GHANA COMMUNICATION TECHNOLOGY UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE
FIRST SEMESTER 2025 ACADEMIC YEAR
BSc. COMPUTER SCIENCE - LEVEL 300
End-of-Semester Project
CSBC 311 - Distributed Systems

Preamble

You are the lead distributed systems specialist in your company. Management has noticed that teams working across different regions struggle to coordinate because the current internal messaging tools are unreliable and not built to handle distributed environments. You have been tasked by Management to design and build a communication system that supports both group messaging and one-to-one private messaging between members in a distributed setup. The goal is to create a simple but functional distributed chat application system that reflects real-world distributed communication concepts.

Project Title

Design and Implementation of a Distributed Communication system capable of supporting group and Private messaging.

Project Description

Students will build a functional distributed communication application where multiple users can connect across a network. Users must be able to join groups, send messages to the entire group (multicast), and send private messages to specific online members (unicast). The system must demonstrate key concepts in distributed systems, such as explicit message delivery, active membership management, and fundamental communication reliability.

Core Requirements

1. Group Management

- Users should be able to create a group.
- Users should be able to join or leave a group.
- The system should keep track of active members.

2. Group Communication

- A member can send a message to the entire group.

- All online members of the group should receive the message.

3. Private Messaging

- A member should be able to choose a specific user in the group and send a direct private message.
- Only the intended recipient should receive the private message.

4. Distributed Operation

- The system should run across multiple nodes or machines (or simulated nodes).
- Communication should not rely on a shared memory model; it must use message passing (e.g., sockets, WebSockets, gRPC, or message queues).

5. Reliability Features (Basic Level)

Implement at least one of the following:

- Detect when a member joins or leaves.
- Retry message delivery if a recipient is temporarily unreachable.
- Keep a simple log of messages.

6. Interface (Flexible)

You may choose any of the following:

- Graphical user interface
    - Simple web interface
    - Desktop application
- Command-line interface

Optional Enhancements (Extra Credit)

- Message ordering within a group
- Acknowledgments for delivered messages
- Persistent chat history
- End-to-end encryption for private messages
- Fault detection (e.g., heartbeat checks)

Deliverables

1. **Source code** with brief setup instructions
2. **Short report** (2–3 pages) explaining:
    - Approach
    - System architecture diagram
    - Technologies used
    - Distributed concepts demonstrated
    - Limitations and future improvements
3. **presentation and submission date**

    7th January, 2026