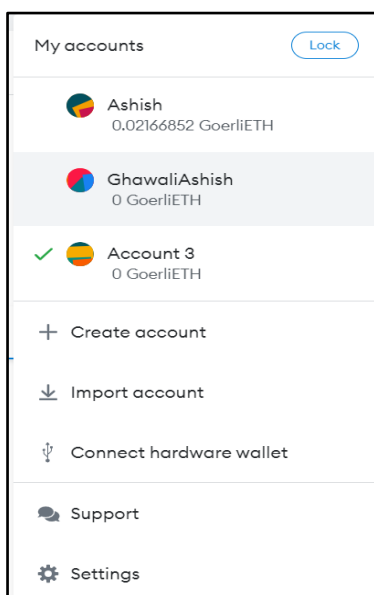
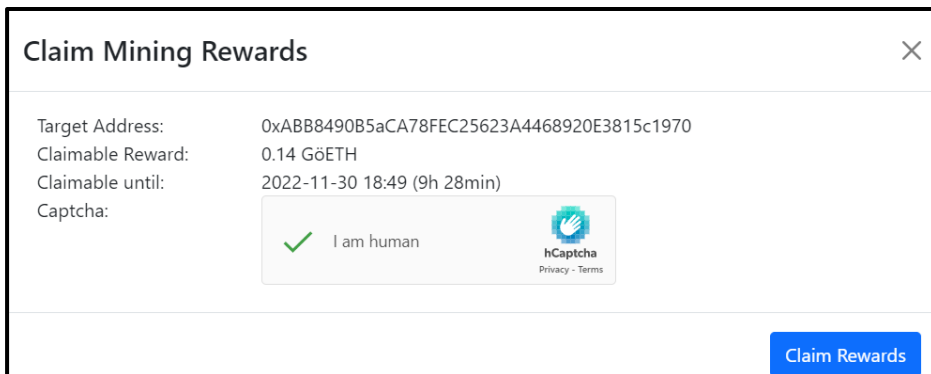
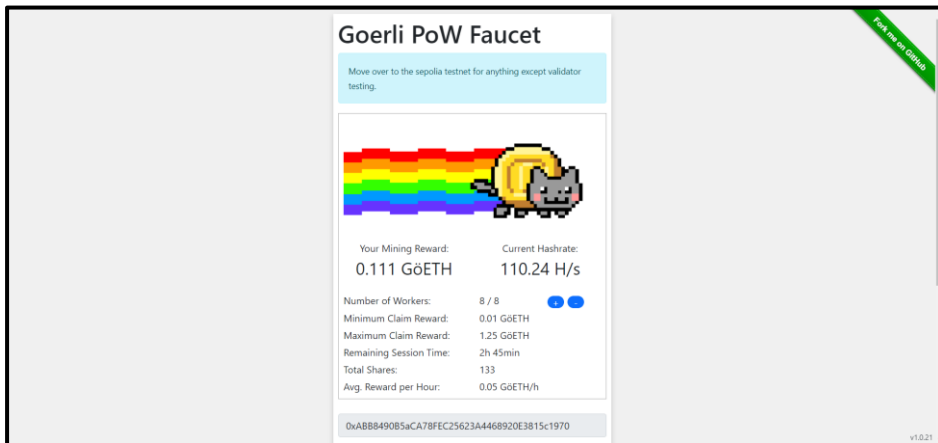


1. Install the metamask in browser. Setup the metamask digital cryptocurrency wallet. Create multiple accounts in metamask and connect with one of the ethereum blockchain test network. Perform the task buy ethers and send ethers from one account to another. Take the screenshots of created accounts, account assets and account transactions which showing the details of transaction.

Output:-



	<b>Send</b> Nov 22 · To: 0xea2...01b2	-0.0012 GoerliETH -0.0012 GoerliETH
	<b>Receive</b> Nov 3 · From: 0x6cc...f455	0.02205826 GoerliETH 0.02205826 GoerliETH
	<b>Receive</b> Nov 3 · From: 0x5ea...1830	0.00000001 GoerliETH 0.00000001 GoerliETH
	<b>Send</b> Nov 3 · To: 0x5e6...33e4	-0.00002 GoerliETH -0.00002 GoerliETH

Send

Status

Confirmed

View on block explorer

Copy transaction ID

From

0xc4C...2877

To

0xea2...01b2

Transaction

Nonce

2

Amount

-0.0012 GoerliETH

Gas Limit (Units)

21000

Gas Used (Units)

21000

Base fee (GWEI)

25.225502104

Priority fee (GWEI)

1.5

Total gas fee

0.000561 GoerliETH

Max fee per gas

0.000000041 GoerliETH

Total

0.00176124 GoerliETH

+ Activity log

Etherscan

Goerli Testnet Network

All Filters

Search by Address / Txn Hash / Block / Token / ENS

Home

Blockchain

Tokens

Misc

Goerli

Transaction Details

Overview

State

[ This is a Goerli Testnet transaction only ]

Transaction Hash:

0x9161ae4b8e1da99bcefe0eaf47d71caf5e837b7f5bc1b1cf08b4b68db9e8f8

Status:

Success

Block:

7997530 5731 Block Confirmations

Timestamp:

9 days 9 hrs ago (Nov-22-2022 06:44:48 AM +UTC)

From:

0xc4c58c869ca2afba746a45631b58dc342c6c2877

To:

0xea291dcbd3e7400deac3011bcc55d3a166fe01b2

Value:

0.0012 Ether (\$0.00)

Transaction Fee:

0.000561235544184 Ether (\$0.00)

Gas Price:

This website uses cookies to improve your experience. By continuing to use this website, you agree to its Terms and Privacy Policy.

2. Write a solidity smart contract to transfer funds (ethers) from user account to contract account using remixIDE and JavaScriptVM environment.

Code:

```
pragma solidity ^0.5.2;
contract Financialcontract2{
    address owner;

    constructor() public{
        owner=msg.sender;
    }

    modifier ifOwner(){
        if(owner!=msg.sender){
            revert();
        }
        else{
            _;
        }
    }

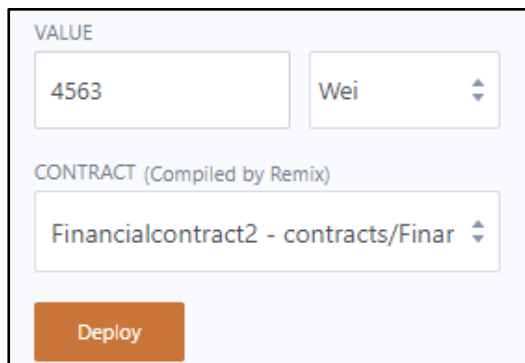
    function receiveDeposit() payable public{

    }

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

    function withdraw(uint funds)public ifOwner{
        msg.sender.transfer(funds);
    }
}
```

Output:



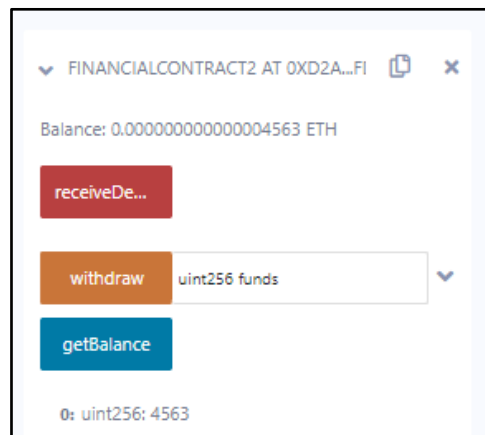
VALUE

4563 Wei

CONTRACT (Compiled by Remix)

Financialcontract2 - contracts/Finar

Deploy



FINANCIALCONTRACT2 AT 0XD2A...FI

Balance: 0.0000000000000004563 ETH

receiveDe...

withdraw uint256 funds

getBalance

0: uint256: 4563

3. Write a solidity smart contract to withdraw funds (ethers) from contract account to user account using remixIDE and JavaScriptVM environment.  
Program:

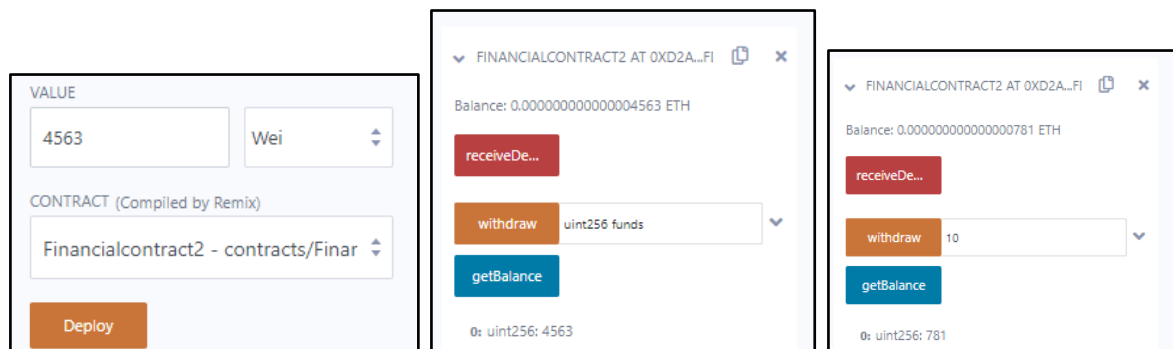
```
pragma solidity ^0.5.2;
contract Financialcontract2{
    address owner;
    constructor() public{
        owner=msg.sender;
    }
    modifier ifOwner(){
        if(owner!=msg.sender){
            revert();
        }
        else{
            _;
        }
    }
    function receiveDeposit() payable public{

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

    function withdraw(uint funds)public ifOwner{
        msg.sender.transfer(funds);
    }
    function getMoney(){

    }
}
```

**Output:**



4. Write a solidity smart contract to apply restriction that only owner of the contract can withdraw funds (ethers) from contract account to his/her user account using remixIDE and JavaScriptVM environment.

### Program:

```
pragma solidity ^0.5.2;
contract Financialcontract2{
    address owner;
    constructor() public{
        owner=msg.sender;
    }
    modifier ifOwner(){
        if(owner!=msg.sender){
            revert();
        }
        else{
            _;
        }
    }
    function receiveDeposit() payable public{

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

    function withdraw(uint funds)public ifOwner{
        msg.sender.transfer(funds);
    }
    function getMoney(){

    }
}
```

### Output:



- Program:**

```
C:\Users\HP>geth attach ipc:\\.\pipe\geth.ipc
Welcome to the Geth JavaScript console!

instance: Geth/v1.10.26-stable-e5eb32ac/windows-amd64/go1.18.5
at block: 0 (Thu Jan 01 1970 05:30:00 GMT+0530 (IST))
datadir: D:\MCA\sem3\Blockchain\Private_Chain\chaindata
modules: admin:1.0 debug:1.0 engine:1.0 eth:1.0 ethash:1.0 miner:1.0 net
t:1.0 personal:1.0 rpc:1.0 txpool:1.0 web3:1.0

To exit, press ctrl-d or type exit
> personal.newAccount()
Passphrase:
Repeat passphrase:
"0x2c2cacffc74a77b000461838042125db1203e487"
> eth.accounts
["0x2c2cacffc74a77b000461838042125db1203e487"]
> eth.coinbase
"0x2c2cacffc74a77b000461838042125db1203e487"
> eth.getBalance(eth.accounts[0])
0
> miner.start()
null
> miner.stop()
null
> eth.getBalance(eth.accounts[0])
23000000000000000000
> personal.newAccount()
Passphrase:
Repeat passphrase:
"0x6680bf98a5b24a5f56cfd77c10fcd8d7a299bd88"
```

[illegible]

### Transfer funds, mine block and account balance before and after the mining of block

## Specific block details

**7. Create new truffle project with migration script, smart contract and configuration file. First compile it using truffle suite. Then connect it with personal private blockchain i.e. Ganache and deploy (migrate) smart contract on Ganache. Open truffle console and create instance of deployed (migrated) contract of Ganache. Then interact with smart contract using created instance. Take screenshots of all transaction's details and block details from Ganache**



## Output:

ACCOUNT INFORMATION

ACCOUNT ADDRESS

0×FC511bF6f4a825656C84AB6faF8172F664202663

PRIVATE KEY

d26ac88c2f42b4e4cd13594cb93427abec6d82c8ca09416b8ae11ac413d84674

Do not use this private key on a public blockchain; use it for development purposes only!

DONE

Import account

Imported accounts will not be associated with your originally created MetaMask account Secret Recovery Phrase. Learn more about imported accounts [here](#)

Select Type

Private Key

Enter your private key string here:

.....

Cancel

Import

Send

VaidyaAkshay

0xE8FE4e9f3624Eb66B51B7b191aAF274ECa5baa89

Asset:

ETH

Balance: 100 ETH

Amount:

18 ETH

No conversion rate available

Gas price (GWEI)

20

Gas limit

21000

Cancel

Next

Account 3

0xfC5...2663

100 ETH

Buy

Send

Swap

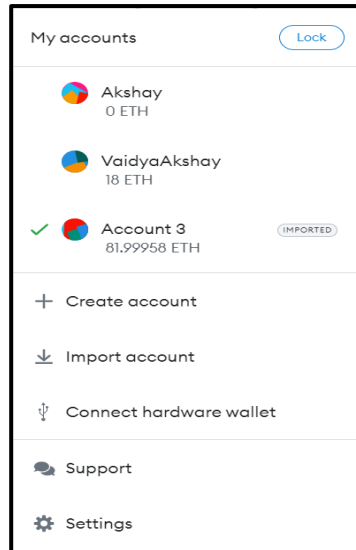
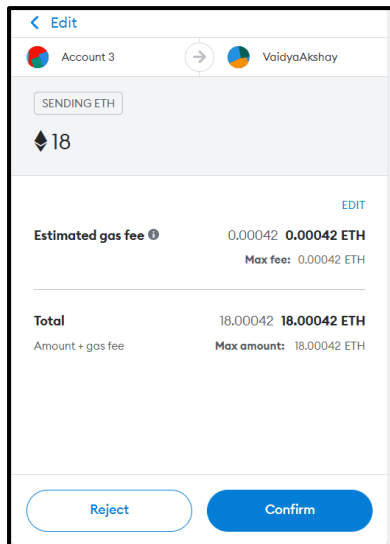
Assets

Activity

Portfolio site

You have no transactions

Need help? Contact [MetaMask support](#)



8. Create new truffle project with migration script, smart contract and configuration file. First compile it using truffle suite. Then connect it with personal private blockchain i.e. Ganache and deploy (migrate) smart contract on Ganache. Open truffle console and create instance of deployed (migrated) contract of Ganache. Then interact with smart contract using created instance. Take screenshots of all transaction's details and block details from Ganache.

## Output:

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  JUPYTER  COMMENTS

PS D:\Finolex\SEM 3\BlockChain\TrupleSuite\blockchain-toolkit> truffle compile

Compiling your contracts...
=====
> Compiling .\contracts\MyContract.sol
> Compiling .\contracts\MyContract.sol
> Artifacts written to D:\Finolex\SEM 3\BlockChain\TrupleSuite\blockchain-toolkit\build\contracts
> Compiled successfully using:
  - solc: 0.5.16+commit.9c3226ce.Emscripten.clang
PS D:\Finolex\SEM 3\BlockChain\TrupleSuite\blockchain-toolkit>

PS D:\Finolex\SEM 3\BlockChain\TrupleSuite\blockchain-toolkit> truffle migrate

Compiling your contracts...
=====
> Compiling .\contracts\MyContract.sol
> Compiling .\contracts\MyContract.sol
> Artifacts written to D:\Finolex\SEM 3\BlockChain\TrupleSuite\blockchain-toolkit\build\contracts
> Compiled successfully using:
  > transaction hash: 0xf87f7dbed388cb59e4a606cc3270863efe6c58c8dc7921f0d8ac69043a0ad3a6
  > Blocks: 0
  > contract address: 0xB282dDeAc746fF23fE8fEd7aC533F60482853Bc3
  > block number: 2
  > block timestamp: 1669773764
  > account: 0xE12501462869090590C91b3AB375F0DDFd27C71
  > balance: 99.99487344
  > gas used: 235328 (0x39740)
  > gas price: 20 gwei
  > value sent: 0 ETH
  > total cost: 0.00470656 ETH

  > Saving artifacts
  > Total cost: 0.00470656 ETH

Summary
=====
> Total deployments: 1
> Final cost: 0.00470656 ETH

PS D:\Finolex\SEM 3\BlockChain\TrupleSuite\blockchain-toolkit>

```

## Practical No:04

**Roll No:16**

[illegible]