***CA-3***

CSC307 – Blockchain Architecture and Design

Submitted By:

Chirag Goyal

12205559

Submitted To:

Mr. Piyush Gururani

**Problem Statement**

4. Create a Solidity contract that tracks goods as they move through different stages in a supply chain. Define a function to allow an authorized entity to update the status of a shipment (e.g., “In Transit,” “Delivered”). Ensure that only authorized entities can update the status, and users can view the status. Include code for role-based access control.

**Solution:**

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

contract SupplyChain {

// Address of the admin

address public admin;

// Mapping for updater roles

mapping(address => bool) public isUpdater;

// Struct to hold shipment details

struct Shipment {

string description;

string status;

address lastUpdatedBy;

}

// Mapping to store shipments by ID

mapping(uint256 => Shipment) public shipments;

// Event emitted when a shipment's status is updated

event ShipmentStatusUpdated(

uint256 indexed shipmentId,

string status,

address indexed updatedBy

);

// Event emitted when an updater is added or removed

event UpdaterRoleChanged(address indexed updater, bool isAuthorized);

// Modifier to restrict access to admin

modifier onlyAdmin() {

require(msg.sender == admin, "Not an admin");

\_; }

// Modifier to restrict access to updaters

modifier onlyUpdater() {

require(isUpdater[msg.sender], "Not an updater");

\_;

}

// Constructor to set the deployer as the admin

constructor() {

admin = msg.sender;

}

// Function to add an updater (only callable by admin)

function addUpdater(address updater) external onlyAdmin {

isUpdater[updater] = true;

emit UpdaterRoleChanged(updater, true);

}

// Function to remove an updater (only callable by admin)

function removeUpdater(address updater) external onlyAdmin {

isUpdater[updater] = false;

emit UpdaterRoleChanged(updater, false);

}

// Function to update a shipment's status (only callable by updaters)

function updateShipmentStatus{ uint256 shipmentId,

string description,

string status

)public onlyUpdater {

shipments[shipmentId] = Shipment({

description: description,

status: status,

lastUpdatedBy: msg.sender

});

emit ShipmentStatusUpdated(shipmentId, status, msg.sender);

}

// Function to view shipment details

function getShipment(uint256 shipmentId) public view returns (

string memory description,

string memory status,

address lastUpdatedBy

) {

Shipment memory shipment = shipments[shipmentId];

return (shipment.description, shipment.status, shipment.lastUpdatedBy);

}

}

**Key Features**

1. **Custom Role-Based Access Control**:
   * **Admin**: The deployer is the admin by default.
   * **Updater**: Admin can grant/revoke updater roles by managing isUpdater.
2. **Shipment Tracking**:
   * Each shipment is stored in the shipments mapping.
   * Only authorized updaters can modify a shipment's status.
3. **Events**:
   * ShipmentStatusUpdated: Logs status changes.
   * UpdaterRoleChanged: Tracks changes to updater roles.
4. **Utility Functions**:
   * addUpdater and removeUpdater: Used by the admin to manage updaters.
   * getShipment: Provides shipment details.

**Deployment and Usage**

1. Deploy the contract; the deployer will be set as the admin.
2. Use addUpdater to authorize addresses as updaters.
3. Use removeUpdater to revoke updater permissions.
4. Updaters can use updateShipmentStatus to update the shipment's details.
5. Anyone can call getShipment to view shipment details.