Lebanese American University
Department of Computer Science and Mathematics
CSC 430 - Computer Networks
Summer 2025

Design Project:
Multithreaded Chat Application

1. Project Overview

This project builds a real-time chat app using Python and sockets with a client-server design. The server handles multiple clients simultaneously, allowing users to chat publicly or privately. Additional features include a GUI or web interface, message logging, encryption, authentication, offline messaging, and file sharing.

2. Core Features

A. Client-Server Architecture

- A central server manages all connections and communications.
- Multiple clients can connect concurrently using unique nicknames.
- The server uses multithreading to manage multiple clients in parallel.

B. Real-Time Messaging

- **Public Chat**: Messages are broadcast to all users in a shared chatroom.
- **Private Chat**: Users can send direct messages to specific individuals.

C. Chatroom Management

- Clients can create or join chatrooms using commands.
- Users can switch between chatrooms.
- Notifications are sent when users join or leave a chatroom.

D. Command-Based Communication

- Simple command syntax for client actions:
  - JOIN room_name: Join or create a chatroom
  - MSG user:message: Send a private message
  - MSG message: Send a public message
  - LEAVE: Leave the current chatroom
  - LIST: Display all active users and rooms

E. Nickname System

- Every client must register with a unique nickname to avoid confusion.

3. Optional / Advanced Features

- Develop a **graphical or web-based user interface** for better usability (e.g., Tkinter or Flask/Django + JS).
- Implement **user authentication** with support for roles such as regular users and administrators.
- Add **end-to-end encryption** to secure message transmission and protect user privacy.
- Support **offline messaging**, where messages are stored and delivered when the user reconnects.
- Enable **file sharing** within chatrooms or private messages (with file size and type restrictions).
- Provide **admin tools** to monitor activity, broadcast announcements, or remove disruptive users.
- Show **real-time user presence** (online/offline) and **typing indicators** to enhance interactivity.

4. Grading

You will work in groups of two. Your grade will be based on three components: the quality of your code, the clarity and completeness of your report, and the effectiveness of your live demo.

5. Deliverables

At the end of this project, you should submit your (thoroughly documented) code along with a project report. In this report, you should describe your high-level approach, the challenges you faced, a list of properties/features of your design, and an overview of how you tested your code. Your report should explain each of the required functions. You should also include screen shots of your running application.

- The submission of the Final Report with the full application and a readme file is due on **the beginning of the last week of the semester on Blackboard**.
- Project Demos will be held in the **Last week of the semester**.

Important Notes:

- Each member of the group will be graded **individually** based on her/his input to the project.
- Code comments MUST clearly indicate which team member contributed which part of the code. Comments should also clarify the code.
- Code used from other sources (if any) must be clearly documented and flagged in the code by a comment.