**Distributed Systems Lab Assignment -2**

**Lab Coding Exercise: Implementing Lamport Logical Clocks in a Distributed System**

Create a simulation involving multiple processes that perform internal events and send/receive messages. Each process will maintain a Lamport clock to track logical time. Demonstrate how these clocks are updated according to the rules of Lamport's logical clock algorithm.

Step 1. Set Up the Process Structure

1. Define a Process class that represents an individual process in a distributed system.

Each process should have: unique process\_id, An integer clock initialized to 0, representing the logical clock,

1. Implement methods:
   * + increment\_clock(): Increments the process's logical clock by 1.
     + internal\_event(): Simulates an internal event by incrementing the clock and printing the event details.
     + send\_message(receiver): Increments the clock, sends a message to another process, and prints the message details.
     + receive\_message(sender\_clock): Receives a message from another process, updates the clock, and prints the event details.

Step 2. Implement the Lamport Logical Clock Algorithm

Implement the Lamport clock algorithm within the Process class:

Step 3. Simulate a Distributed System

Step 4. Analyze and explain the ordering:

For Submission:

1. **Implement Vector Clocks**: Extend your implementation to use vector clocks instead of Lamport clocks. Compare the differences in how events are ordered.
2. **Concurrency Detection:** Modify your simulation to detect and report when two events are concurrent.

* Submit your Python/C++ code and the sample output screenshot in a single pdf document. Include a brief report analyzing the event ordering.
* For 1 and 2 ppt slides for understanding the algorithms are being uploaded on the study material **(SM)** folder.
* Your pdf file name should be your complete roll no.