Practical No : 3

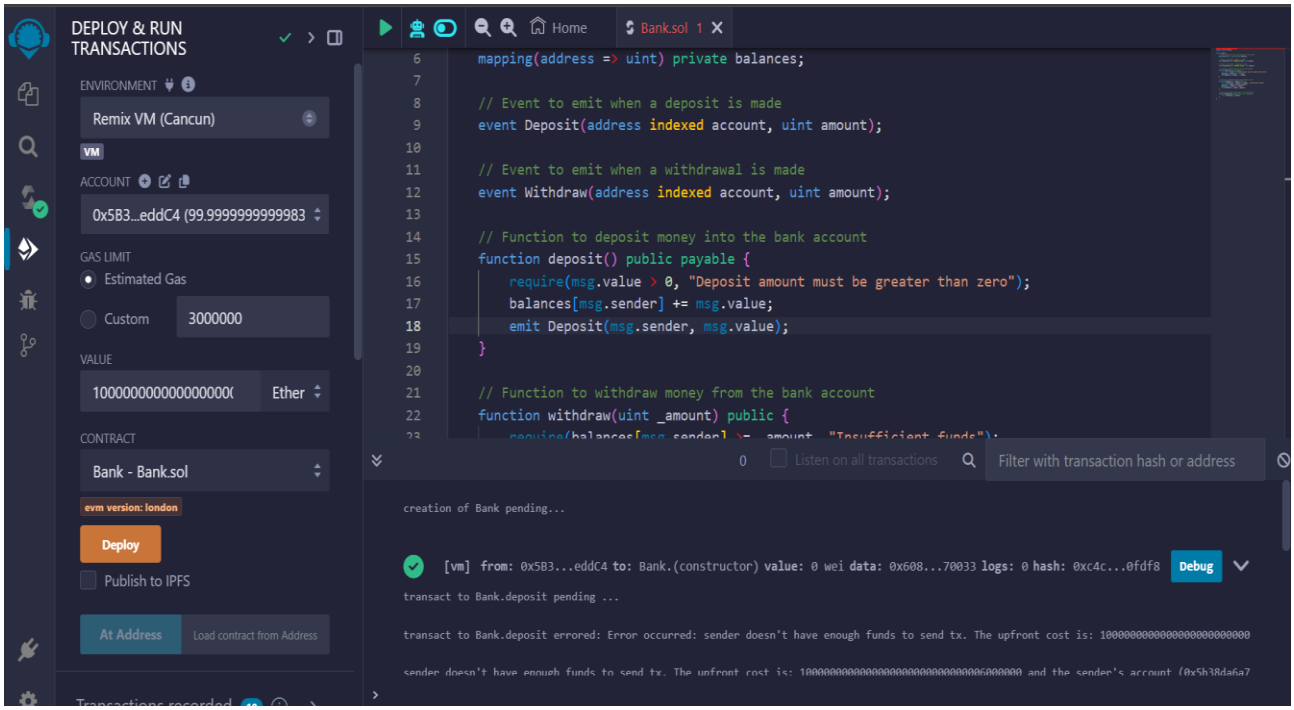
Write a smart contract on a test network, for Bank account of a customer for following operations:

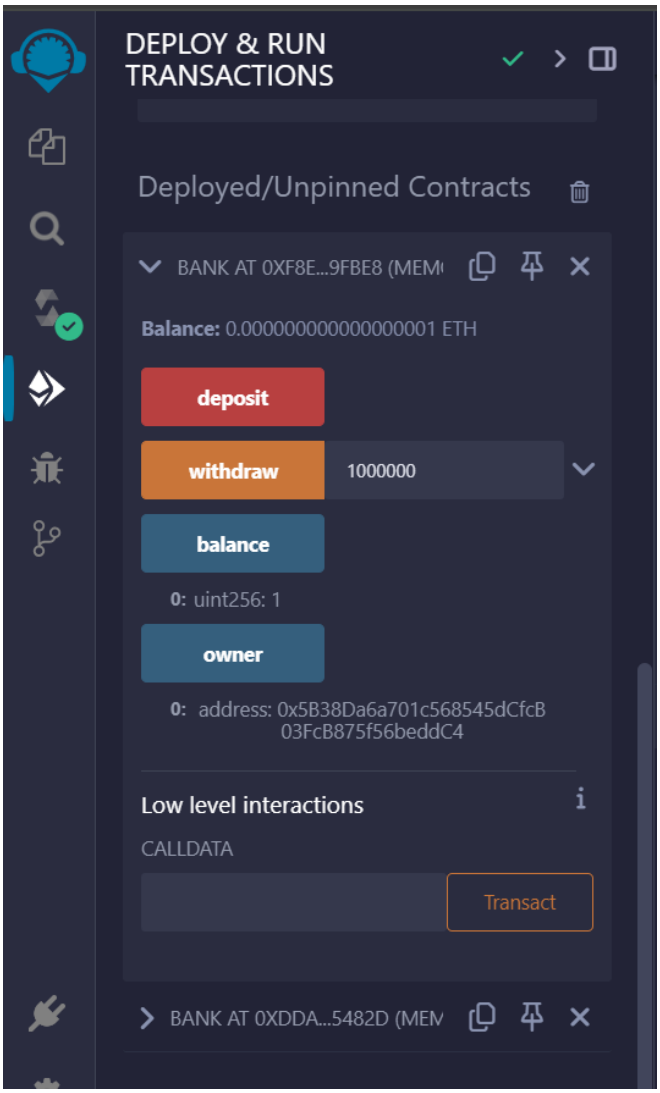
• Deposit money • Withdraw Money • Show balance

Code :

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract Bank {
    // State variable to store the balance of the account
    mapping(address => uint) private balances;

    // Event to emit when a deposit is made
    event Deposit(address indexed account, uint amount);

    // Event to emit when a withdrawal is made
    event Withdraw(address indexed account, uint amount);

    // Function to deposit money into the bank account
    function deposit() public payable {
        require(msg.value > 0, "Deposit amount must be greater than zero");
        balances[msg.sender] += msg.value;
        emit Deposit(msg.sender, msg.value);
    }

    // Function to withdraw money from the bank account
    function withdraw(uint _amount) public {
        require(balances[msg.sender] >= _amount, "Insufficient funds");
        payable(msg.sender).transfer(_amount);
        balances[msg.sender] -= _amount;
        emit Withdraw(msg.sender, _amount);
    }

    // Function to check the balance of the bank account
    function getBalance() public view returns (uint) {
        return balances[msg.sender];
    }
}
```

Output :


DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Remix VM (Cancun)
VM
ACCOUNT
0x5B3...eddC4 (99.9999999999983
GAS LIMIT
Estimated Gas
Custom
3000000
VALUE
100000000000000000(
Ether
CONTRACT
Bank - Bank.sol
evm version: london
Deploy
Publish to IPFS
At Address
Load contract from Address
Home
Bank.sol 1
mapping(address => uint) private balances;
// Event to emit when a deposit is made
event Deposit(address indexed account, uint amount);
// Event to emit when a withdrawal is made
event Withdraw(address indexed account, uint amount);
// Function to deposit money into the bank account
function deposit() public payable {
require(msg.value > 0, "Deposit amount must be greater than zero");
balances[msg.sender] += msg.value;
emit Deposit(msg.sender, msg.value);
}
// Function to withdraw money from the bank account
function withdraw(uint _amount) public {
0
Listen on all transactions
Filter with transaction hash or address
creation of Bank pending...
[vm] from: 0x5B3...eddC4 to: Bank.(constructor) value: 0 wei data: 0x608...70033 logs: 0 hash: 0xc4c...0fdf8
Debug
transact to Bank.deposit pending ...
transact to Bank.deposit errored: Error occurred: sender doesn't have enough funds to send tx. The upfront cost is: 100000000000000000000


DEPLOY & RUN TRANSACTIONS
Deployed/Unpinned Contracts
BANK AT 0XF8E...9FBE8 (MEM
Balance: 0.000000000000000001 ETH
deposit
withdraw
1000000
balance
0: uint256: 1
owner
0: address: 0x5B38Da6a701c568545dCfcB
03FcB875f56beddC4
Low level interactions
CALLDATA
Transact
BANK AT 0XDDA...5482D (MEM