 0XD91...39138 (MEM(
Balance: 0 ETH
add
uint256 a, uint256 b
divide
uint256 a, uint256 b
multiply
uint256 a, uint256 b
subtract
uint256 a, uint256 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;

    }

}
```

```
aspur@LAPTOP-LG4IQEFB MINGW64 ~/OneDrive/BCT/EXPERIMENTS/04A
$ solcjs --abi --bin HelloWorld.sol

aspur@LAPTOP-LG4IQEFB MINGW64 ~/OneDrive/BCT/EXPERIMENTS/04A
$ cat ./HelloWorld_sol_HelloWorld.abi
[{"inputs":[],"stateMutability":"nonpayable","type":"constructor"},{"inputs":[],
"name":"getGreeting","outputs":[{"internalType":"string","name":"","type":"strin
g"}],"stateMutability":"view","type":"function"},{"inputs":[],"name":"greeting",
"outputs":[{"internalType":"string","name":"","type":"string"}],"stateMutability
":"view","type":"function"}]
```

```
aspur@LAPTOP-LG4IQEFB MINGW64 ~/OneDrive/BCT/EXPERIMENTS/04A
$ cat ./HelloWorld_sol_HelloWorld.bin
608060405234801561000f575f80fd5b506040518060400160405280600d81526020017f48656c6c6f2c20576f726c6421000000000000000000
0000000000000000000008152505f90816100539190610293565b50610362565b5f8151905091905056 5b7f4e487b7100000000000000000000
0000000000000000000000000000000000005f52604160045260245ffd5b7f4e487b7100000000000000000000000000000000000000000000000000
00000000005f526022600452 60245ffd5b5f600282049050600182168061 00d457607f821691505b602082108103610 0e7576100e661009056
5b5b50919050565b5f819050815f5260205f209050919050565b5f6020601f830104905091905056 5b5f82821b905092915050565b5f6008830
26101497ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff8261010e565b610153868361010e565b9550801984
16935080861684179250505093925050565b5f819050919050565b5f819050919050565b5f610197610192610 18d8461016b565b610174565
b61016b565b9050919050565b5f819050919050565b6101b08361017d565b6101c46101bc8261019e565b84845461011a565b82555050505056
5b5f90565b6101d86101cc565b6101e38184846101a7565b505050565b5b818110156102065 76101fb5f826101d0565b600181019050610 1e95
65b5050565b601f82111561024b5761021c816100ed565b610225846100ff565b810160208510156102345781905050 5b61024861024085600ff
806040523480156100 0f575f80fd5b506004361061003 4575f3560e01c8063ef690cc014610038578063fe50cc7214610056575b5f80fd5b610
040610074565b60405161004d91906101fe565b60405180910390f35b61005e6100ff565b60405161006b91906101fe565b604051809103 90f3
5b5f8054610080906102 4b565b80601f0160208091040260200160405190810160405280929190818152602001828054610 0ac9061024b565b8
0156100f75780601f106100ce576101008083540402835291602001916100f7565b820191905f5260205f20905b8154815290600101906020 01
8083116100da57829003601f168201915b505050505081565b60605f805461010d9061024b565b80601f0160208091040260200160405190810
160405280929190818152602001828054610139061024b565b8015610184578060 1f1061015b5761010080835404028352916020019161 0184
565b820191905f5260205f20905b81548152906001019060200180831161016757829003601f168201915b505050505090509056 5b5f8151905
0919050565b5f828252602082019050929150565b8281835e5f8383015250505056 5b5f601f19601f8301169050919050565b5f6101d08261
018e565b6101da8185610198565b93506101ea818560208601 6101a8565b6101f3816101b6565b840191505092915050565b5f6020820190508
181035f830152610216 81846101c6565b905092915050565b7f4e487b7100000000000000000000000000000000000000000000000000000000
5f526022600452602 45ffd5b5f60028204905060018216806102625760 7f821691505b602082108103610275761027461021e565b5b5091905
056fea2646970667358221220131617f823c4895cde0851ce73e6939b911f99794d7ad2f08f92c394963a897d64736f6c63430008190033
aspur@LAPTOP-LG4IQEFB MINGW64 ~/OneDrive/BCT/EXPERIMENTS/04A
```

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 and Remix's documentation!

[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 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