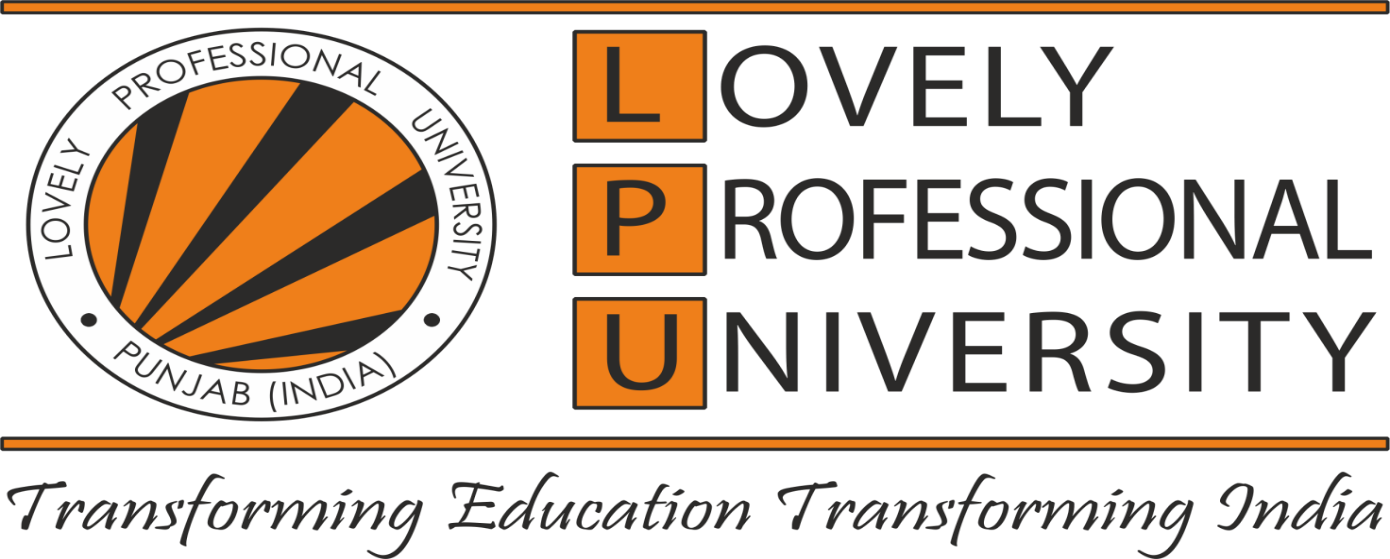
**Academic Task CA-03**

**CSC307**

****

**Submitted To: Piyush Gururani Sir**

**Submitted By: Aditya Sudheer Nambiar**

**Registration Number: 12214396**

**Section: K22CS**

**Roll Number: 07**

**Problem statement**

Q 2] The following contract tracks ownership of items in a supply chain. Identify and explain the security vulnerabilities in this contract. Rewrite the vulnerable function(s) to enhance its security, particularly against reentrancy and unauthorized access.

pragma solidity ^0.8.0;

contract SupplyChain {

struct Item {

uint id;

address owner;

}

mapping(uint => Item) public items;

function transferItem(uint itemId, address newOwner) public {

require(items[itemId].owner == msg.sender, "Not the owner");

items[itemId].owner = newOwner;

(bool sent, ) = newOwner.call{value: msg.value}("");

require(sent, "Transfer failed");

}

}

**Vulnerabilities:**

1. Reentrancy Vulnerability:

The transferItem function interacts with the external address newOwner using call before completing the state update (items[itemId].owner = newOwner). This opens the contract to reentrancy attacks where a malicious newOwner could exploit the incomplete state update to reenter the function and manipulate logic or steal assets.

1. Lack of Access Control for Transfers:

While the function ensures only the current owner can initiate a transfer, it does not have robust access control mechanisms. For instance, there is no mechanism to prevent unauthorized transfers of items if the ownership mapping is compromised.

**Enhanced code to prevent vulnerabilities:**

// SPDX-License-Identifier: UNLICENSED

pragma solidity ^0.8.0;

contract SupplyChain {

struct Item {

uint id;

address owner; }

mapping(uint => Item) public items;

modifier onlyOwner(uint itemId) {

require(items[itemId].owner == msg.sender, "Not the owner");

\_; }

function transferItem(uint itemId, address newOwner) public onlyOwner(itemId) {

require(newOwner != address(0), "Invalid new owner address");

address previousOwner = items[itemId].owner;

items[itemId].owner = newOwner;

}

}