**Class activities on ATOMIC**

**Name : LAW IVAN**

**Matric No : 224615**

Question:

In a distributed system, the concept of atomicity ensures that a group of operations either all occur or none occur, especially during communication between processes.

Scenario:

Consider a distributed banking system where a fund transfer involves two operations:

1. Deducting an amount from Account A (at Node X),

2. Adding the same amount to Account B (at Node Y).

---

1. Explain how atomicity can be maintained in this transmission.
    - In a distributed banking system, atomicity ensures that deduction from Account A and addition to Account B must both succeed or both fail. It can only be logically one operation split across two physical nodes. WIthout atomicity, inconsistent state and financial errors will occurs.

2. What problems could arise if atomicity is not ensured?
    - Partial Transfer - Amount deducted from Account A but Account B did not receive the money, which is called money lost. Money created will also occur when an amount is added to Account B but is not deducted from Account A.
    - Inconsistency - Different nodes have conflicting views of te transaction state.
    - Duplicate Processing - Transaction may be re-processed incorrect if the rollback is not coordinated properly.

3. Briefly describe how a Two-Phase Commit (2PC) protocol can help achieve atomic transmission in such a scenario.

- **Phase 1: Prepare Phase**
    1. Coordinator sends a <PREPARE> request to all participants (Node X and Node Y).
    2. Each participant checks if it can commit:
        - Node X checks if it can deduct the amount.
        - Node Y checks if it can add the amount.
    3. If a participant wants to proceed and has validated local constraints like enough funds at Node X, it will send a message <READY> to the coordinator.
    4. Each node replies with:
        - Vote-Commit: ready to proceed.
        - Vote-Abort: cannot proceed.

- **Phase 2: Commit Phase**
    1. If all participants vote to commit, the coordinator sends a <COMMIT> message.
    2. If any participant votes to abort, the coordinator sends an <ABORT> message.
    3. All participants then commit or roll back accordingly.