Program: -

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;
contract banking{
  mapping(address=>uint) public userAccount;
  mapping(address=>bool) public userExists;

  function createAcc() public payable returns(string memory){
     require(userExists[msg.sender]==false,'Account Already
Created');
     if(msg.value==0){
        userAccount[msg.sender]=0;
        userExists[msg.sender]=true;
        return 'account created';
     }
     require(userExists[msg.sender]==false,'account already created');
     userAccount[msg.sender] = msg.value;
     userExists[msg.sender] = true;
     return 'account created';
  }

  function deposit() public payable returns(string memory){
     require(userExists[msg.sender]==true, 'Account is not created');
     require(msg.value>0, 'Value for deposit is Zero');
     userAccount[msg.sender]=userAccount[msg.sender]+msg.value;
     return 'Deposited Succesfully';
  }

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

    address payable senderAddress = payable(msg.sender);
    senderAddress.transfer(amount);

    userAccount[msg.sender] -= amount;

    return 'Withdrawal Successful';
}
```

```
  function TransferAmount(address payable userAddress, uint
amount) public returns(string memory){
      require(userAccount[msg.sender]>amount, 'insufficeint balance
in Bank account');
      require(userExists[msg.sender]==true, 'Account is not created');
      require(userExists[userAddress]==true, 'to Transfer account does
not exists in bank accounts ');
      require(amount>0, 'Enter non-zero value for sending');
      userAccount[msg.sender]=userAccount[msg.sender]-amount;
      userAccount[userAddress]=userAccount[userAddress]+amount;
      return 'transfer succesfully';
  }

  function sendAmount(address payable toAddress , uint256 amount)
public payable returns(string memory){
      require(amount>0, 'Enter non-zero value for withdrawal');
      require(userExists[msg.sender]==true, 'Account is not created');
      require(userAccount[msg.sender]>amount, 'insufficeint balance
in Bank account');
      userAccount[msg.sender]=userAccount[msg.sender]-amount;
      toAddress.transfer(amount);
      return 'transfer success';
  }

  function userAccountBalance() public view returns(uint){
      return userAccount[msg.sender];
  }

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

}
```

Output:

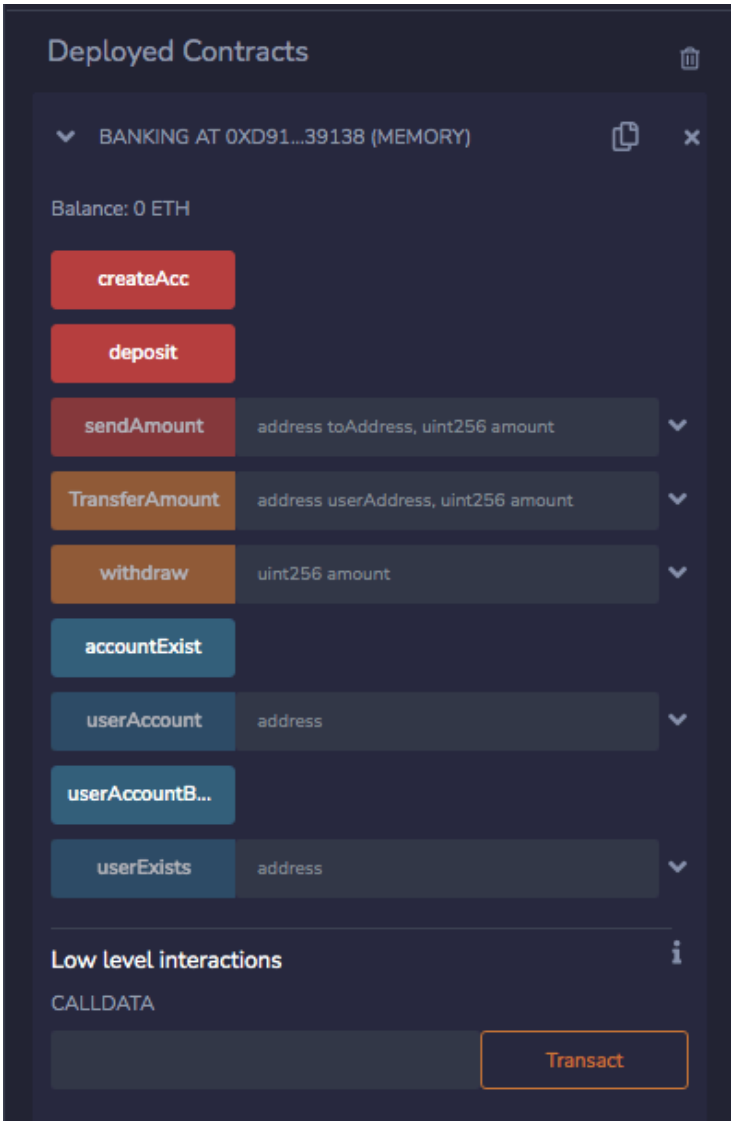

Fig: 1 Deployed instances

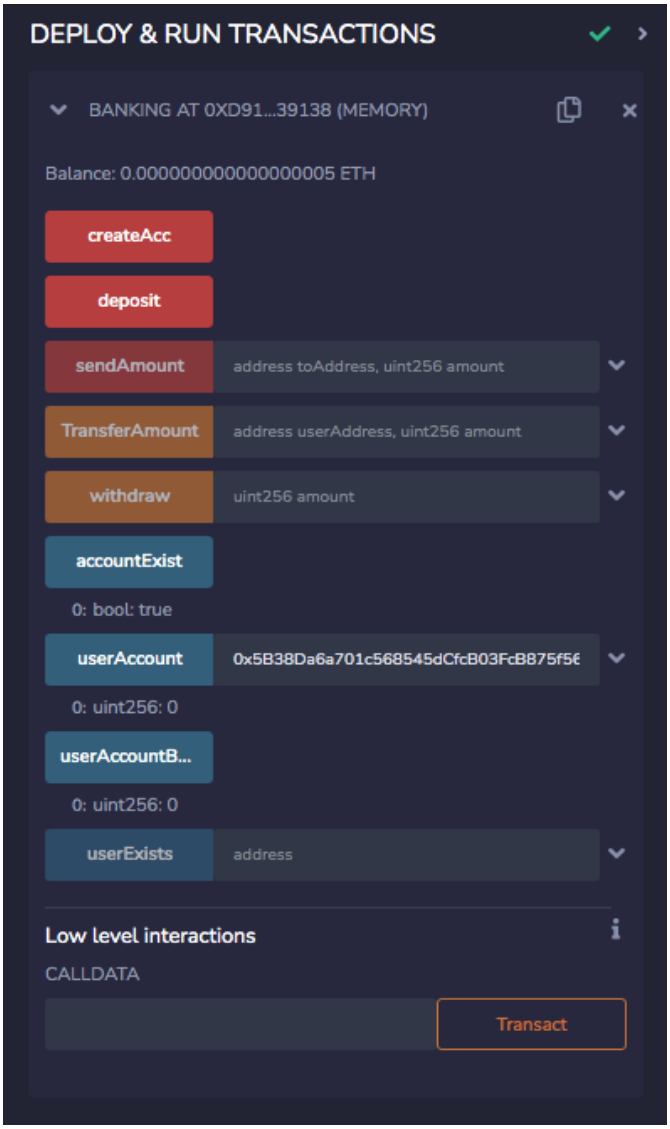

Fig:2 Balance of account 1 using userAccountBalance instance

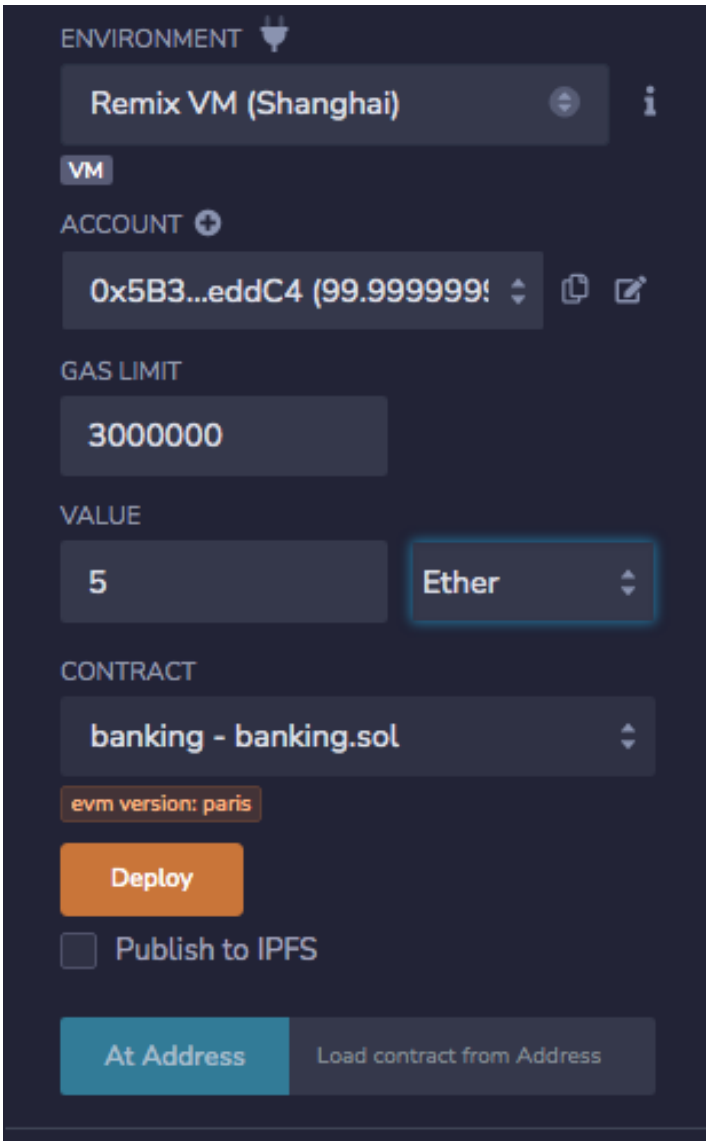

Fig:3 Deposit Money to Account 1

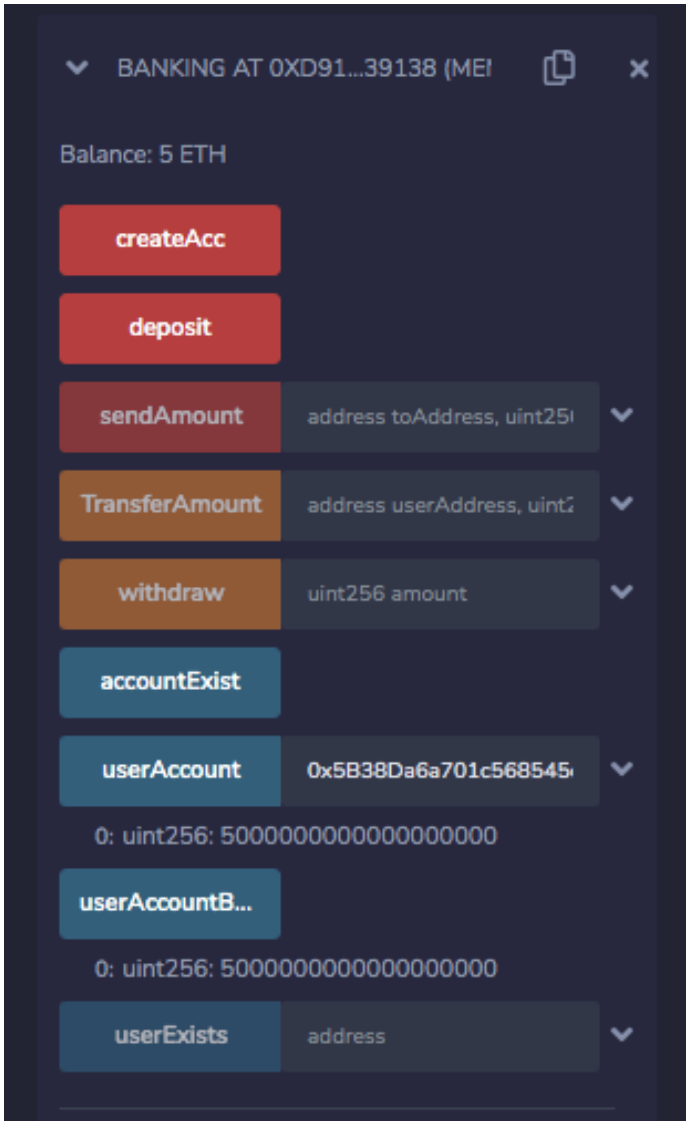

Fig:4 View Balance of Account 1

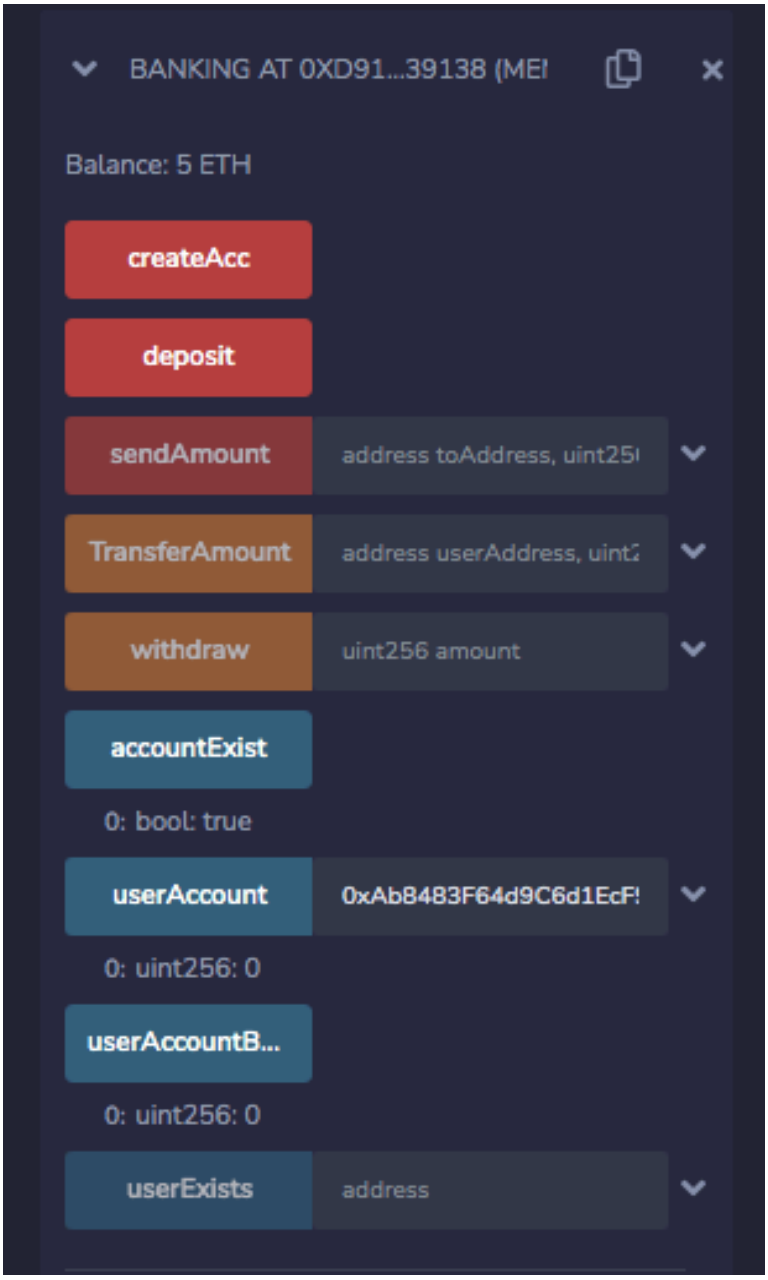

Fig:5 Create Account 2

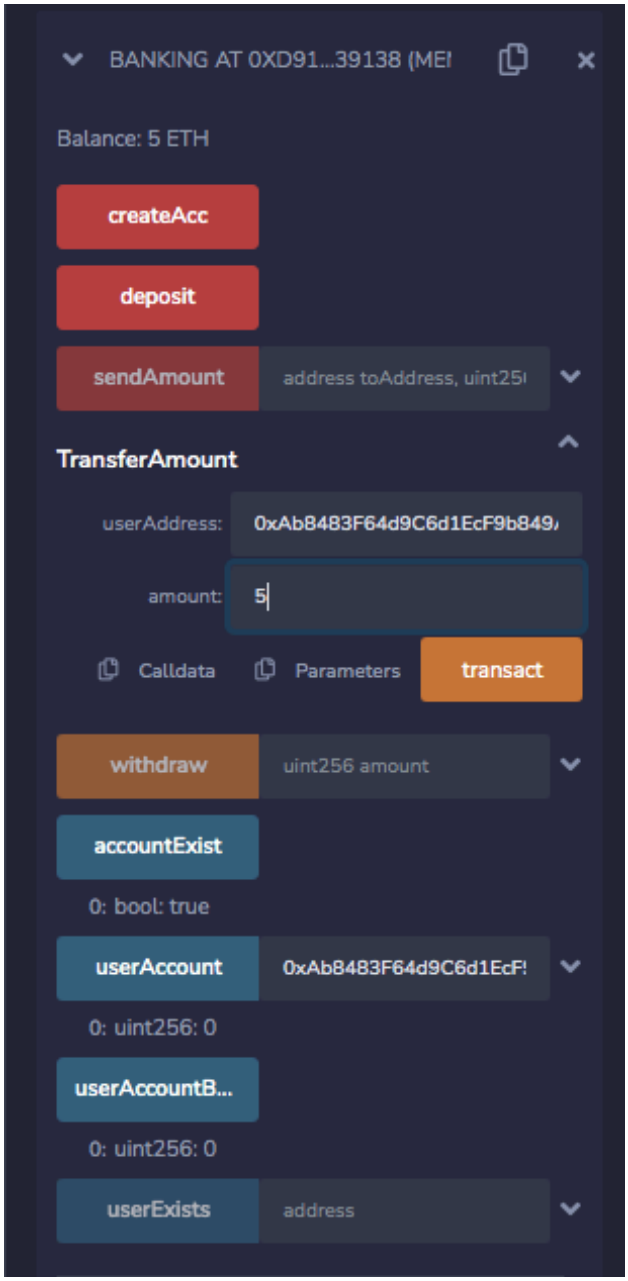

Fig:6 Transfer Money from Account 1 to Account 2

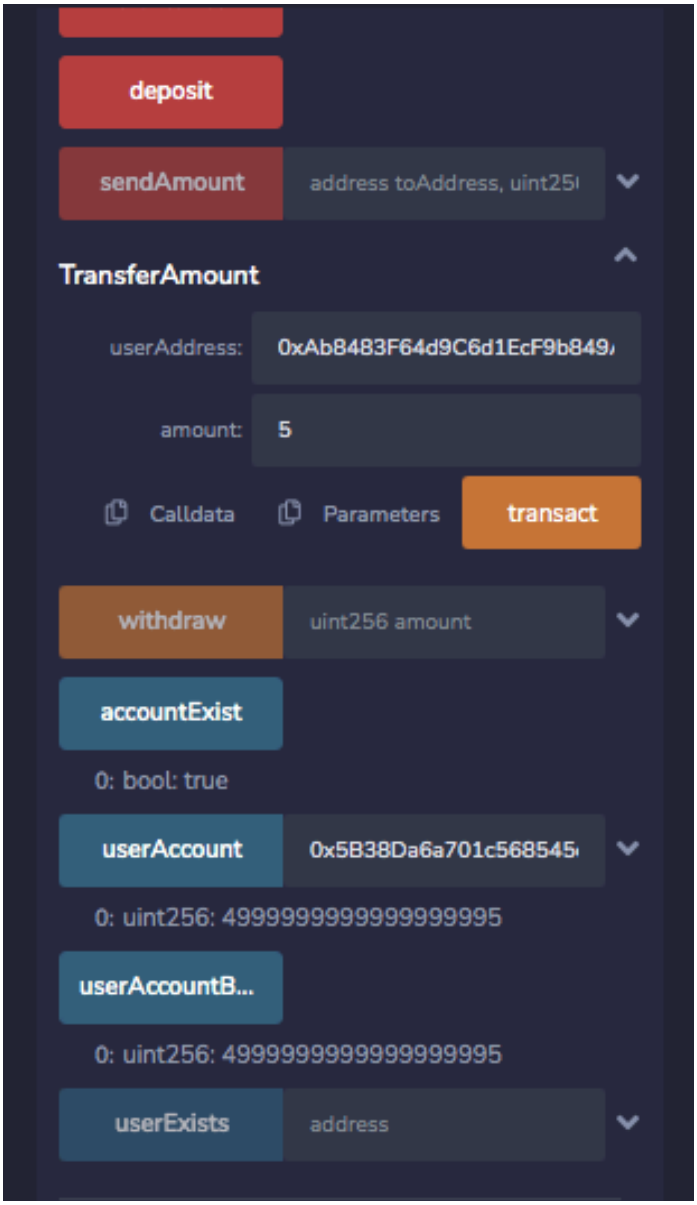

Fig:7 Balance after the transaction in Account 1

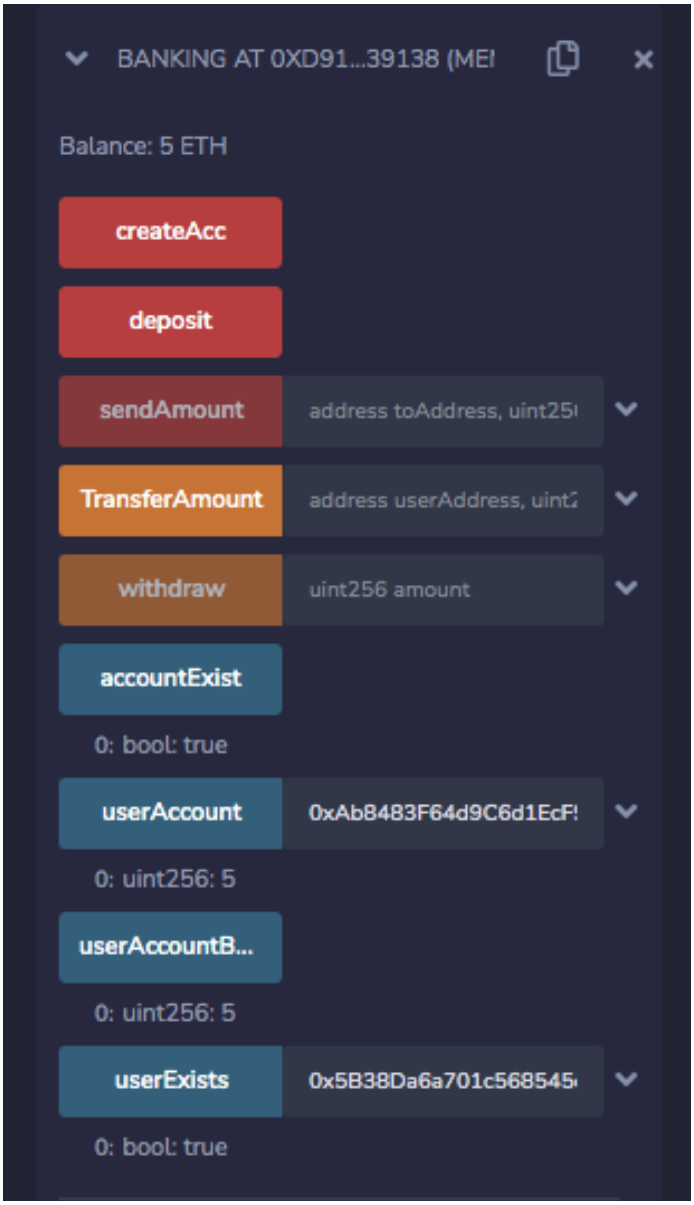

Fig:8 Balance after the transaction in Account 2

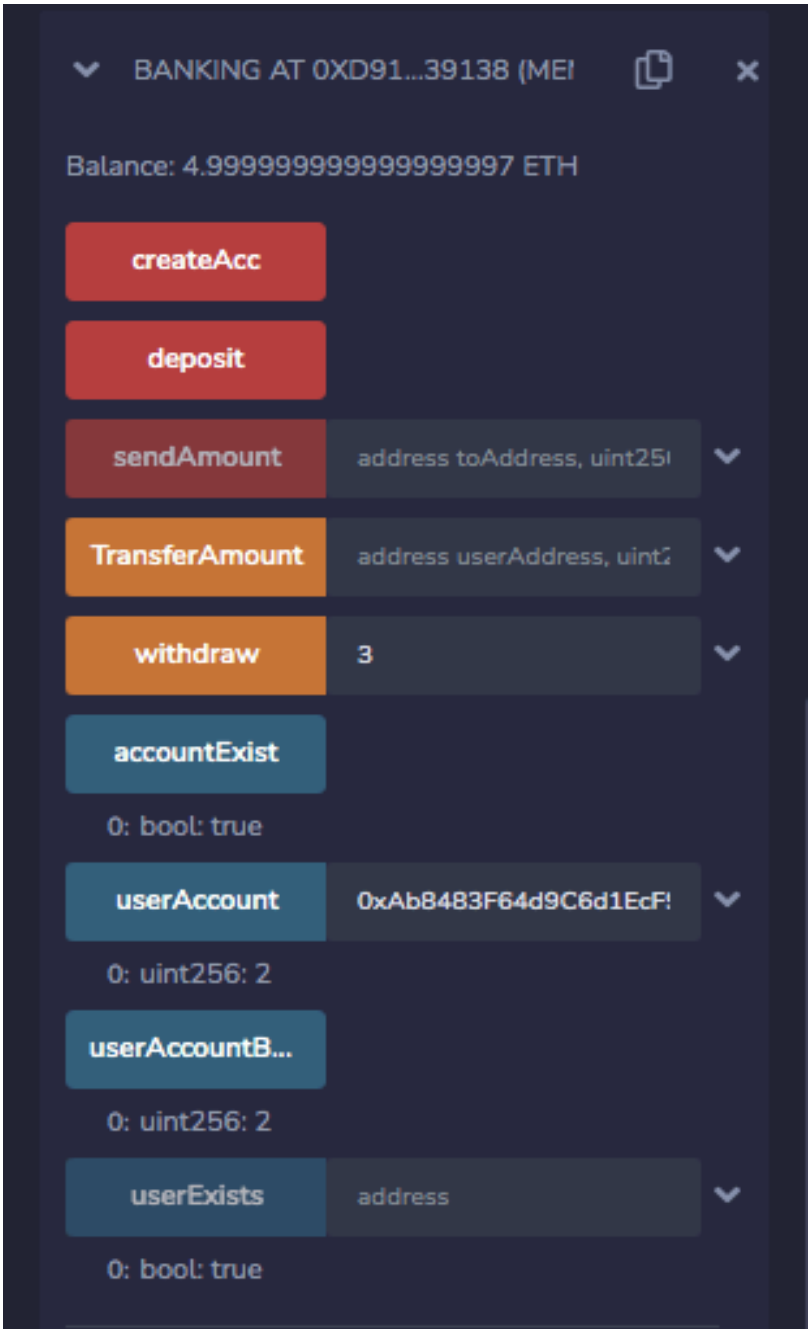

Fig:9 Withdraw Money from Account 2