**Distributed Systems Lab 5.**

**Indirect Communication using Java.**

**Topics:** Message Passing, Publish-Subscribe, and Group Communication

[You must submit code + screenshot of the output of at least THREE of the following assignments.

Submit your solutions to [sc.mlbda@gmail.com](mailto:sc.mlbda@gmail.com) with subject “**Lab 5 Submission**” and name your file as your enrollment number.

Last date of submission is September 28, 2024]

**1. Message Passing using Message Queues**

Implement a distributed system where multiple clients send messages to a server using a message queue (e.g., RabbitMQ or ActiveMQ). The server processes the messages and sends back a response to the clients.

**2. Publish-Subscribe Model using JMS (Java Message Service)**

Implement a Publish-Subscribe system where publishers send messages to a topic, and subscribers receive messages from the topic. The system should support multiple subscribers for a single topic.

**3. Group Communication using Multicast Sockets**

Implement a distributed system where multiple clients communicate with each other using multicast sockets. Each client should be able to send messages to a group, and all group members should receive the messages.

**5. Load Balancing using Indirect Communication**

- Problem: Implement a distributed system where clients send tasks to a load balancer, which distributes the tasks among multiple workers using message queues.

**6. Chat Application using Publish-Subscribe**

Implement a distributed chat application where users can join different chat rooms. Messages sent in a chat room should be broadcasted to all users in the room using the Publish-Subscribe model.

**7. Notification System using Indirect Communication**

Implement a distributed notification system where a notification service publishes events to various subscribers (e.g., email, SMS, mobile app) based on the type of notification.

**8. Reliable Communication using Acknowledgments**

Implement a messaging system where clients send messages to a server, and the server acknowledges the receipt of each message. Ensure that no message is lost in the communication.