FINOLEX ACADEMY OF MANAGEMENT AND TECHNOLOGY, RATNAGIRI

DEPARTMENT OF MCA

PRACTICAL NO. 04

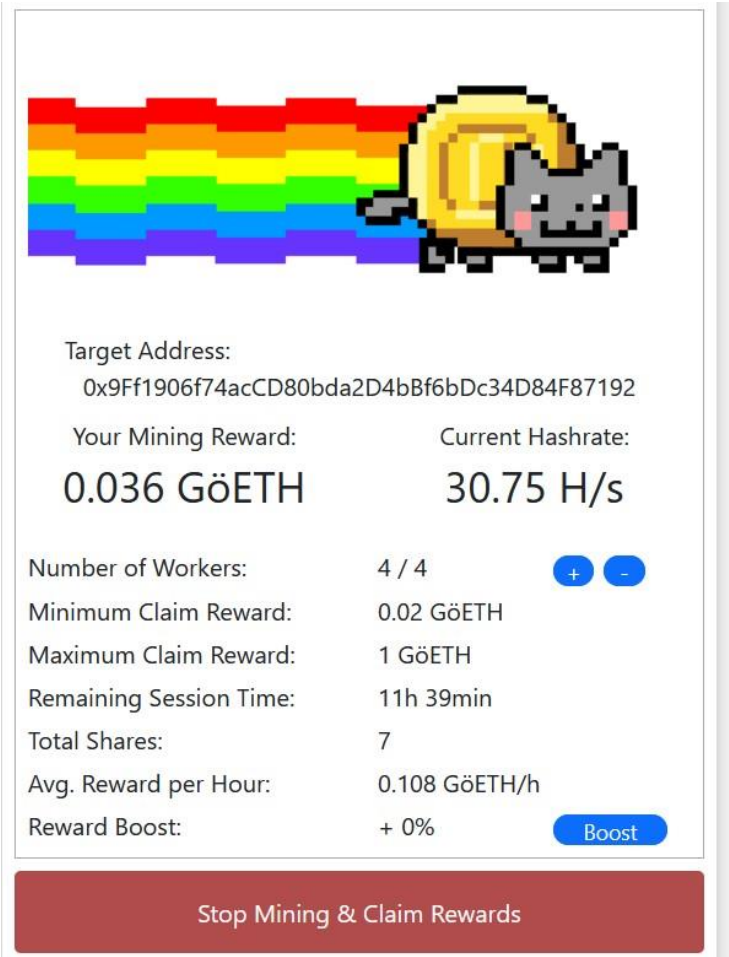
Ethereum

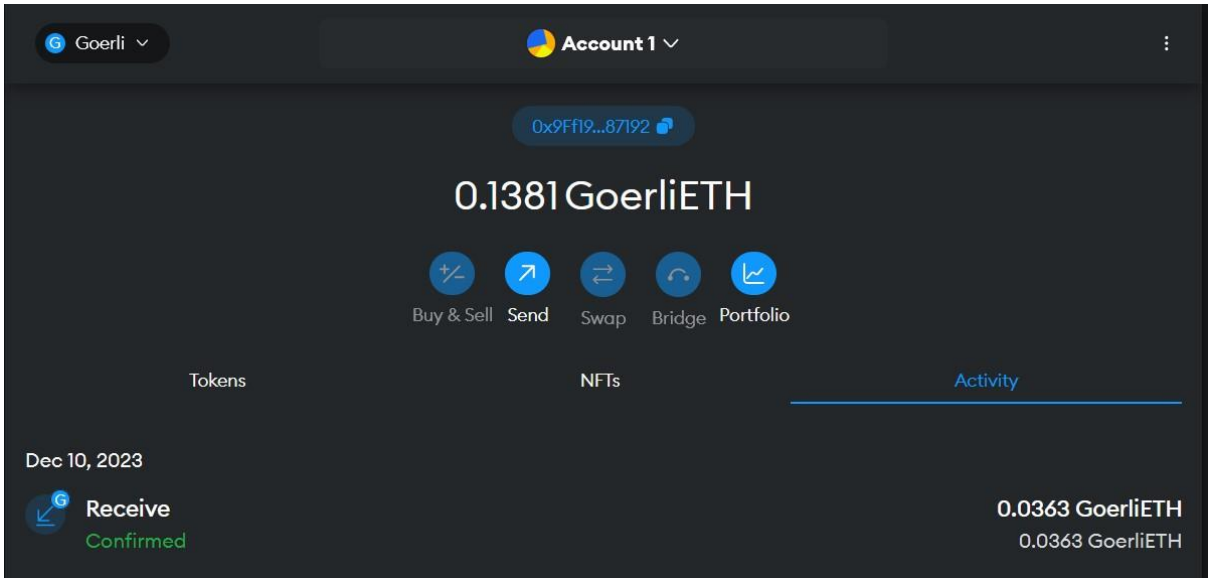
1. **Install the metamask in browser. Setup the metamask digital cryptocurrency wallet. Create multiple accounts in metamask and connect with one of the etherum blockchain test network. Perform the task buy ethers and send ethers from one account to another. Take the screenshots of created accounts,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

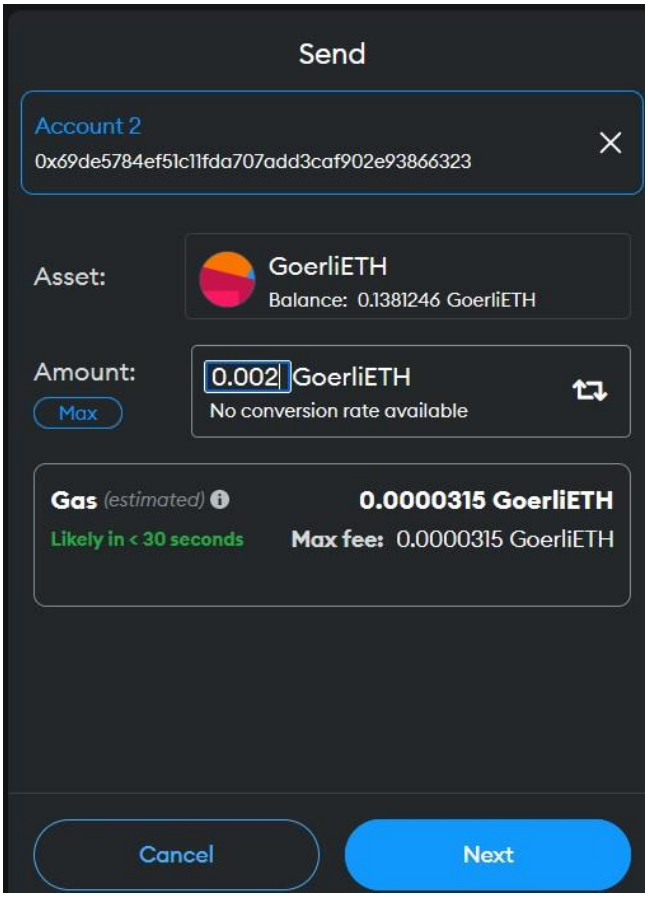
(Use following url to get free ether for Goerli Test Network:

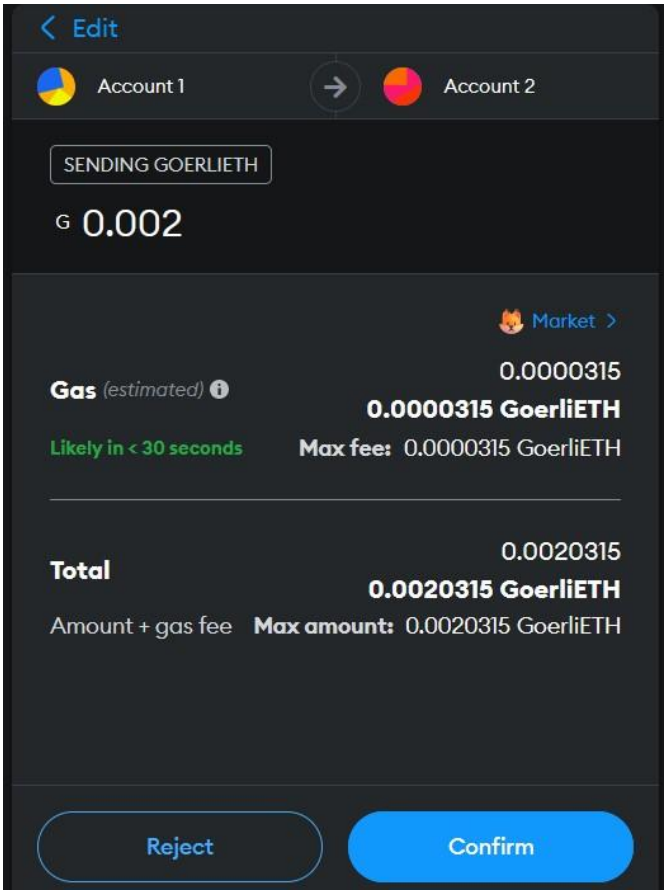
https://goerlifaucet.pk910.de/) Ans :

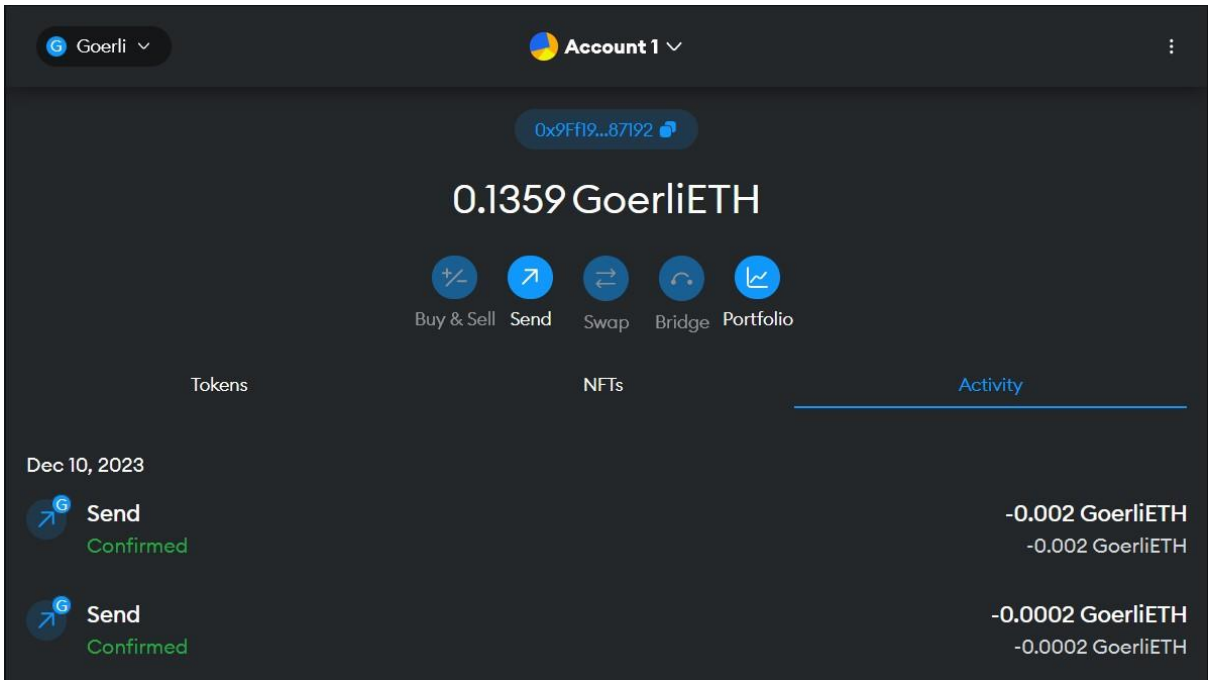
Output :

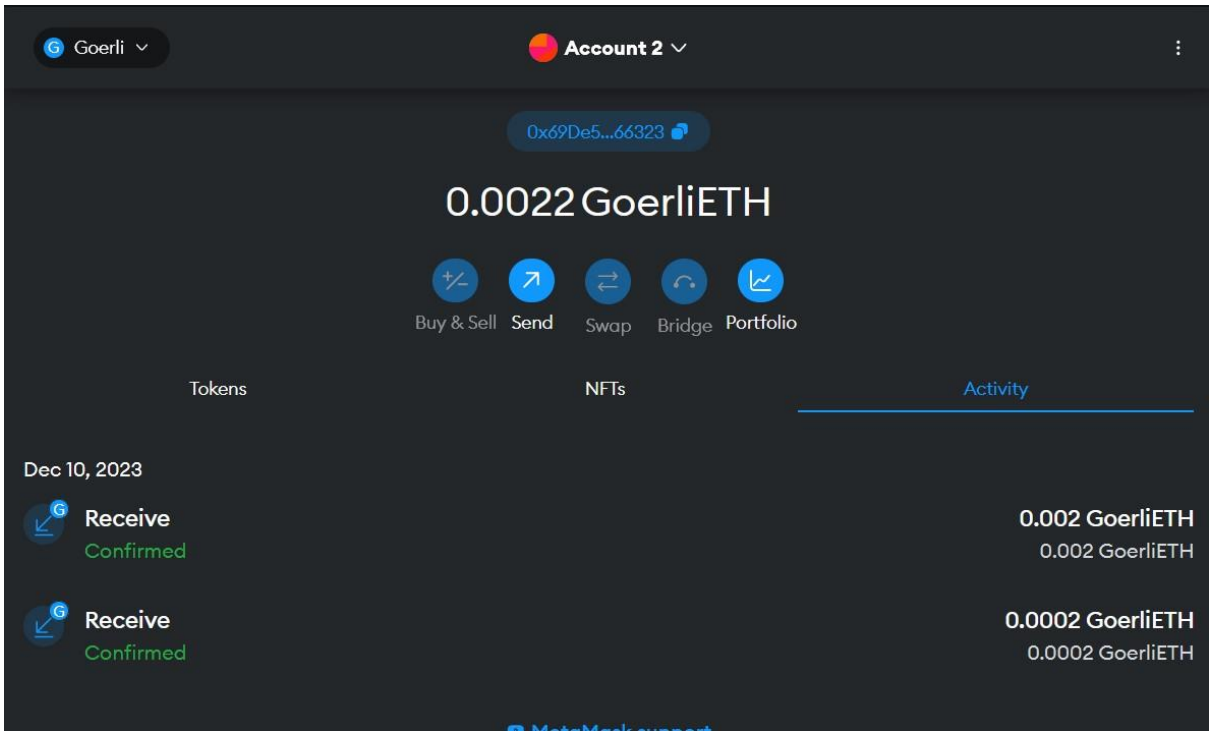



Goerli
Account 1
0x9Ff19...87192
0.1381 GoerliETH
Buy & Sell
Send
Swap
Bridge
Portfolio
Tokens
NFTs
Activity
Dec 10, 2023
Receive
Confirmed
0.0363 GoerliETH
0.0363 GoerliETH


Send
Account 2
0x69de5784ef51c11fda707add3caf902e93866323
Asset:
GoerliETH
Balance: 0.1381246 GoerliETH
Amount:
Max
0.002 GoerliETH
No conversion rate available
Gas (estimated)
0.0000315 GoerliETH
Likely in < 30 seconds
Max fee: 0.0000315 GoerliETH
Cancel
Next


Edit
Account 1
Account 2
SENDING GOERLIETH
G 0.002
Market
Gas (estimated)
0.0000315
0.0000315 GoerliETH
Likely in < 30 seconds
Max fee: 0.0000315 GoerliETH
Total
0.0020315
0.0020315 GoerliETH
Amount + gas fee
Max amount: 0.0020315 GoerliETH
Reject
Confirm

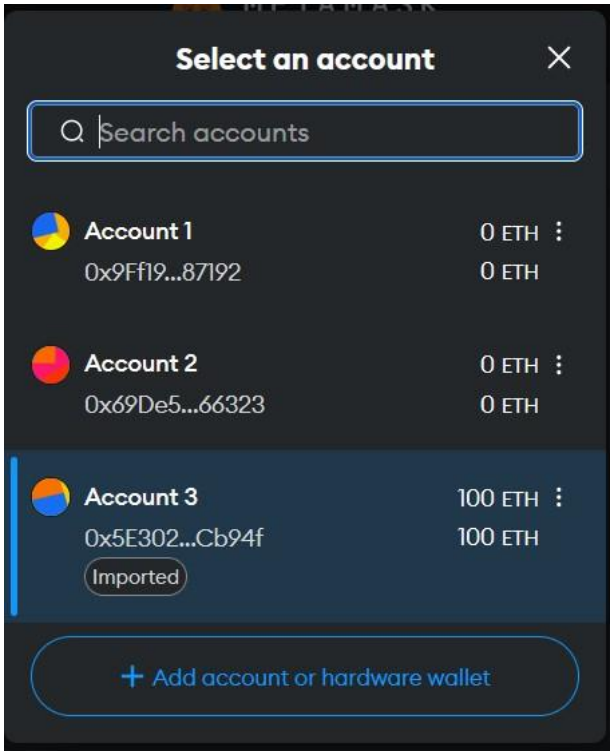


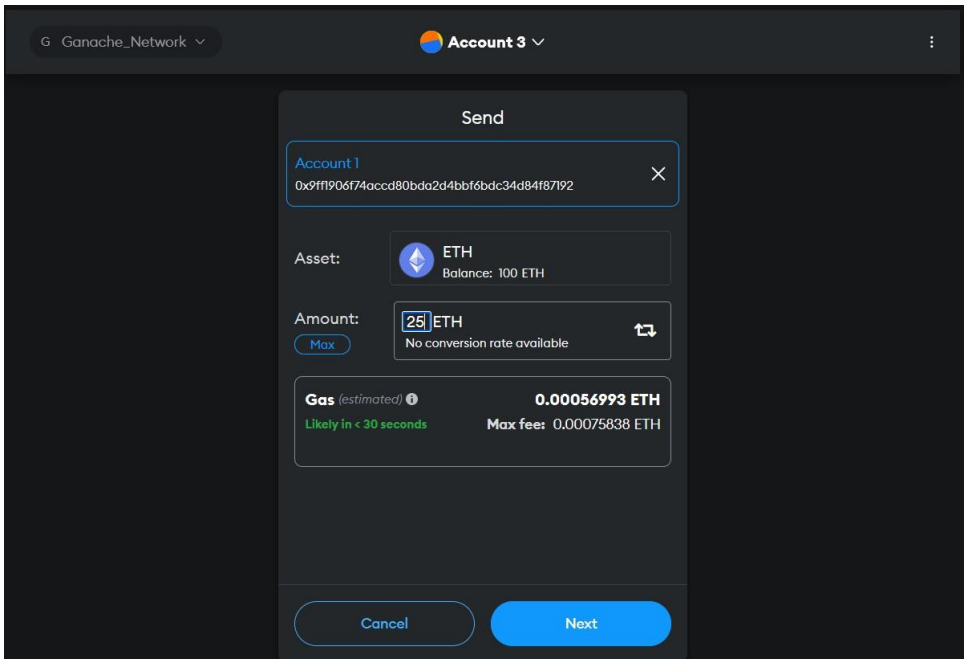


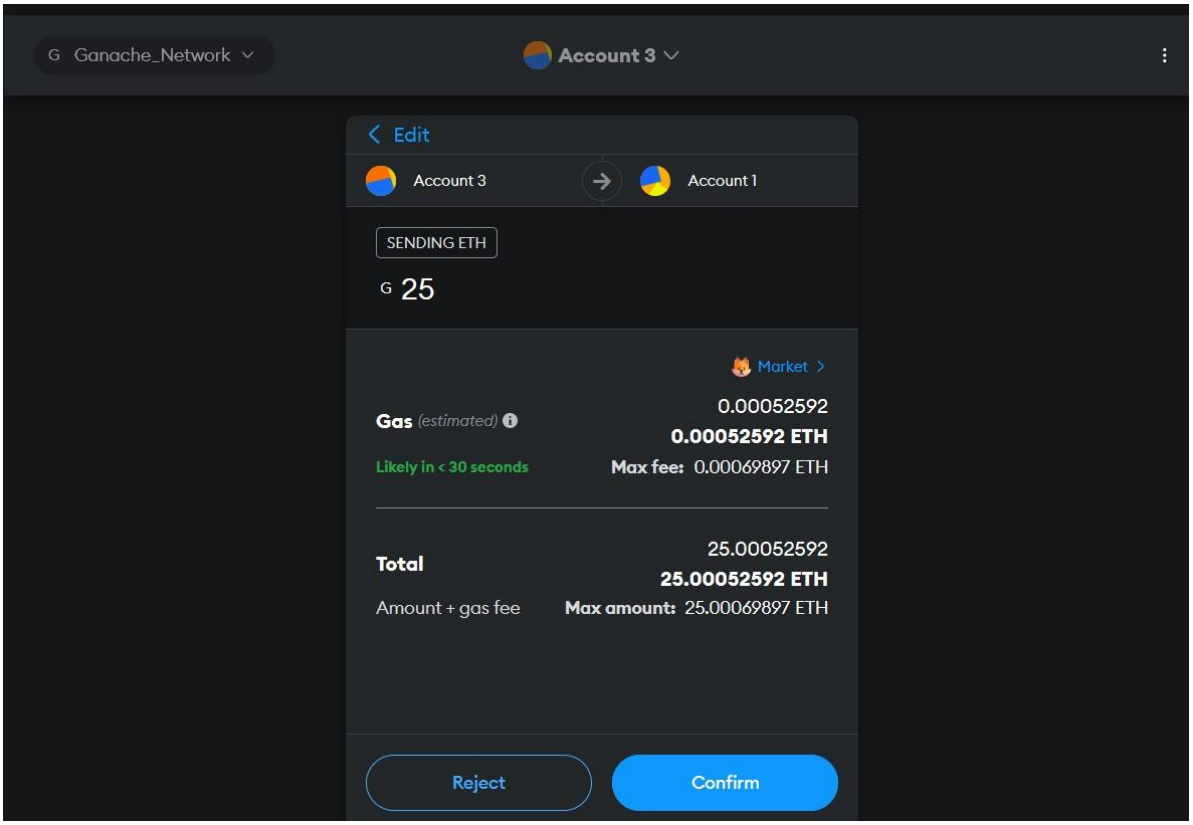
2. Start Ganache (your personal private blockchain network). Connect

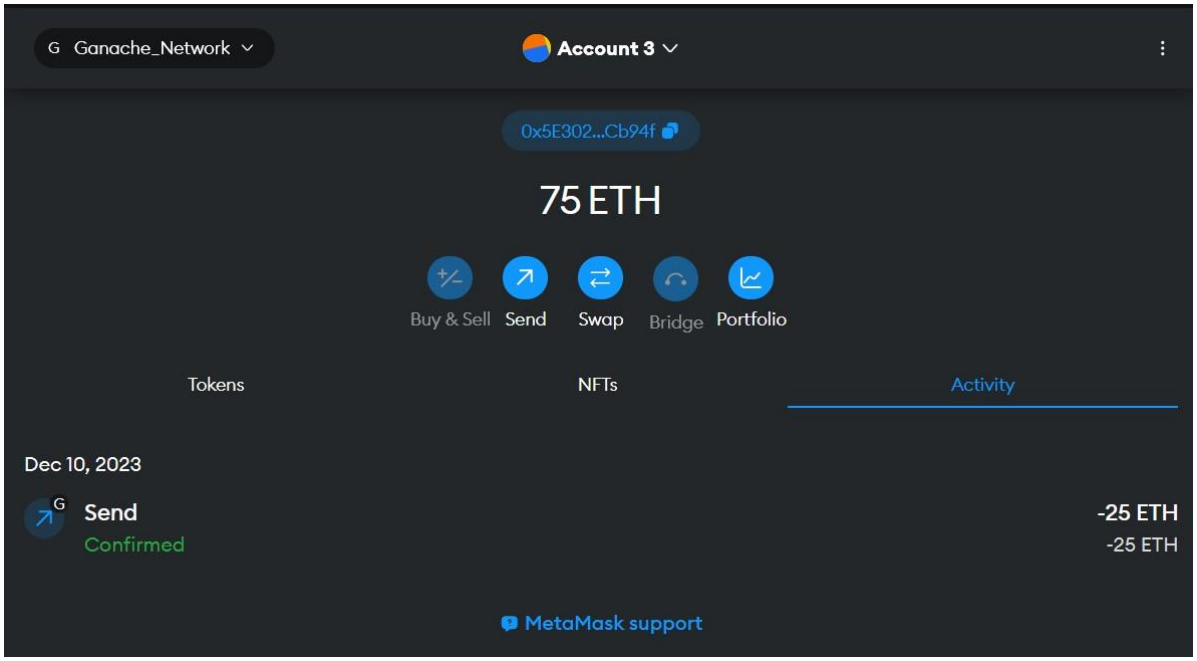
Ganache with MetaMask and import the account from Ganache to MetaMask. Transfer funds from imported account to other account of MetaMask. Take the screenshots of created accounts, account assets and account transactions which showing the details of transaction from MetaMask and Ganache interface. GoEthereum(Geth) Ans :

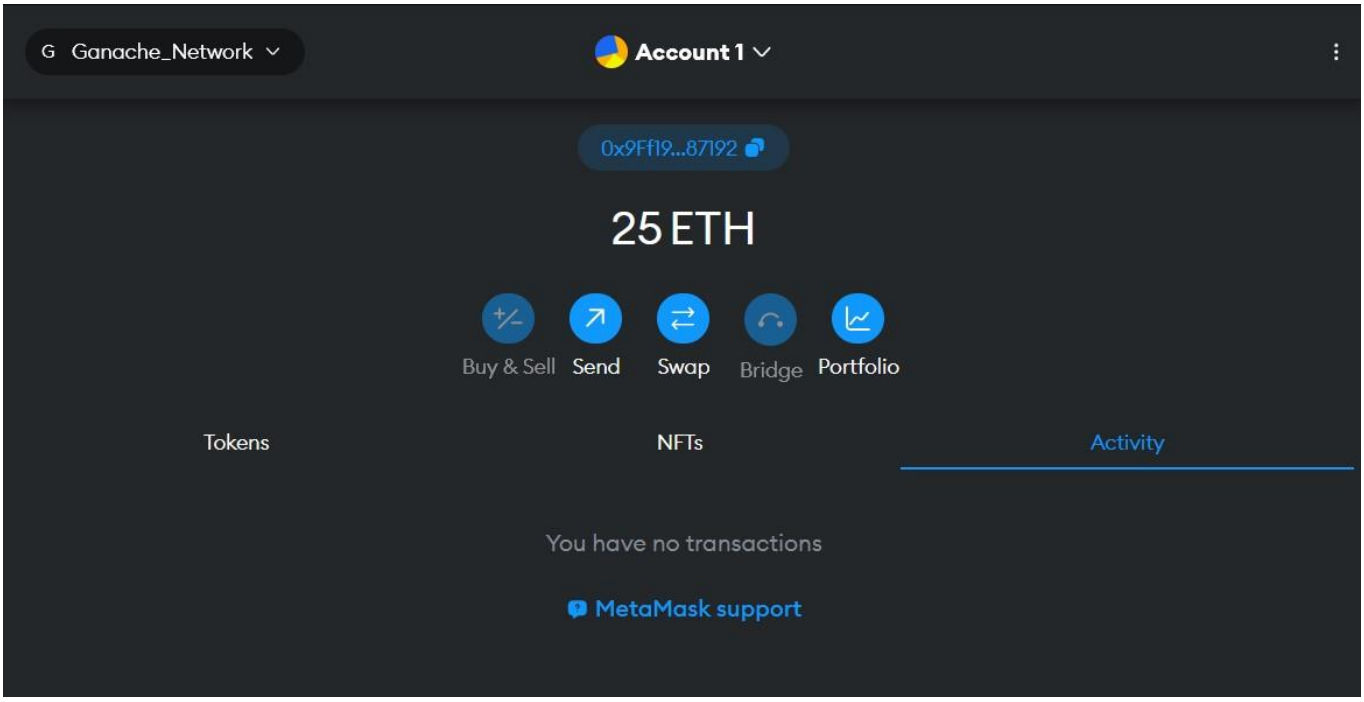
- **Output :**

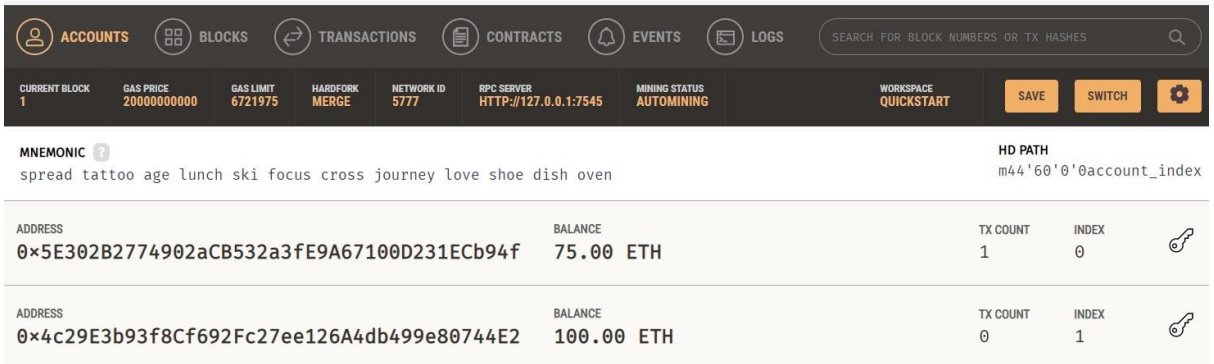





Ganache_Network
Account 3
Edit
Account 3
Account 1
SENDING ETH
25
Market
Gas (estimated)
0.00052592
0.00052592 ETH
Likely in < 30 seconds
Max fee: 0.00069897 ETH
Total
25.00052592
25.00052592 ETH
Amount + gas fee
Max amount: 25.00069897 ETH
Reject
Confirm


Ganache_Network
Account 3
0x5E302...Cb94f
75 ETH
Buy & Sell
Send
Swap
Bridge
Portfolio
Tokens
NFTs
Activity
Dec 10, 2023
Send
Confirmed
-25 ETH
-25 ETH
MetaMask support


Ganache_Network
Account 1
0x9Ff19...87192
25 ETH
Buy & Sell
Send
Swap
Bridge
Portfolio
Tokens
NFTs
Activity
You have no transactions
MetaMask support

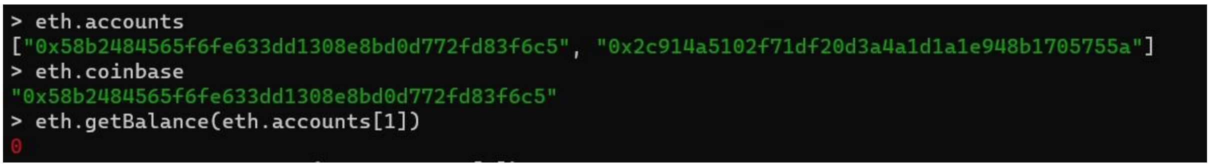

ACCOUNTS
BLOCKS
TRANSACTIONS
CONTRACTS
EVENTS
LOGS
SEARCH FOR BLOCK NUMBERS OR TX HASHES
CURRENT BLOCK 1
GAS PRICE 20000000000
GAS LIMIT 6721975
HARDFORK MERGE
NETWORK ID 5777
RPC SERVER HTTP://127.0.0.1:7545
MINING STATUS AUTOMINING
WORKSPACE QUICKSTART
SAVE
SWITCH
MNEMONIC
spread tattoo age lunch ski focus cross journey love shoe dish oven
HD PATH
m44'60'0'0account_index
ADDRESS
0x5E302B2774902aCB532a3fE9A67100D231ECb94f
BALANCE
75.00 ETH
TX COUNT
1
INDEX
0
ADDRESS
0x4c29E3b93f8Cf692Fc27ee126A4db499e80744E2
BALANCE
100.00 ETH
TX COUNT
0
INDEX
1

3. **Create Ethereum node using Geth (GoEthereum) and create genesis block and create your personal private Ethereum blockchain. And use IPC to interact with Geth node to perform following task: create account, transfer funds using send transaction, mine the block, show the account balance before and after the mining the block, show the specific block details and access chain details. Remix IDE – Injected Provider - Public Test Network (Goerli, Sepolia) or Ganache Ans :**

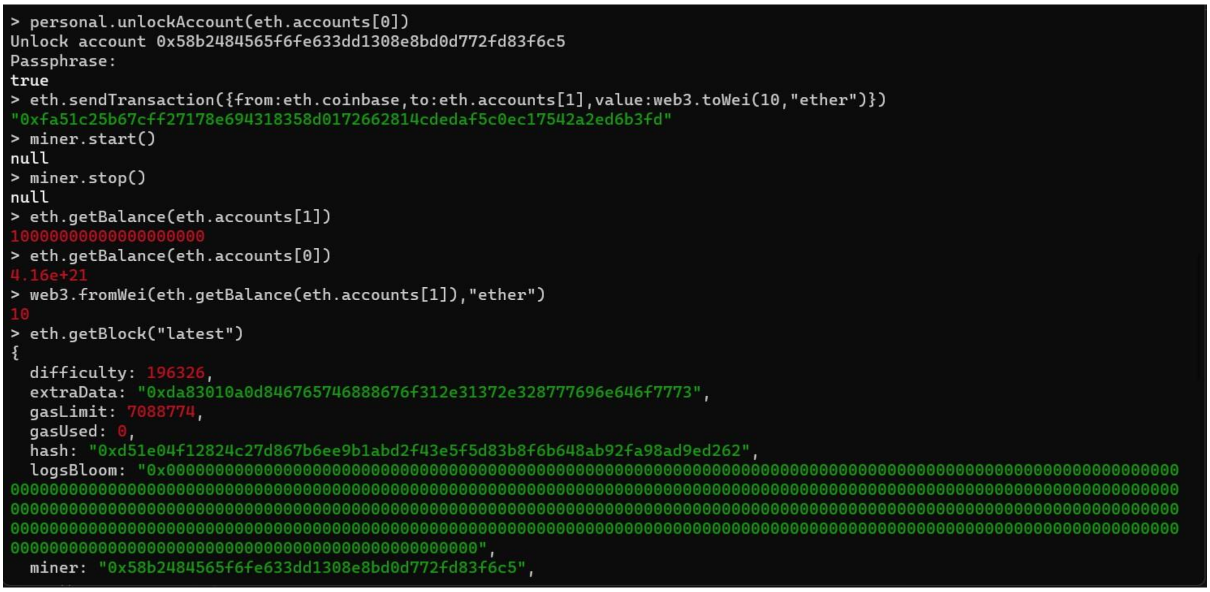
- **Output :**

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Tanuja Tabib\chaindata> geth --datadir=./chaindata/ init ./genesis.json
INFO [12-13|09:41:39.300] Maximum peer count                       ETH=50 LES=0 total=50
INFO [12-13|09:41:39.302] Set global gas cap                       cap=50,000,000
INFO [12-13|09:41:39.303] Allocated cache and file handles         database="C:\\Users\\Tanuja Tabib\\chaindata\\chainda
ta\\geth\\chaindata" cache=16.00MiB handles=16
INFO [12-13|09:41:39.311] Writing custom genesis block
INFO [12-13|09:41:39.312] Persisted trie from memory database      nodes=0 size=0.00B time="560.4µs" gcnodes=0 gcsize=0.
00B gctime=0s livenodes=1 livesize=0.00B
INFO [12-13|09:41:39.314] Successfully wrote genesis state         database=chaindata hash=2fb1a7..f0181a
INFO [12-13|09:41:39.314] Allocated cache and file handles         database="C:\\Users\\Tanuja Tabib\\chaindata\\chainda
ta\\geth\\lightchaindata" cache=16.00MiB handles=16
INFO [12-13|09:41:39.322] Writing custom genesis block
INFO [12-13|09:41:39.322] Persisted trie from memory database      nodes=0 size=0.00B time=0s        gcnodes=0 gcsize=0.
00B gctime=0s livenodes=1 livesize=0.00B
INFO [12-13|09:41:39.324] Successfully wrote genesis state         database=lightchaindata hash=2fb1a7..f0181a
PS C:\Users\Tanuja Tabib\chaindata> geth --datadir=./chaindata/
INFO [12-13|09:41:42.647] Starting Geth on Ethereum mainnet...
INFO [12-13|09:41:42.648] Bumping default cache on mainnet         provided=1024 updated=4096
INFO [12-13|09:41:42.650] Maximum peer count                       ETH=50 LES=0 total=50
WARN [12-13|09:41:42.651] Sanitizing cache to Go's GC limits       provided=4096 updated=2611
INFO [12-13|09:41:42.651] Set global gas cap                       cap=50,000,000
INFO [12-13|09:41:42.651] Allocated trie memory caches             clean=391.00MiB dirty=652.00MiB
INFO [12-13|09:41:42.652] Allocated cache and file handles         database="C:\\Users\\Tanuja Tabib\\chaindata\\chainda
ta\\geth\\chaindata" cache=1.27GiB handles=8192
INFO [12-13|09:41:42.685] Opened ancient database                  database="C:\\Users\\Tanuja Tabib\\chaindata\\chainda
```

```
Command Prompt - geth atta
Microsoft Windows [Version 10.0.22621.2715]
(c) Microsoft Corporation. All rights reserved.

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

instance: Geth/v1.10.13-stable-7a0c19f8/windows-amd64/go1.17.2
at block: 0 (Thu Jan 01 1970 05:30:00 GMT+0530 (IST))
 datadir: C:\Users\Tanuja Tabib\chaindata\chaindata
 modules: admin:1.0 debug:1.0 eth:1.0 ethash:1.0 miner:1.0 net: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:
"0x58b2484565f6fe633dd1308e8bd0d772fd83f6c5"
> eth.accounts
["0x58b2484565f6fe633dd1308e8bd0d772fd83f6c5"]
> eth.coinbase
"0x58b2484565f6fe633dd1308e8bd0d772fd83f6c5"
> eth.getBalance(eth.accounts[0])
0
> miner.start()
null
> eth.getBalance(eth.accounts[0])
280000000000000000000
> personal.newAccount()
Passphrase:
Repeat passphrase:
"0x2c914a5102f71df20d3a4a1d1a1e948b1705755a"
```



```
> eth.accounts
["0x58b2484565f6fe633dd1308e8bd0d772fd83f6c5", "0x2c914a5102f71df20d3a4a1d1a1e948b1705755a"]
> eth.coinbase
"0x58b2484565f6fe633dd1308e8bd0d772fd83f6c5"
> eth.getBalance(eth.accounts[1])
0
```



```
> personal.unlockAccount(eth.accounts[0])
Unlock account 0x58b2484565f6fe633dd1308e8bd0d772fd83f6c5
Passphrase:
true
> eth.sendTransaction({from:eth.coinbase,to:eth.accounts[1],value:web3.toWei(10,"ether")})
"0xfa51c25b67cff27178e694318358d0172662814cdedaf5c0ec17542a2ed6b3fd"
> miner.start()
null
> miner.stop()
null
> eth.getBalance(eth.accounts[1])
10000000000000000000
> eth.getBalance(eth.accounts[0])
4.16e+21
> web3.fromWei(eth.getBalance(eth.accounts[1]),"ether")
10
> eth.getBlock("latest")
{
  difficulty: 196326,
  extraData: "0xda83010a0d846765746888676f312e31372e328777696e646f7773",
  gasLimit: 7088774,
  gasUsed: 0,
  hash: "0xd51e04f12824c27d867b6ee9b1abd2f43e5f5d83b8f6b648ab92fa98ad9ed262",
  logsBloom: "0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000",
  miner: "0x58b2484565f6fe633dd1308e8bd0d772fd83f6c5",
```

```
  mixHash: "0x3b9fab58ad4b1424a1a3beb76fe1e9004418543e5470ce10f669f0eef451f0d1",
  nonce: "0x2d68d060f699981b",
  number: 834,
  parentHash: "0x399c0b0c0668ad96ab091573efd5fdfba70c9d9f9d0b08e39975e806e10fd47f",
  receiptsRoot: "0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
  sha3Uncles: "0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d49347",
  size: 540,
  stateRoot: "0xfda62bd4a71dea0019d959ea1006abc01985373b7cd119b0e45d50536690b2da",
  timestamp: 1702441754,
  totalDifficulty: 134820795,
  transactions: [],
  transactionsRoot: "0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
  uncles: []
}
> eth.getBlock(35)
{
  difficulty: 133250,
  extraData: "0xda83010a0d846765746888676f312e31372e328777696e646f7773",
  gasLimit: 3250719,
  gasUsed: 0,
  hash: "0x16cad8040835cd2bbdba2fb87a7567535057651812afa28121dd191c809c7cfb",
  logsBloom: "0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000",
  miner: "0x58b2484565f6fe633dd1308e8bd0d772fd83f6c5",
  mixHash: "0x3583a33a2a9056fa13950074679b49a0969c76e3b6cacd486d67b61aa2d16e76",
  nonce: "0x26e589ed289485a0",
  number: 35,
```

```
  parentHash: "0x01550bb219d1191070c7fdbf034e8540554fd81ab11e307197ac9dad61d7abcc",
  receiptsRoot: "0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
  sha3Uncles: "0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d49347",
  size: 538,
  stateRoot: "0x00891736487e1b0c09fb043fabeef74b8db64707a232d87277313ef0285cce44",
  timestamp: 1702440861,
  totalDifficulty: 4756675,
  transactions: [],
  transactionsRoot: "0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
  uncles: []
}
```

4. Write a solidity smart contract for performing following task using remixIDE and deployed it on public test network – Goerli / Sapolia using Injected provider environment.

a. To transfer funds (ethers) from user account to contract account.

b. To withdraw funds (ethers) from contract account to user account.

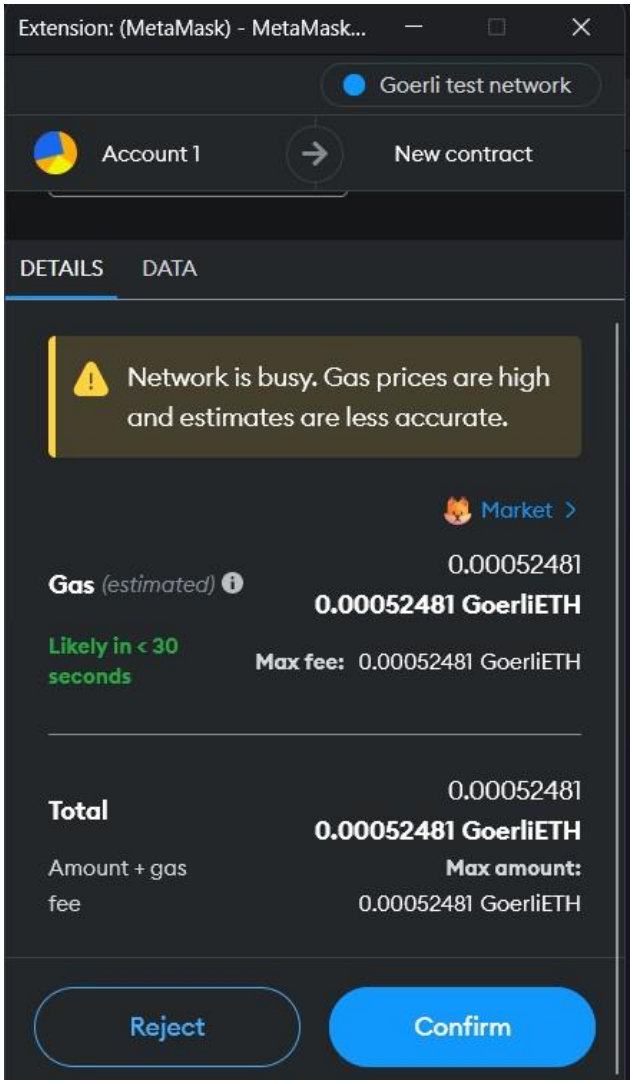
c. To apply restriction that only owner of the contract can withdraw funds (ethers) from contract account to his/her user account.

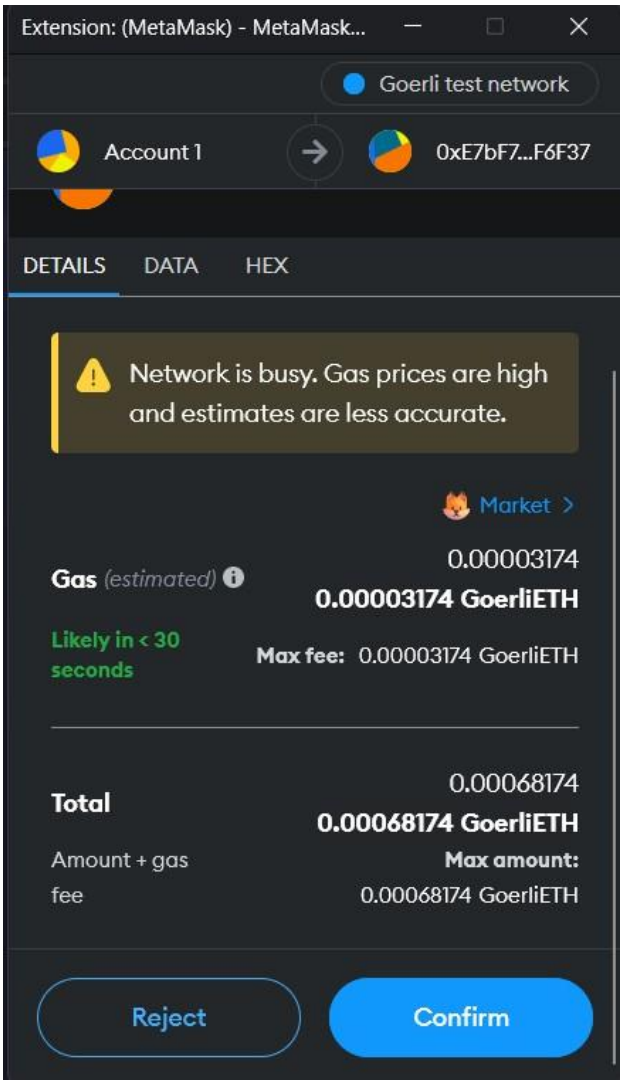
Ans :

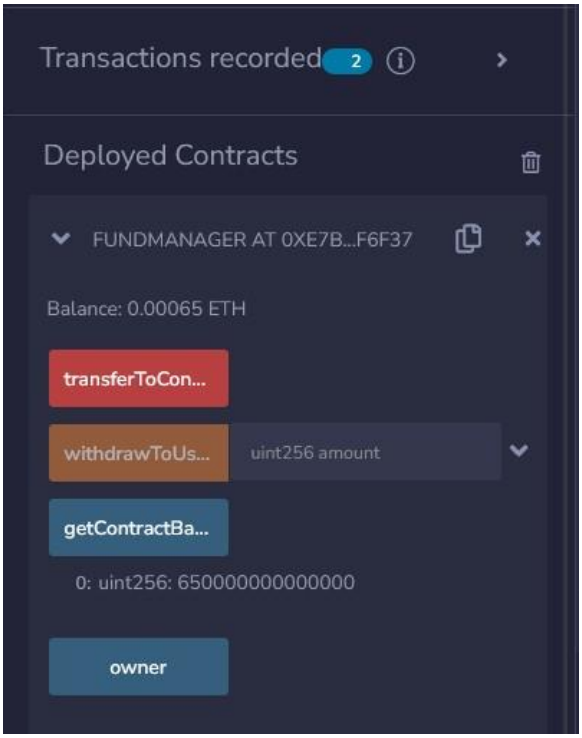
- **Program :**
  FundManager.sol

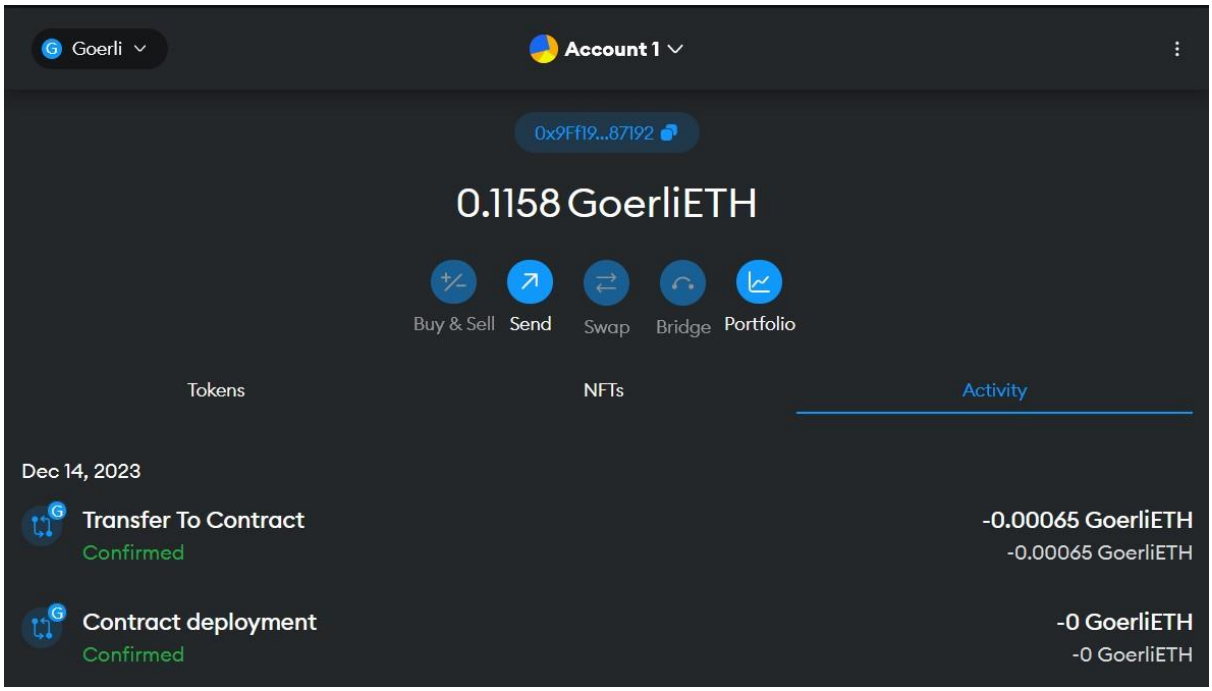
```
// SPDX-License-Identifier: MIT pragma solidity ^0.8.0; contract
FundManager {     address public owner;     modifier onlyOwner() {
require(msg.sender == owner, "Only owner can call this function");
        _;
    }     constructor() {
owner = msg.sender;
    }
    // Function to transfer funds from user account to contract account
function transferToContract() external payable {
        // No logic needed, funds are transferred with the transaction
    }
    // Function to withdraw funds from contract account to user account
function withdrawToUser(uint256 amount) external onlyOwner {
require(amount > 0, "Amount must be greater than 0");
require(address(this).balance >= amount, "Insufficient funds in the
contract");         payable(msg.sender).transfer(amount);
    }
    // Function to get the contract's balance
    function getContractBalance() external view returns (uint256) {
return address(this).balance;
    } }
```

- **Output :**


Extension: (MetaMask) - MetaMask...
Goerli test network
Account 1
New contract
DETAILS
DATA
Network is busy. Gas prices are high and estimates are less accurate.
Market
Gas (estimated)
0.00052481
0.00052481 GoerliETH
Likely in < 30 seconds
Max fee: 0.00052481 GoerliETH
Total
0.00052481
0.00052481 GoerliETH
Amount + gas fee
Max amount:
0.00052481 GoerliETH
Reject
Confirm


Extension: (MetaMask) - MetaMask...
Goerli test network
Account 1
0xE7bF7...F6F37
DETAILS
DATA
HEX
Network is busy. Gas prices are high and estimates are less accurate.
Market
Gas (estimated)
0.00003174
0.00003174 GoerliETH
Likely in < 30 seconds
Max fee: 0.00003174 GoerliETH
Total
0.00068174
0.00068174 GoerliETH
Amount + gas fee
Max amount:
0.00068174 GoerliETH
Reject
Confirm


Transactions recorded 2
Deployed Contracts
FUNDMANAGER AT 0XE7B...F6F37
Balance: 0.00065 ETH
transferToCon...
withdrawToUs...
uint256 amount
getContractBa...
0: uint256: 650000000000000
owner

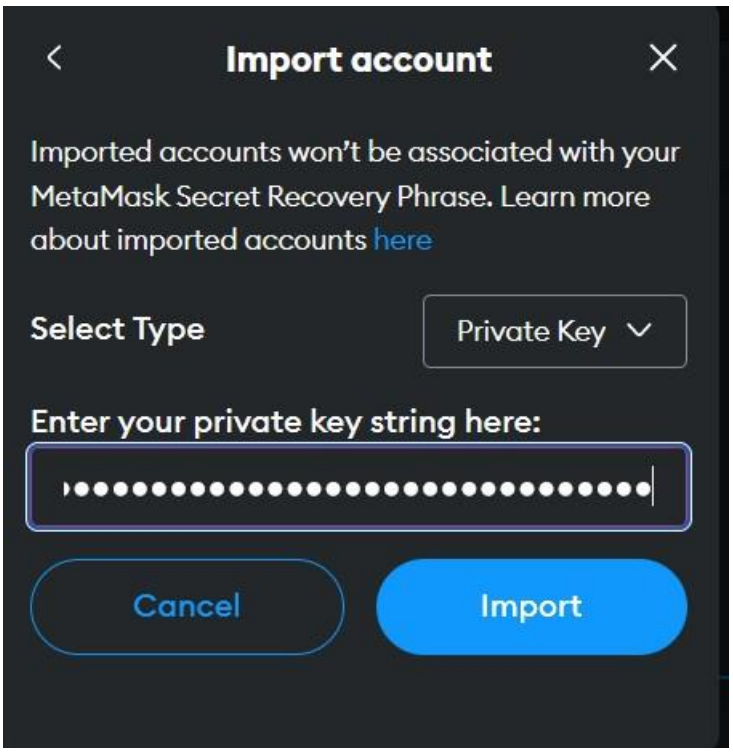

Goerli
Account 1
0x9Ff19...87192
0.1158 GoerliETH
Buy & Sell
Send
Swap
Bridge
Portfolio
Tokens
NFTs
Activity
Dec 14, 2023
Transfer To Contract
Confirmed
-0.00065 GoerliETH
-0.00065 GoerliETH
Contract deployment
Confirmed
-0 GoerliETH
-0 GoerliETH

5. Write a smart contract to calculate the compound interest and deploy it on Ganache using injected provider. Truffle – Ganache Ans :

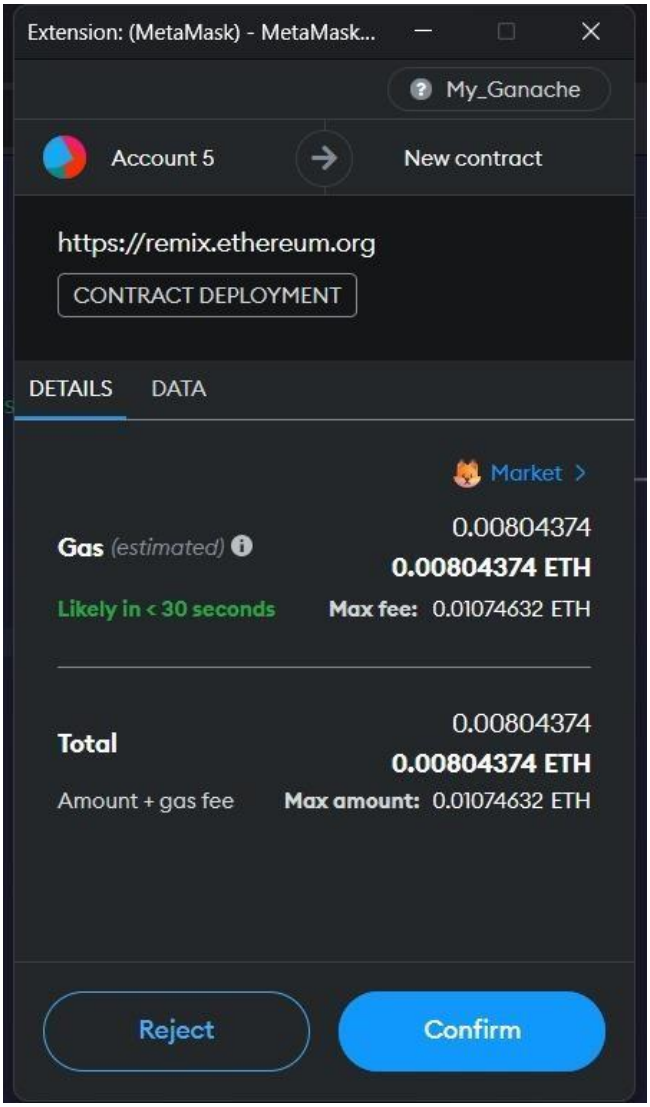
- **Program :**
  Orange.sol

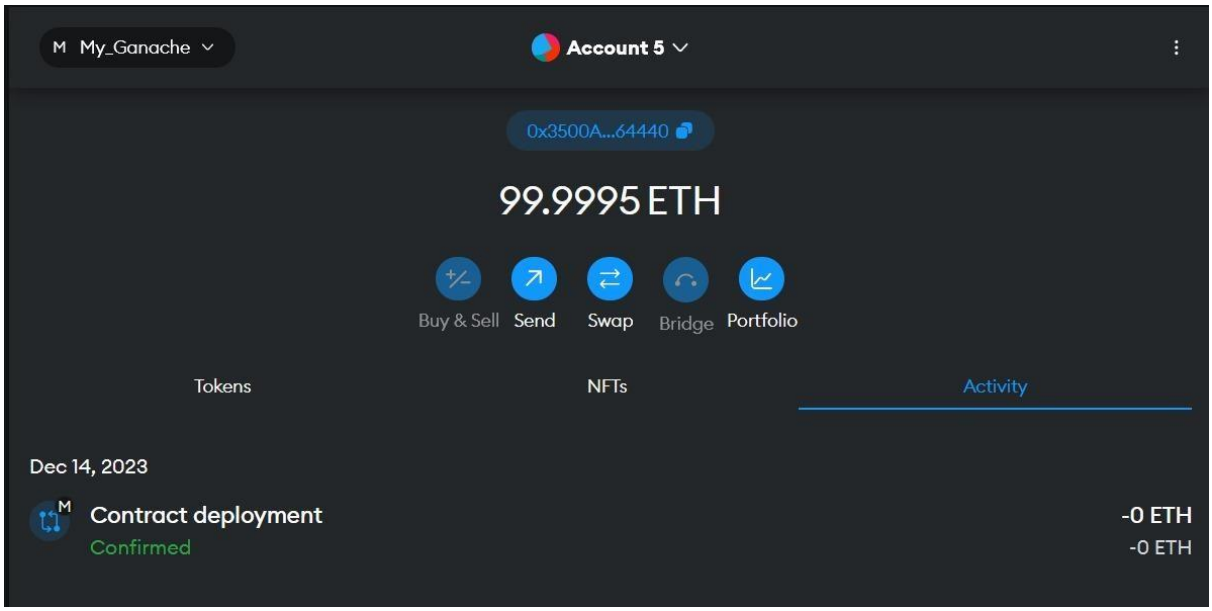
```
// SPDX-License-Identifier: MIT
pragma solidity >=0.8.9;

contract Orange {
   function compoundInterest(uint _principal, uint _periods, uint _rate)
external pure returns(uint) {        uint yield;        for(uint i=0; i< _periods;
i++){            yield = _principal * _rate / 100;
          _principal += yield;
      }        return
yield;
   }
}
```

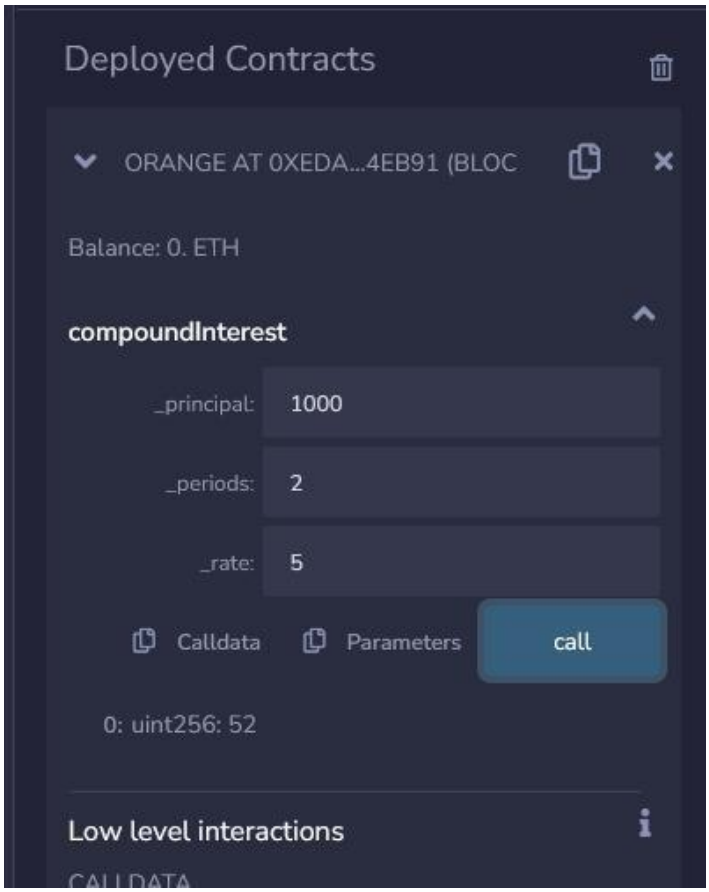
- **Output :**

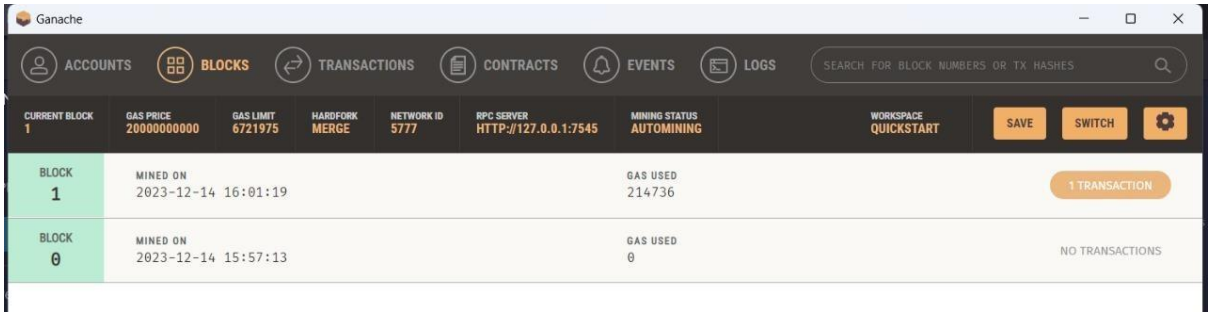


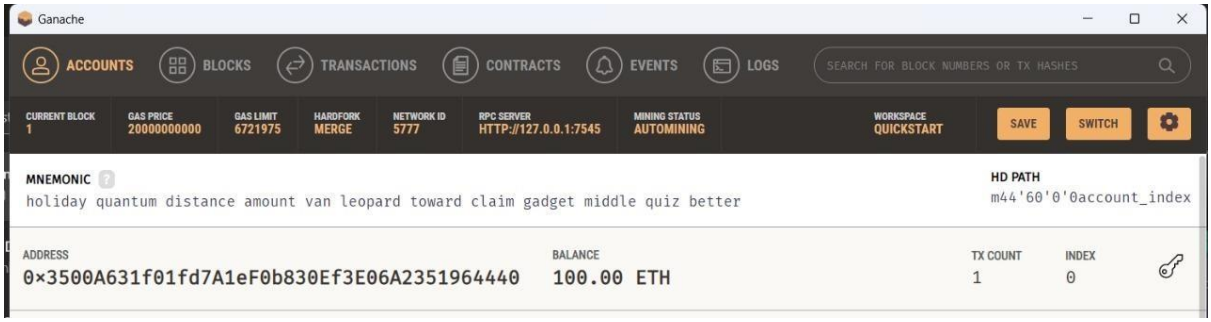

DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Provider - MetaMask
Custom (5777) network
ACCOUNT
0x350...64440 (100 ether)
GAS LIMIT
3000000
VALUE
0
Wei
CONTRACT
Orange - contracts/CompoundInterest.
evm version: shanghai
Deploy
Publish to IPFS
Home
CompoundInterest.sol
// SPDX-License-Identifier: MIT
pragma solidity >=0.8.9;
contract Orange {
function compoundInterest(uint _principal, uint _periods, uint _rate) external pure returns(uint) {
Infinite gas
uint yield;
for(uint i=0; i< _periods; i++){
yield = _principal * _rate / 100;
_principal += yield;
}
return yield;
}
}


Extension: (MetaMask) - MetaMask...
My_Ganache
Account 5
New contract
https://remix.ethereum.org
CONTRACT DEPLOYMENT
DETAILS
DATA
Market
Gas (estimated)
0.00804374
0.00804374 ETH
Likely in < 30 seconds
Max fee: 0.01074632 ETH
Total
0.00804374
0.00804374 ETH
Amount + gas fee
Max amount: 0.01074632 ETH
Reject
Confirm


My_Ganache
Account 5
0x3500A...64440
99.9995 ETH
Buy & Sell
Send
Swap
Bridge
Portfolio
Tokens
NFTs
Activity
Dec 14, 2023
Contract deployment
Confirmed
-0 ETH
-0 ETH


Deployed Contracts
ORANGE AT 0XEDA...4EB91 (BLOC
Balance: 0. ETH
compoundInterest
_principal: 1000
_periods: 2
_rate: 5
Calldata
Parameters
call
0: uint256: 52
Low level interactions





6. Build and test decentralized application (Dapp) for Election Voting System on the local Ethereum Blockchain Network Ganache using truffle suite. Ans :

- **Program :**

Election.sol

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.5.16;
contract Election { struct
Candidate { uint id; string name;
uint voteCount;
}
mapping(uint => Candidate) public candidates;
mapping(address => bool) public voters; uint
public candidatesCount; constructor() {
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(_candidateId > 0 && _candidateId <= candidatesCount, "Invalid candidate ID");
require(!voters[msg.sender], "You have already voted");
voters[msg.sender] = true; candidates[_candidateId].voteCount++;
}
}
```

- **Output :**

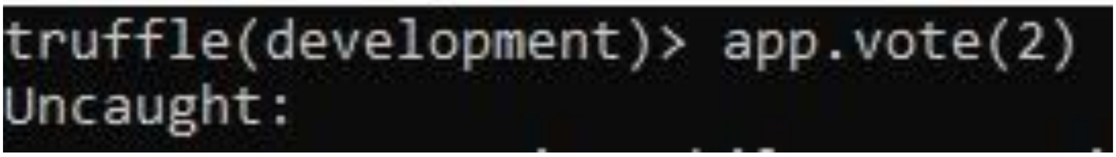
```
C:\election-truffle>truffle compile

Compiling your contracts...
===========================
> Compiling .\contracts\Election.sol
> Artifacts written to C:\election-truffle\build\contracts
> Compiled successfully using:
   - solc: 0.5.16+commit.9c3226ce.Emscripten.clang
```

```
C:\election-truffle>truffle migrate

Compiling your contracts...
===========================
> Compiling .\contracts\Election.sol
> Artifacts written to C:\election-truffle\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)


2_deploy_contracts.js
=====================

   Deploying 'Election'
   --------------------
   > transaction hash:    0x390899b1bd2a81483ee313072ae3532ec3a56fbee6dd74241e6e5f4deb4e082e
   > Blocks: 0            Seconds: 0
   > contract address:    0xC30F3E6a0877577B6d746617E62A8764dB3F3276
   > block number:        2
   > block timestamp:     1702465534
   > account:             0x7B213663aDe9dC5c53C263A29180679bC026bC32
   > balance:             99.998032037347978318
   > gas used:            357234 (0x57372)
   > gas price:           3.273325473 gwei
   > value sent:          0 ETH
   > total cost:          0.001169343152021682 ETH

   > Saving artifacts
   -------------------------------------
   > Total cost:     0.001169343152021682 ETH
```

```
Summary
=======
> Total deployments:   1
> Final cost:          0.001169343152021682 ETH
```

```
C:\election-truffle>truffle console
truffle(development)> Election.deployed().then((instance)=>{app=instance})
undefined
```

```
truffle(development)> app.vote(1)
{
  tx: '0x9d970afe0fc2f38c87dbd07949cb219791221f215fa5ef39063aeb3238d38aa5',
  receipt: {
    transactionHash: '0x9d970afe0fc2f38c87dbd07949cb219791221f215fa5ef39063aeb3238d38aa5',
    transactionIndex: 0,
    blockNumber: 3,
    blockHash: '0x18b3907ce4b64dc614195b13431eefcbdaee48561bbf5ec4daeb8100754ea50b',
    from: '0x7b213663ade9dc5c53c263a29180679bc026bc32',
    to: '0xc30f3e6a0877577b6d746617e62a8764db3f3276',
    cumulativeGasUsed: 68107,
    gasUsed: 68107,
    contractAddress: null,
    logs: [],
truffle(development)>
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000',
    status: true,
    effectiveGasPrice: 3186934232,
    type: '0x2',
    rawLogs: []
  },
  logs: []
}
```



```
truffle(development)> app.vote(2)
Uncaught:
```

```
  code: -32000,
  data: {
    hash: null,
    programCounter: 543,
    result: '0x08c379a000000000000000000000000000000000000000000000000000000000000002000000000000000000000000000000000000000000000000000000000000000016596f7520
616c726561647920766f74656400000000000000000000000000',
    reason: 'You have already voted',
    message: 'revert'
  },
  reason: 'You have already voted',
```

7. Build and test decentralized application, (Dapp) for Banking System on the local Ethereum Blockchain Network Ganache using truffle suite.

Ans :

- **Program :**

Banking.sol

```
// SPDX-License-Identifier: MIT pragma solidity >=0.6<0.9;
contract Banking {     mapping (address => uint) public
userAccount; // Balance     mapping (address => bool) public
userExists;     function createAcc() public payable returns (string
memory) {         require(!userExists[msg.sender], 'Account
already created');         if (msg.value == 0) {
userAccount[msg.sender] = 0;
        } else {
            userAccount[msg.sender] = msg.value;
        }
        userExists[msg.sender] = true;
return 'Account created!';
    }
    function deposit() public payable returns (string memory) {
require(userExists[msg.sender], 'Account is not created');
require(msg.value > 0, 'Value for deposit is not zero');
userAccount[msg.sender] += msg.value;         return
'Deposited successfully';
    }

    function withdraw(uint amount) public returns(string memory) {
require(userExists[msg.sender], 'Account is not created');
        require(userAccount[msg.sender] >= amount, 'Insufficient balance in
bank account');         require(amount > 0, 'Enter a non-zero value for
withdrawal');         userAccount[msg.sender] -= amount;
```

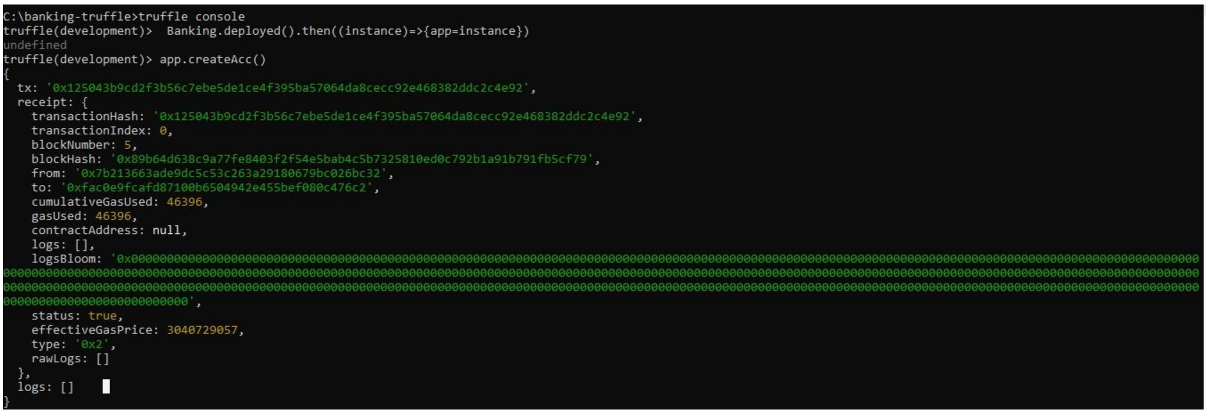
```
payable(msg.sender).transfer(amount);        return 'Withdrawal
successful';
    }
    function transferAmount(address payable userAddress, uint amount)
public returns (string memory) {
require(userExists[msg.sender], 'Account is not created');
        require(userAccount[msg.sender] >= amount, 'Insufficient balance in
bank account');
        require(userExists[userAddress], 'Recipient account does not exist in
bank accounts');        require(amount > 0, 'Enter a non-zero value
for sending');        userAccount[msg.sender] -= amount;
userAccount[userAddress] += amount;
        return 'Transfer successful';
    }
    function sendAmount(address payable toAddress, uint256 amount)
public returns (string memory) {
require(userExists[msg.sender], 'Account is not created');
        require(userAccount[msg.sender] >= amount, 'Insufficient balance in
bank account');        require(amount > 0, 'Enter a non-zero value for
withdrawal');        userAccount[msg.sender] -= amount;
toAddress.transfer(amount);
        return 'Transfer successful';
    }
    function userAccountBalance() public view returns (uint) {
return userAccount[msg.sender];
    }
```

```
    function accountExists() public view returns(bool) {
  return userExists[msg.sender];
    }
}
```

- **Output :**

```
C:\banking-truffle>truffle compile

Compiling your contracts...
===========================
> Compiling .\contracts\Banking.sol
> Artifacts written to C:\banking-truffle\build\contracts
> Compiled successfully using:
   - solc: 0.5.16+commit.9c3226ce.Emscripten.clang
```

```
C:\banking-truffle>truffle migrate

Compiling your contracts...
===========================
> Compiling .\contracts\Banking.sol
> Artifacts written to C:\banking-truffle\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)


2_deploy_contracts.js
=====================

   Deploying 'Banking'
   -------------------
   > transaction hash:    0x76c1cf4696c809aafa12dd4710b1bc416560b1c6a5014a9e3d29594621709899
   > Blocks: 0            Seconds: 0
   > contract address:    0xfac0E9Fcafd87100B6504942e455BEf080c476C2
   > block number:        4
   > block timestamp:     1702468079
   > account:             0x7B213663aDe9dC5c53C263A29180679bC026bC32
   > balance:             99.995978079364391096
   > gas used:            592014 (0x9088e)
   > gas price:           3.102807457 gwei
   > value sent:          0 ETH
   > total cost:          0.001836905453848398 ETH

   > Saving artifacts
   -------------------------------------
   > Total cost:     0.001836905453848398 ETH

Summary
=======
> Total deployments:   1
> Final cost:          0.001836905453848398 ETH
```



```
C:\banking-truffle>truffle console
truffle(development)>  Banking.deployed().then((instance)=>{app=instance})
undefined
truffle(development)> app.createAcc()
{
  tx: '0x125043b9cd2f3b56c7ebe5de1ce4f395ba57064da8cecc92e468382ddc2c4e92',
  receipt: {
    transactionHash: '0x125043b9cd2f3b56c7ebe5de1ce4f395ba57064da8cecc92e468382ddc2c4e92',
    transactionIndex: 0,
    blockNumber: 5,
    blockHash: '0x89b64d638c9a77fe8403f2f54e5bab4c5b7325810ed0c792b1a91b791fb5cf79',
    from: '0x7b213663ade9dc5c53c263a29180679bc026bc32',
    to: '0xfac0e9fcafd87100b6504942e455bef080c476c2',
    cumulativeGasUsed: 46396,
    gasUsed: 46396,
    contractAddress: null,
    logs: [],
    logsBloom: '0x000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000',
    status: true,
    effectiveGasPrice: 3040729057,
    type: '0x2',
    rawLogs: []
  },
  logs: []
}
```

```
C:\banking-truffle>truffle console
truffle(development)> Banking.deployed().then((instance)=>{app=instance})
undefined
truffle(development)> app.createAcc()
{
  tx: '0x68ed95f654caea268e1ad79ecff9bf1b46f39dd6531fb4df67b64348924a299d',
  receipt: {
    transactionHash: '0x68ed95f654caea268e1ad79ecff9bf1b46f39dd6531fb4df67b64348924a299d',
    transactionIndex: 0,
    blockNumber: 2,
    blockHash: '0xc6bd3f166698938ef230963fadd11781cd8ced5bde9d2dd21b7c45e26a609fbd',
    from: '0x1b2db4f14459b0106c4148b8acfd3b622050306e',
    to: '0x7b556b1c5c14ce035ff5e21fc762952fd86c7df5',
    cumulativeGasUsed: 46396,
    gasUsed: 46396,
    contractAddress: null,
    logs: [],
    logsBloom: '0x000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000',
    status: true,
    effectiveGasPrice: 3285613139,
    type: '0x2',
    rawLogs: []
  },
  logs: []
}
```

```
truffle(development)> app.deposit({value:500})
{
  tx: '0x0c2410b4e8cbf94fa910913fc596f271fbd015c3e81c145c193db3e99d1e7e3d',
  receipt: {
    transactionHash: '0x0c2410b4e8cbf94fa910913fc596f271fbd015c3e81c145c193db3e99d1e7e3d',
    transactionIndex: 0,
    blockNumber: 3,
    blockHash: '0xe9419bf8a26ae4647d7c51993951066b8d4606e7e2b294fad331518da49dd4a6',
    from: '0x1b2db4f14459b0106c4148b8acfd3b622050306e',
    to: '0x7b556b1c5c14ce035ff5e21fc762952fd86c7df5',
    cumulativeGasUsed: 46073,
    gasUsed: 46073,
    contractAddress: null,
    logs: [],
    logsBloom: '0x000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000',
    status: true,
    effectiveGasPrice: 3188767100,
    type: '0x2',
    rawLogs: []
  },
  logs: []
}
```

```
truffle(development)> app.withdraw(10)
{
  tx: '0xf514a40fca1b482e0771f50c3483829e483f1e102cdc249f1ad15b05869fa2eb',
  receipt: {
    transactionHash: '0xf514a40fca1b482e0771f50c3483829e483f1e102cdc249f1ad15b05869fa2eb',
    transactionIndex: 0,
    blockNumber: 4,
    blockHash: '0x7f55e4ef32ef0cbdab2b8bd6f8ca54393a03a7981da5a360f3ecb323c291c174',
    from: '0x1b2db4f14459b0106c4148b8acfd3b622050306e',
    to: '0x7b556b1c5c14ce035ff5e21fc762952fd86c7df5',
    cumulativeGasUsed: 36270,
    gasUsed: 36270,
    contractAddress: null,
    logs: [],
    logsBloom: '0x000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000',
    status: true,
    effectiveGasPrice: 3103851431,
    type: '0x2',
    rawLogs: []
  },
  logs: []
}
```

```
truffle(development)> app.userAccountBalance()
BN {
  negative: 0,
  words: [ 490, <1 empty item> ],
  length: 1,
  red: null
}
truffle(development)>
```