**Brainstorm**

* There is a server that receives messages from users and sends them to the correct users
* A user can log into a client GUI application
* A user logged into the client application can send a private message to another user on the system
* A user logged into the client can create and add other users to a group chat
* A user can send messages in a group chat
* A user can scroll through a list of users or search by name so they can message them or invite them to a group
* If a user logs out, when they log back in any messages sent to them offline will be available for to read
* If I send a message to a logged out user, it will be held until the user logs in
* A log is created for each conversation and updated when new messages are sent
* An IT user can access the logs of all conversations
* An IT use can add or remove users and change their passwords

**Scope**

The goal of this system is to provide organizations with an application that allows members to communicate in real time via text messages over TCP. This system will provide both direct and group messaging services. The system will require users to authenticate with a username and password before being able to send, receive, and read messages. To provide system administrators with the means to audit their organization’s communications, the system will provide logging features that will permit IT users with the special privilege of viewing all conversations on the system.

Users will interact with the system via a graphical user interface which will allow them to log in, view conversations, and send messages. They will be able to view messages sent to them while they were offline so that no communications get lost when sent outside of working hours.

**\*\*\* Requirements in italics may represent optional requirements that may need to be cut for deadlines \*\*\***

**Functional Requirements**

1. Client Application
   1. A user can connect to the service via a client application.
   2. The client application provides a graphical user interface for the user to interact with
   3. A user can log in to the instant messenger service by typing their username and password into text fields on the GUI and clicking a button labeled “Log In”
      1. If the server responds that the authentication was successful, they will be connected to the server and taken to their home page.
      2. If the server responds that authentication failed, they will be prompted to sign in again.
   4. A user can click a button labeled “Log Out” to disconnect from the system.
   5. Authenticated users will see a home page which lists all conversations of which they are a participant.
      1. A user can click on one of the conversations in their conversations list to view the conversation.
      2. *A user can scroll through a conversation to see all messages sent.*
      3. A user can type and send a new message to all conversation participants.
      4. When a new message is available in the conversation, the GUI of all connected users will update to display the new message.
      5. Users who are not currently active will be able to view messages sent while they were signed off upon logging in and opening the conversation.
      6. Conversations may be either direct conversations (two participants) or group conversations (many participants).
   6. In addition to their own private messages, authenticated IT users will be able to view the all conversations on the system.
2. Server Application
   1. The server will accept multiple concurrent users without blocking other users from using the service.
   2. When a user tries to connect to the server, the server will request the user to authenticate.
      1. The server will use username and password for authentication.
      2. The server will store the passwords of all users for authentication.
      3. The server will maintain a list of all registered users in the system.
      4. Each user will be classified as either a regular user or an IT user.
   3. The server will maintain a list of currently signed in users.
   4. If a user’s connection to the server breaks, the server will sign the user out of the system and require authentication before they can access private information.
   5. The server will maintain logs of all conversations.
   6. When a user sends a new message, the server will make the message available to all participants of the conversation.
   7. The server will provide IT users with access to all conversations in the system.

**Non Functional Requirements**

1. Both the client and server applications must be written in Java
2. The application should run on JDK 21
3. All communication must occur via TCP/IP
4. The number of simultaneous users supported should only be limited by the hardware running the server.
5. A full design should be completed by March 26th, 2024.
6. The final implementation of the application should be completed by April 30th, 2024.
7. Unit testing should be implemented using the JUnit test.

**Definitions**

**Conversation-** A log of messages between two or more participants

**Participant-** Someone who can send and receive messages as part of a conversation

**Use Cases**

1. As a user, I can log in to the messaging system through a GUI client application by providing my username and password.
2. If the username and password is not correct, the system will tell me “Invalid username/password” and prompt me to log back in.
3. Once I’ve logged in, I can see all the conversations I’m participating in.
4. I can find other users on the system to send messages to.
5. I can create a group message with multiple participants.
6. I can send messages directly to another user.
7. I can send messages to a group of users.
8. I can receive messages for any conversation I am a part of
9. I can log out of the system.
10. When I log in, I can see all the messages in all the conversations I am a participant in, including messages sent while I was logged out.
11. As an IT user, I can see all the conversations on the system, but does not count as a participant towards that conversation.
12. Each conversation is logged to a file on the server.
13. As a user, if my network connection breaks, I am logged out of the server and prompted to sign back in.
14. If I try to sign in without an internet connection, the client application will tell me “No network connection.”

**Class Candidates**

* Client
  + Many clients to one server
  + GUI
  + Socket to connect to the server
  + Log in user
  + Send message
  + Receive message
* Server
  + One server
  + List of users
  + List of connected clients
  + List of conversations
  + Forward message to the appropriate conversation
* User
  + Username : String
  + Password : String
  + List of conversations
  + Send message
  + Receive message
  + Log in
  + Log out
  + Create conversation
* ITUser : User
  + Subclass of user
  + Snoop on a conversation
* Conversation
  + List of participants
  + File containing a log of all messages
  + Conversation’s responsibility to log each new message as it comes in.

**Assignments**

Caleb will do the use case diagrams

Francis will do the sequence diagrams

Quang will do the UML

Marian will do the Gantt chart

Ashish can do the slides for the presentation.

Everyone will edit, clean up, and finalize the SRS sheet for Thursday. Caleb will draft a scope and high level overview.