



JUNAID GIRKAR
60004190057
BE COMPS A2

BLOCKCHAIN TECHNOLOGIES

EXPERIMENT - 2

AIM: Create, Test and Deploy an Ethereum Smart Contract

THEORY:

A "smart contract" is simply a program that runs on the Ethereum blockchain. It's a collection of code (its functions) and data (its state) that resides at a specific address on the Ethereum blockchain.

Smart contracts are a type of Ethereum account. This means they have a balance and can be the target of transactions. However they're not controlled by a user, instead they are deployed to the network and run as programmed. User accounts can then interact with a smart contract by submitting transactions that execute a function defined on the smart contract. Smart contracts can define rules, like a regular contract, and automatically enforce them via the code. Smart contracts cannot be deleted by default, and interactions with them are irreversible.

With conventional contracts, a document outlines the terms of a relationship between two parties, which is enforceable by law. If one Party A violates the terms, Party B can take Party A to court for not complying with the agreement. A smart contract fortifies such agreements in code so the rules are automatically enforced without courts (or any third party) getting involved.

Some common ways of using smart contracts are:

- Multisignature accounts: Funds can only be spent when a required percentage of people agree.
- Encoding financial agreements: Manage agreements between users. Say, if one person buys insurance from an insurance company, the rules of when the insurance can be redeemed can be programmed into a smart contract.
- Agreements based on the outside world: Pull in data from the outside world (financial, political, or whatever) with the help of oracles.
- Provide third party: Similar to how a software library works, smart contracts can work with other smart contracts in a chain.



Shri Vile Parle Kelavani Mandal's DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING

(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA: 3.18)



- **Storage:** Store information about an application, such as domain registration information or membership records. Storage in a blockchain like Ethereum is unique in that the data is immutable and can't be erased.

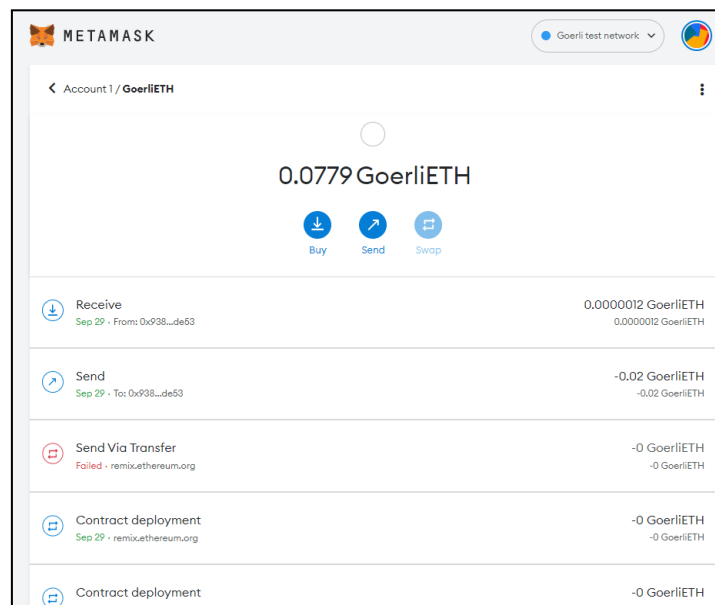
How is a smart contract set up?

- A developer can create a smart contract by writing a slab of code – spelling out the rules, such as that 10 ether can only be retrieved by Alice 10 years from now.
- The developer then pushes the smart contract to the Ethereum network, which is what enforces the contract – not allowing anyone to take the money unless they follow the exact rules in the code. Thousands of computers from around the world then all have a copy of this smart contract.

How do I use a smart contract?

- Anyone can use smart contracts if they have Ethereum's native token ether, which can be bought on cryptocurrency exchanges.
- Ethereum apps will usually provide instructions for how to use their specific app and underlying smart contracts. A common method is to use an Ethereum wallet tool, such as Metamask, to send the ether.
- Users can use smart contracts for a range of use cases. Users can publish uncensorable posts to microblogging apps or lend out money without an intermediary, using a variety of Ethereum apps.

Creation of a Metamask Wallet and using free Goerli Ethers from the faucet:





Doing a transaction of Goerli Ethers to another account id:

Transaction Details

[Overview](#) [State](#)

[This is a Goerli Testnet transaction only]

Transaction Hash:	0x3c5788e8ffdd34cb1818e4dc09d63ef0ae00a0aaf2cd3fdc5cfee8cafca531
Status:	Success
Block:	7679033 5 Block Confirmations
Timestamp:	1 min ago (Sep-29-2022 06:33:12 AM +UTC)
From:	0xdc78ee259af88fb86862a17ef9382600210c6b9a
To:	0xd2a5d5306167d6d81ee23be24804c4b9f974279f
Value:	0.0005 Ether (\$0.00)
Transaction Fee:	0.000031500001176 Ether (\$0.00)
Gas Price:	0.000000001500000056 Ether (1.500000056 Gwei)
Gas Limit & Usage by Txn:	21,000 21,000 (100%)
Gas Fees:	Base: 0.0000000056 Gwei Max: 1.500000079 Gwei Max Priority: 1.5 Gwei
Burnt & Txn Savings Fees:	Burnt: 0.00000000001176 Ether (\$0.00) Txn Savings: 0.00000000000483 Ether (\$0.00)
Others:	Txn Type: 2 (EIP-1559) Nonce: 0 Position: 36
Input Data:	0x

[Click to see Less](#)

ⓘ A transaction is a cryptographically signed instruction from an account that changes the state of the blockchain. Block explorers track the details of all transactions in the network. Learn more about transactions in our [Knowledge Base](#).

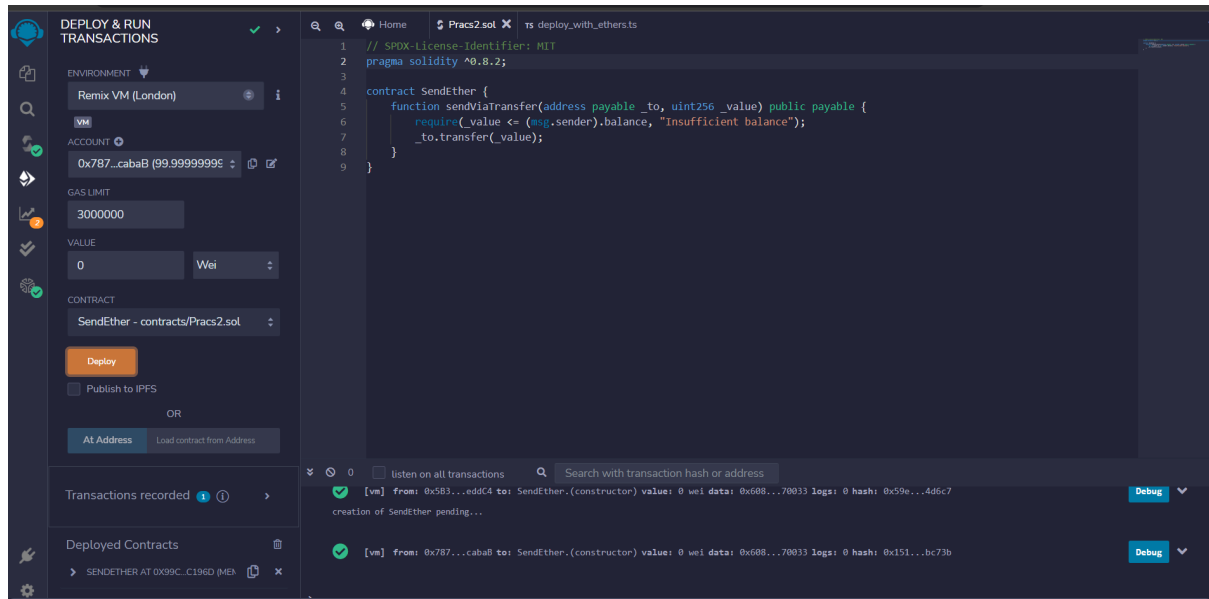
Creating a Smart Contract:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.2;

contract SendEther {
    function sendViaTransfer(address payable _to) public payable {
        require(msg.value <= (msg.sender).balance, "Insufficient
balance");
        _to.transfer(msg.value);
    }
}
```



Deploying the Smart Contract:



Transaction Fees for deploying the Smart Contract:

Etherscan

All Filters Search by Address / Txn Hash / Block

Goerli Testnet Network Home

Transaction Details < >

Overview State

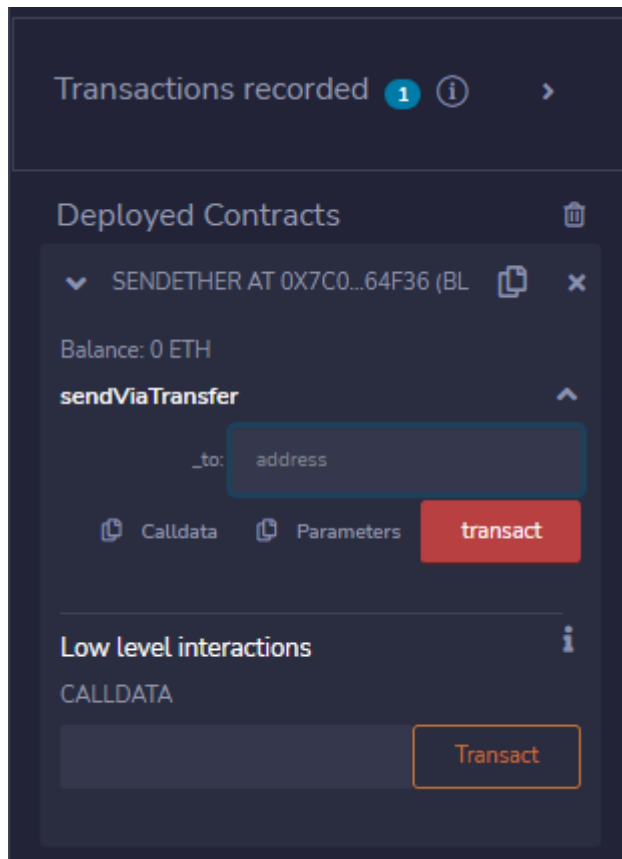
[This is a Goerli Testnet transaction only]

Transaction Hash:	0xf632f385942c47e00949a319a67ec0939cd69a572d65f20b395229d58236503c
Status:	Success
Block:	7710317 2 Block Confirmations
Timestamp:	20 secs ago (Oct-04-2022 01:01:12 PM +UTC)
From:	0xd2a5d5306167d6d81ee23be24804c4b9f974279f
To:	[Contract 0x7c0f20cbb621c7042b84c95c2096376028c64f36 Created]
Value:	0 Ether (\$0.00)
Transaction Fee:	0.000428897503774298 Ether (\$0.00)
Gas Price:	0.000000002500000022 Ether (2.500000022 Gwei)

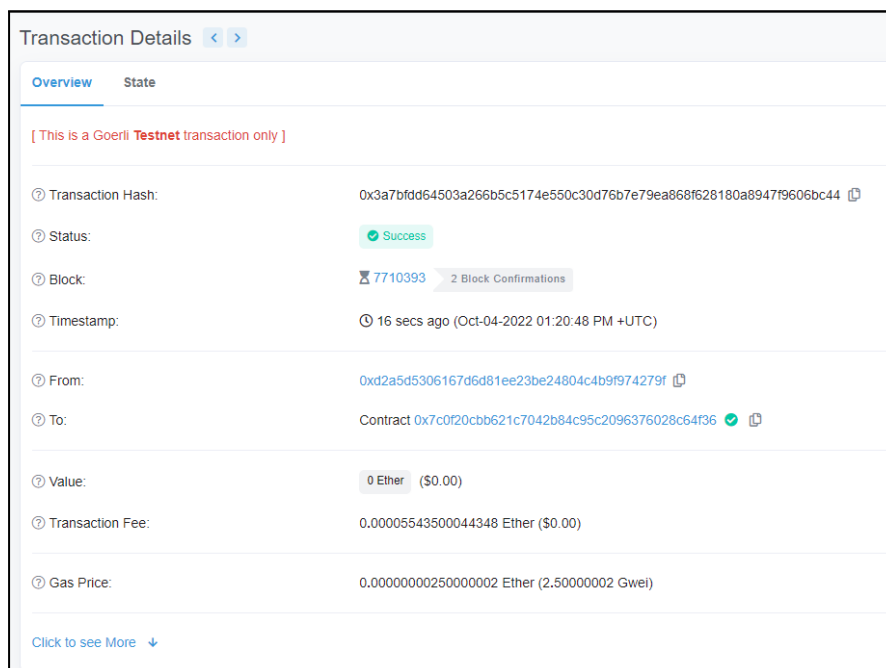
Click to see More



Using the Smart Contract:



Ethers sent via Smart Contract:





Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING

(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA: 3.18)

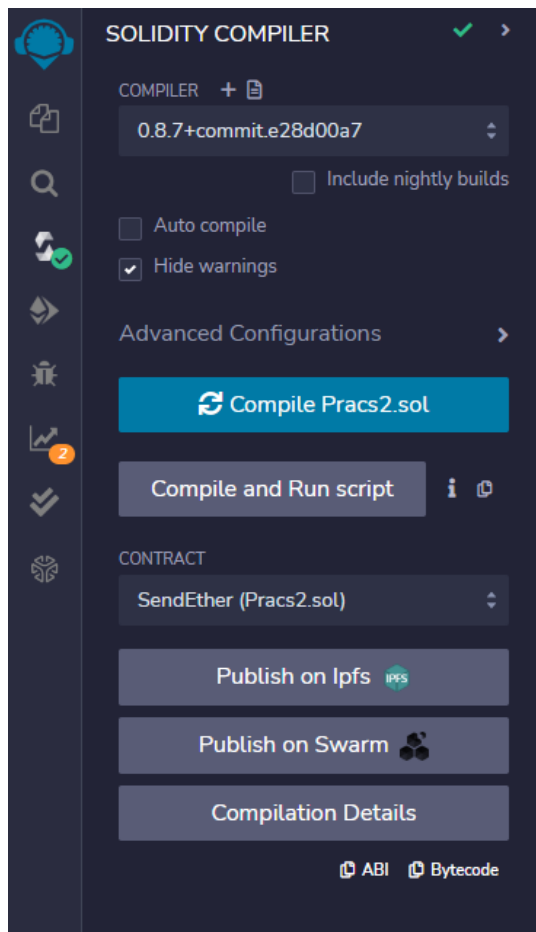


Creating a Second Smart Contract that can take value input:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.2;

contract SendEther {
    function sendViaTransfer(address payable _to, uint256 _value) public
    payable {
        require(_value <= (msg.sender).balance, "Insufficient balance");
        _to.transfer(_value);
    }
}
```

Compiling the Smart Contract on EVM (Ethereum Virtual Machine)





Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA: 3.18)



Deploying the Smart Contract:

The screenshot shows the Remix IDE interface with a smart contract named 'SendEther' being deployed. The contract code is visible in the editor, and the deployment process is shown in the left sidebar. A MetaMask notification is displayed on the right, showing the transaction details and gas fees.

MetaMask Notification Details:

DETAILS	DATA
fee	0.000469 GoerliETH
Site suggested	Very likely in <15 seconds
Max fee	0.0004688 GoerliETH
Total	0.0004688 GoerliETH
Amount + gas fee	0.0004688 GoerliETH

Buttons: Reject, Confirm

Taking Transaction Value as input:

The screenshot shows the 'Deployed Contracts' panel in Remix IDE. It displays the 'SENDETHER AT 0XD4F...F4FE9 (BL)' contract. The 'Balance: 0 ETH' is shown. The 'sendViaTransfer' function is selected, and the input fields for '_to:' and '_value:' are visible. The '_to:' field contains the address '0xd2A5D5306167D6d81EE23be' and the '_value:' field contains '0'. A 'transact' button is present. Below the function, there is a section for 'Low level interactions' with a 'CALLDATA' field and a 'Transact' button.



Transaction:

Etherscan

All Filters ▾ Search by Address / Tx

Goerli Testnet Network

Transaction Details < >

Overview Internal Txns State

[This is a Goerli Testnet transaction only]

⑦ Transaction Hash:	0xfbc2c9838fcac304b49d159b5c16444379c3cb8f60c2784fb9d44cfe666f2f55e
⑦ Status:	Success
⑦ Block:	7710414 29 Block Confirmations
⑦ Timestamp:	7 mins ago (Oct-04-2022 01:26:12 PM +UTC)
⑦ From:	0xd2a5d5306167d6d81ee23be24804c4b9f974279f
⑦ To:	Contract 0xd4f3ed74d46003f3527a7b85d29469fa40df4fe9
⑦ Value:	0 Ether (\$0.00)
⑦ Transaction Fee:	0.000056192500404586 Ether (\$0.00)
⑦ Gas Price:	0.000000002500000018 Ether (2.500000018 Gwei)

Click to see More ↓

Lottery.sol

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

pragma solidity ^0.8.11;

contract Lottery {
    address public owner;
    address payable[] public players;
    uint public lotteryId;
    mapping (uint => address payable) public lotteryHistory;

    constructor() {
        owner = msg.sender;
        lotteryId = 1;
    }
}
```




Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING

(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA: 3.18)



```
}

function getWinnerByLottery(uint lottery) public view returns
(address payable) {
    return lotteryHistory[lottery];
}

function getBalance() public view returns (uint) {
    return address(this).balance;
}

function getPlayers() public view returns (address payable[] memory)
{
    return players;
}

function enter() public payable {
    require(msg.value > .01 ether);

    // address of player entering lottery
    players.push(payable(msg.sender));
}

function getRandomNumber() public view returns (uint) {
    return uint(keccak256(abi.encodePacked(owner,
block.timestamp)));
}

function pickWinner() public onlyowner {
    uint index = getRandomNumber() % players.length;
    players[index].transfer(address(this).balance);

    lotteryHistory[lotteryId] = players[index];
    lotteryId++;

    // reset the state of the contract
    players = new address payable[](0);
}

modifier onlyowner() {
    require(msg.sender == owner);
    _;
}
```



Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA: 3.18)



}

Compile the code:

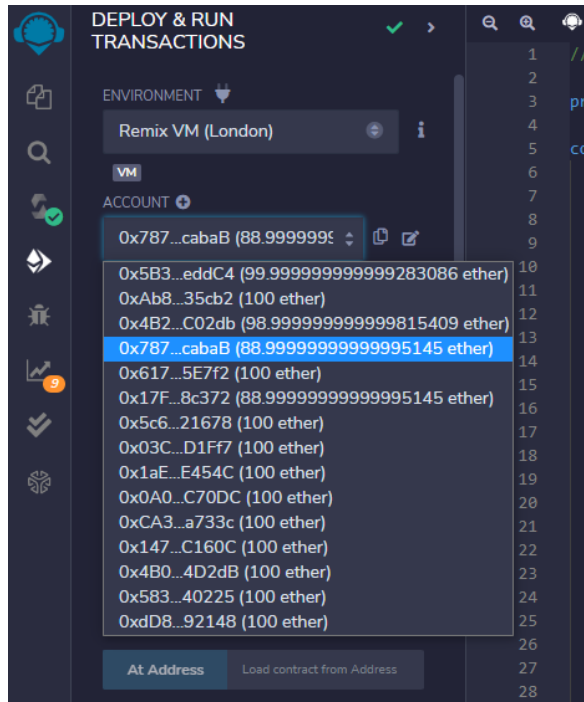
```
1 // SPDX-License-Identifier: MIT
2
3 pragma solidity ^0.8.11;
4
5 contract Lottery {
6     address public owner;
7     address payable[] public players;
8     uint public lotteryId;
9     mapping (uint => address payable) public lotteryHistory;
10
11     constructor() {
12         owner = msg.sender;
13         lotteryId = 1;
14     }
15
16     function getWinnerByLottery(uint lottery) public view returns (address payable) {
17         return lotteryHistory[lottery];
18     }
19
20     function getBalance() public view returns (uint) {
21         return address(this).balance;
22     }
23
24     function getPlayers() public view returns (address payable[] memory) {
25         return players;
26     }
27
28     function enter() public payable {
29         require(msg.value > .01 ether);
30
31         // address of player entering lottery
32     }
33 }
```

After Deploying the code and entering the lottery through 3 accounts:

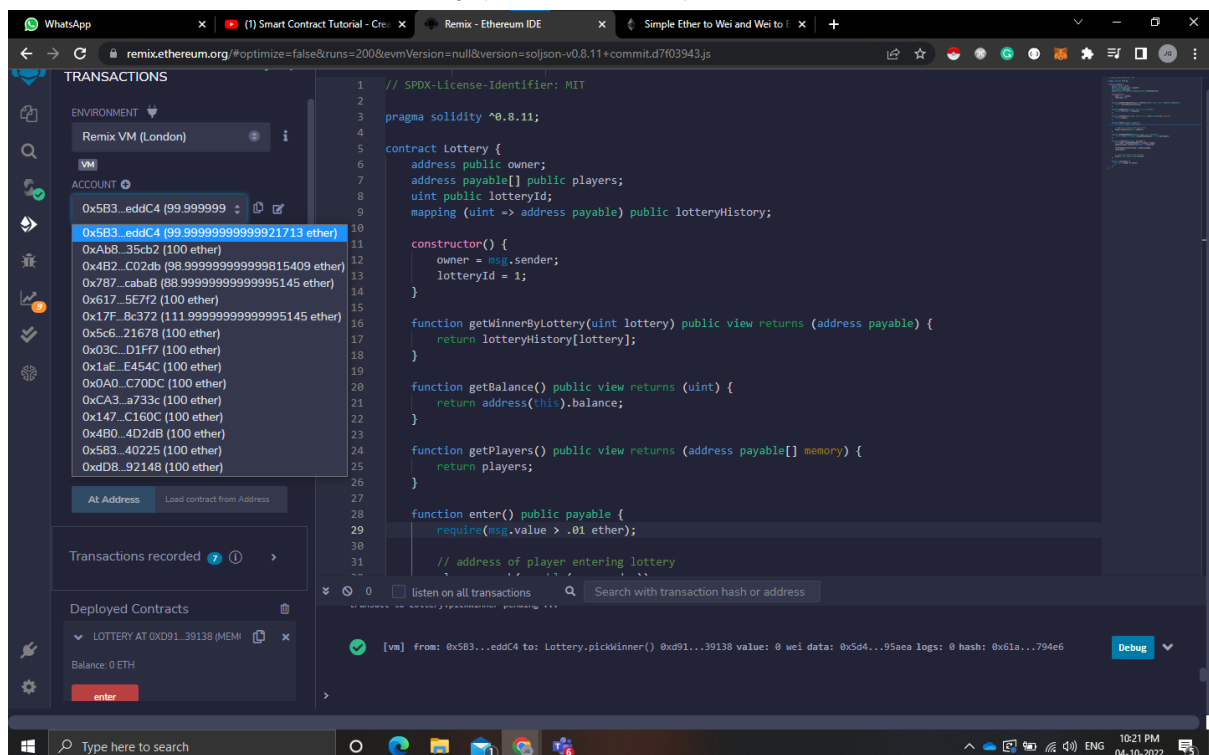
```
1 // SPDX-License-Identifier: MIT
2
3 pragma solidity ^0.8.11;
4
5 contract Lottery {
6     address public owner;
7     address payable[] public players;
8     uint public lotteryId;
9     mapping (uint => address payable) public lotteryHistory;
10
11     constructor() {
12         owner = msg.sender;
13         lotteryId = 1;
14     }
15
16     function getWinnerByLottery(uint lottery) public view returns (address payable) {
17         return lotteryHistory[lottery];
18     }
19
20     function getBalance() public view returns (uint) {
21         return address(this).balance;
22     }
23
24     function getPlayers() public view returns (address payable[] memory) {
25         return players;
26     }
27
28     function enter() public payable {
29         require(msg.value > .01 ether);
30
31         // address of player entering lottery
32     }
33 }
```



Wallet balance of players after entering the lottery:



After picking a winner of the lottery (Account 6 won):





Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING

(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA: 3.18)



CONCLUSION:

In this experiment, we learnt about cryptocurrency wallets like Metamask and test ethers like Goreli Ethers. We also learnt about smart contracts and implemented them using Solidity in Remid IDE. We did multiple transactions using the metamask wallet and smart contracts and implemented a randomised lottery system.