CS 454: Software Engineering for Distributed Systems

Multi-Client and Multithreaded Chatting Application

Objective

The aim of this assignment is to create a simple multithreaded chatting application that utilizes: (i) a client-server architecture, (ii) socket programming, and (iii) multithreading. The chatting application should utilize a GUI for the involved clients, through which any information, requested from the client, would be entered. The chatting application should provide the following list of features:

- **Supported Clients:** The application should support four clients concurrently **at least**.
- **Logging into the application:** To use the chatting application, existing users would need to login, while new users need to register first. To login, the user needs to send his username and password. To register, the user needs to send his name, username, and password. Invalid login/registration scenarios should be handled as follows:
    - if password is wrong, 401 error should appear
    - if username is not found, 404 error should appear (not found)
    - if a username cannot be used to register a new user because it is already reserved, some custom error should appear.
- **Contacts list:** Once logged in, the user can see his own list of contacts in his GUI client. Contacts may be added by a user, provided that such contact person is notified and approves that addition in advance.
- **Handling concurrent messages:** Each client should be able to handle concurrent messages from multiple origins.
- **One-to-one chat:** A user can have a private chat with a client from his contact list only.
- **One-to-several:** A user can have a multicast chat with several clients from his contact list.
- **Join existing chat:** A user can join an ongoing one-to-one or one-to-many chat
- **Blocking inappropriate content:** All the content written by all clients should be monitored by the server. If an inappropriate word is written, the client who wrote that word should be kicked out from his current chat, and a counter of his warnings should be kept on the server.

P.S. You must handle any exceptions that may occur.
Team size: The assignment should be done in teams of size 2 at most.
Submission: You should submit all the needed source code to properly run the application, as one zip file. Submission should be done solely through Blackboard. Email submissions will be ignored.
Deadline: Monday, November 30th at 11:59 pm.