**CS 401 Group 4 Communication System Project**

Software Requirements Specification

Revision History

| **Date** | **Revision** | **Description** | **Author** |
| --- | --- | --- | --- |
| 02/08/2023 | 1.0 | Initial Version | Jonathan Lunn |
| 2/20/2023 | 1.1 | Requirements for chat logs & users. | Vicente Chavez |
| 02/26/2023 | 1.2 | Requirements for users module. | Victoria Swanson-Oswood |
| 2/27/2023 | 1.3 | Requirements for GUI | Vicente Chavez |
| 2/27/2023 | 1.4 | Requirements for Server | Charles Scrivens |
| 2/27/2023 | 1.5 | Requirements for Client | Xinye Daniel Zhang |
| 2/28/2023 | 1.6 | Definitions | Xinye Daniel Zhang |
| 2/28/2023 | 1.7 | UML Diagram | Vicente Chavez |
| 2/28/2023 | 1.8 | Product architecture, requirements for users | Victoria Swanson-Oswood |
| 2/28/2023 | 1.9 | External/Internal Interface Reqs. + non. Func. Requirements | Xinye Daniel Zhang |
| 3/1/2023 | 2.0 | Added the new message sequence diagram | Vicente Chavez |
| 03/01/2023 | 2.1 | Added Use Cases and Diagrams | Victoria Swanson-Oswood |
| 04/05/2023 | 2.2 | Added Use Case Diagram | Victoria Swanson-Oswood |
| 4/5/2023 | 2.3 | Added Sequence/ Reformatted Document | Xinye Daniel Zhang |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

**1.** **Purpose 4**

1.1. Scope 4

1.2. Definitions, Acronyms, Abbreviations 4

1.3. References 4

1.4. Overview 4

**2.** **Overall Description 5**

2.1. Product Perspective 5

2.2. Product Architecture 5

2.3. Product Functionality/Features 5

2.4. Constraints 5

2.5. Assumptions and Dependencies 5

**3.** **Specific Requirements 6**

3.1. Functional Requirements 6

3.2. External Interface Requirements 6

3.3. Internal Interface Requirements 7

**4.** **Non-Functional Requirements 8**

4.1. Security and Privacy Requirements 8

4.2. Environmental Requirements 8

4.3. Performance Requirements 8

# Purpose

This document outlines the requirements for the Communication system that we are developing.

## Scope

This document will catalog the user, system, and hardware requirements for the communication system. It will not, however, document how these requirements will be implemented.

## Definitions, Acronyms, Abbreviations

Users - person that is using this system include: Employees or Supervisors

Logger - a device/program that records events and data.

## References

Use Case Specification Document – Step 2 in assignment description

UML Use Case Diagrams Document – Step 3 in assignment description

Class Diagrams – Step 5 in assignment description

Sequence Diagrams – Step 6 in assignment description

## Overview

The communication program that we are developing is designed for a large business to have a text based contact system for inner organization conversations and communication. This will be server client based and have logs for all conversations.

# Overall Description

## Product Perspective

## Product Architecture

The system will be organized into 5 major modules: the client module, the server module,

the message thread module, the user module, and the GUI module.

## Product Functionality/Features

The high-level features of the system are as follows (see section 3 of this document for more detailed requirements that address these features):

## Constraints

String only messaging.

All messages are logged and stored.

Each User can only have one account.

Users can only access the system after login.

Users must start the client in order to access the UI.

Server must be up and running for users to send and receive messages.

## Assumptions and Dependencies

Each user has a unique identifier and only one for themselves to use.

Users have access to the internet.

Users belong to the purchasing company.

The user has access to a computer.

Users only need to send messages with string type.

Users have preset roles: General User and Admin User.

# Specific Requirements

## Functional Requirements

### Common Requirements:

3.1.1.1 All users must belong to the organization.

3.1.1.2 The messages have a size limit.

3.1.1.3 Users must be online to receive messages.

3.1.1.4 All Users must possess a Username and Password

### Client Module Requirements:

3.1.2.1 The inputs/requests/interactions from the User will be handled by the client which will send them to the server.

3.1.2.2. The client must be connected to the server.

3.1.2.3 The client must be able to retrieve data from the server.

3.1.2.4. The client will handle the GUI upon launching the program.

3.1.2.5 The client must be able to ping other clients/Users and get their status from the Server.

3.1.2.6 The client must be able to neatly display the data from the Server through the use of a GUI.

3.1.2.7 The client must log all the messages and logins into a log file.

### Server Module Requirements:

3.1.3.1 Server manages User file to verify

3.1.3.2 Server transmits Messages from Client to Client.  
3.1.3.3 Server transmits Messages from Client to Groups.

3.1.3.4 Server manages the connection status of the client between the client and the server.

3.1.3.5 Server manages the connection between the Message Thread and the Logs

3.1.3.6 Server manages authentication and verification of all users.

3.1.3.7 Server manages the state (status) of clients.

### Message Thread Module Requirements:

* + - 1. Message threads will only display the X most recent messages.
      2. Once a message thread is created, users will not be able to be added or removed.
      3. Only valid users can be a part of a message thread.
      4. A message thread must have at least 2 valid users.

### Users Module Requirements:

* + - 1. Users should be one of two types: Regular and Admins.
      2. Admins can see logs of all message threads.
      3. Regular users should not be able to view logs.
      4. All users will be able to create new groups.
      5. All users will be able to join existing groups.
      6. Users will be able to send and receive messages to other users.
      7. Users will each have unique username and password using a string of alphanumeric characters, with a maximum of 12 characters each.
      8. Users current status will be held to display inactivity for offline messaging.
      9. All of a users messages will be logged and can be accessed later by the users and admin.

## GUI Module Requirements

* + - 1. User will be prompted with a login screen upon launching client
      2. Username and password are input boxes with writable text.
      3. The login button will not be clickable until the Username and password fields have at least 1 character.
      4. The login button will be grayscale until the Username and password fields have at least 1 character.
      5. The User will be prompted with an invalid login message upon invalid login.
      6. User will be shown the homepage screen immediately after a successful login.
      7. The homepage for the user will have clickable buttons for each of the following actions: view chat log, create new chat log, logout.
      8. The homepage will display how many chat logs the user has unread message from.
      9. Upon clicking the logout button, the user will be taken to the initial login screen.
      10. Upon clicking the view chat log button, the user will be taken to the chat log screen.
      11. A chat log summary displays the chat log name, how many users are a part of it, date/time of latest message, notification if there are any unread messages in the chat log.
      12. The chat log screen will display all the chat logs summary the user belongs to in a scrollable box with a scroll bar.
      13. Each chat log summary will be in an individual container.
      14. Each chat log summary container will be selectable and will turn grayscale upon selection.
      15. If no chat log container is selected the chat log main box will be a blank white screen.
      16. Upon selection of a chat log container, that chat log will be displayed in the chat log main box.
      17. The chat log main box will display the last 30 messages in a container which can be scrolled.
      18. The chat log main box will have a new message input box with writable text
      19. The chat log main box will have a send button next to the new message input box, which will be grayscale until the new message box has at least 1 character.
      20. The chat log main box will display the users that belong to the selected chat log container.
      21. The chat log screen will have a clickable button to make new chat logs
      22. Upon clicking the new chat log button the user will be taken to the create chat log screen.
      23. The chat log screen will prompt the user to enter the users which will be a part of the chat log.
      24. The chat log screen will have a clickable button to create the new chat log.
      25. The create chat log button will be grayscale until the user has entered at least 1 valid user to the chat log.
      26. Upon clicking the create chat log button, the user will be taken to the chat log screen with the created chat log selected.

## External Interface Requirements

3.2.1 The chat system must provide a LoginUI that asks for User’s Login ID and Password.

3.2.2 A UserUI must be shown after successful login that is specific to the User’s role.

3.2.3 The system must have a general interface to neatly display information, such as active users, chats, and user states.

3.2.4 The system must have a general interface that allows the users to send a message, create a new message, and retrieve old messages.

## Internal Interface Requirements

3.3.1 The system must check if the user is active or inactive on the client side.

3.3.2 The system must ping the server occasionally to see that the client is still connected to the server.

3.3.3 The system must check that the data on the client side is current and updated.

# Non-Functional Requirements

## Security and Privacy Requirements

4.1.1 The general user is not able to see messages that are not sent to them.

4.1.2 The Administrators and IT users can review message logs for everyone in the organization.

4.1.3 A User’s Login ID and Password are created by administrators.

## Environmental Requirements

4.2.1 The Communication Application is built on Java using any IDE.

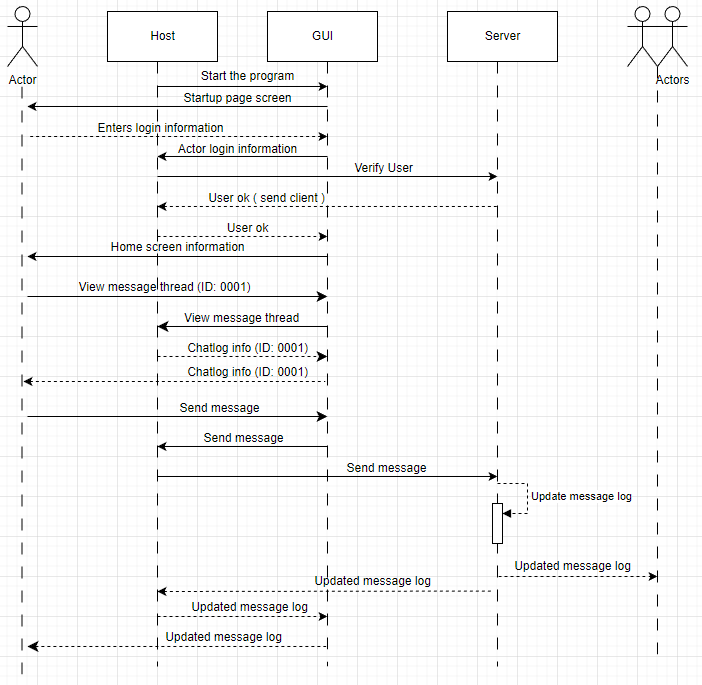
## Performance Requirements

4.3.1 The system must consistently update logs without interrupting the client and server connection

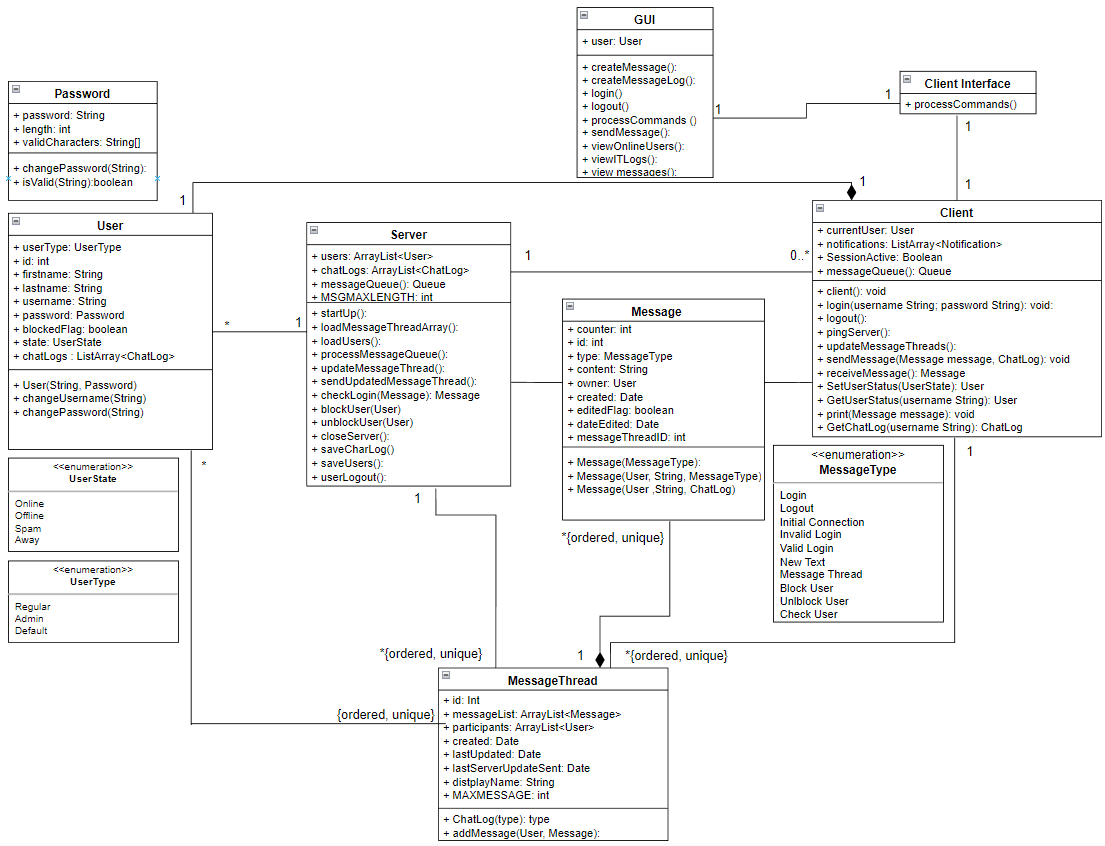
4.3.2 The server must be ran on a network with smooth latency

# Sequence Diagrams

## Login + New Message Sequence Diagram



# Class Diagram UML



# Use Cases

**Use Case ID:** 1

**Use Case Name:** User logging in and starting a new chat.

**Relevant requirements: 3.1.1.4, 3.1.5.7, 3.1.6.2, 3.1.6.1, 3.2.1, 4.1.3, 3.1.6.7, 3.1.6.26, 3.1.3.2, 3.1.6.5**

**Primary Actor:** User

**Pre-conditions:** User has an existing username and password. There are other users with accounts.

**Post-condition:** User has successfully logged in and new chat was created.

**Basic Flow or Main Scenario:**

1. User enter’s username and password

2. System verifies credentials

3. Status is changed to active

4. User selects option to create new chat

5. User inputs individual’s username whom they wish to chat with.

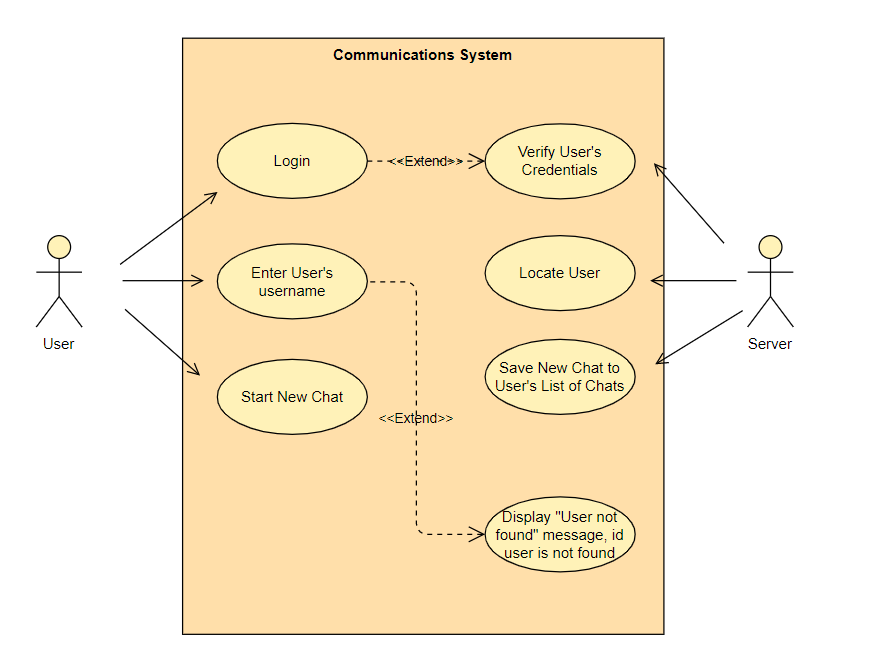
6 . If no chat already exists, a new chat is created.

**Extensions or Alternate Flows:**

1. User login is unsuccessful.

2. User is prompted to re-enter credentials.

**Exceptions:** If other user cannot be found, display message stating that the other user does not exist.



**Use Case ID:** 2

**Use Case Name:** User logging in and creating a new group

**Relevant requirements: 3.1.1.4, 3.1.5.7, 3.1.6.2, 3.1.6.1, 3.2.1, 4.1.3, 3.1.3.3, 3.1.5.4, 3.1.5.5, 3.1.6.5**

**Primary Actor:** User

**Pre-conditions:** User has an existing username and password. There are other users with accounts.

**Post-condition:** User has successfully logged in and created a group with multiple members.

**Basic Flow or Main Scenario:**

1. User enter’s username and password

2. System verifies credentials

3. Status is changed to active

4. User selects option to create new group

5. User inputs other users to be added to group

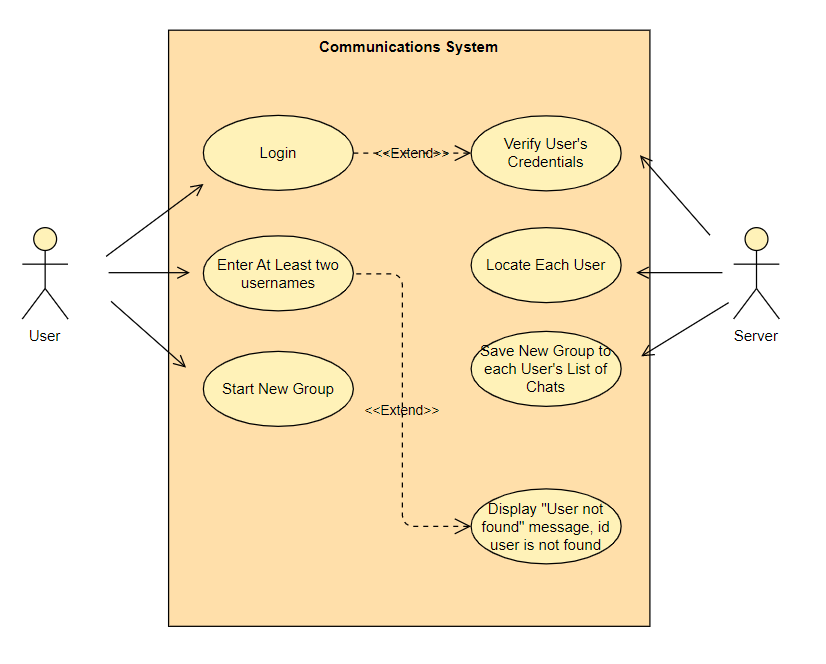
**Extensions or Alternate Flows:**

1. User login is unsuccessful.

2. User is prompted to re-enter credentials.

**Exceptions:** If user cannot be found, display message stating the user does not exist.

**Related Use Cases:** User logging in and creating a new chat



**Use Case ID:** 3

**Use Case Name:** User sending message in chat to inactive user

**Relevant requirements: 3.1.1.4, 3.1.5.7, 3.1.6.2, 3.1.6.1, 3.2.1, 4.1.3, 3.1.2.5, 3.1.3.7, 3.1.5.8, 3.1.6.5**

**Primary Actor:** User

**Pre-conditions:** User has an existing username and password. User that chat will be sent to is inactive.

**Post-condition:** User has logged in and message is sent and ready to be received when recipient logs in.

**Basic Flow or Main Scenario:**

1. User enter’s username and password

2. System verifies credentials

3. Status is changed to active

4. User selects an existing chat with an inactive user.

5. User composes message with text only and sends it to inactive user.

6. Message is stored and sent to user who is inactive when they log in.

**Extensions or Alternate Flows:**

1. User enter’s username and password

2. System verifies credentials

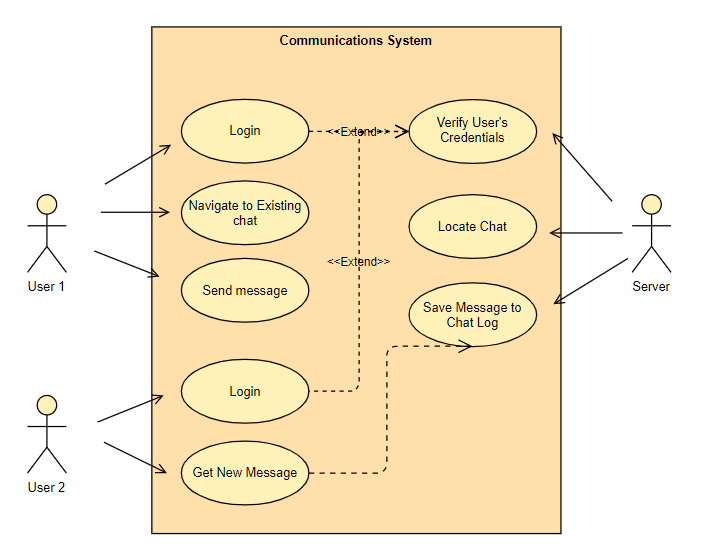
3. Status is changed to active

4. User composes message to be sent.

5. User is blocked by recipient.

6. Message is displayed explaining that the user is blocked.

**Exceptions:** If user cannot be found, display message stating the user does not exist.



**Use Case ID:** 4

**Use Case Name:**  User sending message to active user.

**Relevant requirements: 3.1.1.4, 3.1.5.7, 3.1.6.2, 3.1.6.1, 3.2.1, 4.1.3, 3.1.2.5, 3.1.3.7, 3.1.5.8 , 3.1.6.5**

**Primary Actor:** User

**Pre-conditions:** User has an existing username and password. User that message will be sent to is active.

**Post-condition:** User logs in successfully and message is received immediately by recipient.

**Basic Flow or Main Scenario:**

1. User enter’s username and password

2. System verifies credentials

3. Status is changed to active

4. User selects an existing chat with an active user.

5. User composes message with text only and sends it to active user.

6. Message is created and sent to its recipient.

**Extensions or Alternate Flows:**

1. User enter’s username and password

2. System verifies credentials

3. Status is changed to active

4. User composes message to be sent.

5. User is blocked by recipient.

6. Message is displayed explaining that the user is blocked.

**Exceptions:** If user cannot be found, display message stating the user does not exist..

**Related cases:** User sending message to inactive user.

**Use Case ID:** 5

**Use Case Name:**  User updates current status

**Relevant requirements: 3.1.1.4, 3.1.5.7, 3.1.6.2, 3.1.6.1, 3.2.1, 4.1.3, 3.1.2.5, 3.1.3.4, 3.1.3.7, 3.1.5.8, 3.1.6.5**

**Primary Actor:** User

**Pre-conditions:** User has an existing account.

**Post-condition:** User has successfully logged in and status has been changed.

**Basic Flow or Main Scenario:**

1. User enters their username and password.

2. Server verifies credentials and login is successful.

3. Status is updated to active once logged in.

4. User selects desired status.

5. New status is displayed only to users that do not have the user blocked.

**Extensions or Alternate Flows:**

1. User enters username and password

2. Display that credentials are not valid and request new login.

**Exceptions:** User has flagged a group chat and has been blocked by an admin.

**Related Use Cases:** Admin blocks a user from messaging in a group; User is flagged for spamming group chats

**Use Case ID:** 6

**Use Case Name:**  Admin logs in and views chat logs.

**Relevant requirements: 3.1.6.1, 3.1.6.5, 3.2.1, 3.1.5.1, 3.1.5.2, 3.1.5.9, 4.1.2**

**Primary Actor:** Admin

**Pre-conditions:** Admin has an existing account. Some chat logs have been saved.

**Post-condition:** Admin has file with chat log history.

**Basic Flow or Main Scenario:**

1. Admin enters username and password.

2. System verifies credentials.

3. Admin selects option to view chat logs.

4. System verifies that user is an administrator.

4. System retrieves requested log and returns it to the admin.

**Extensions or Alternate Flows:**

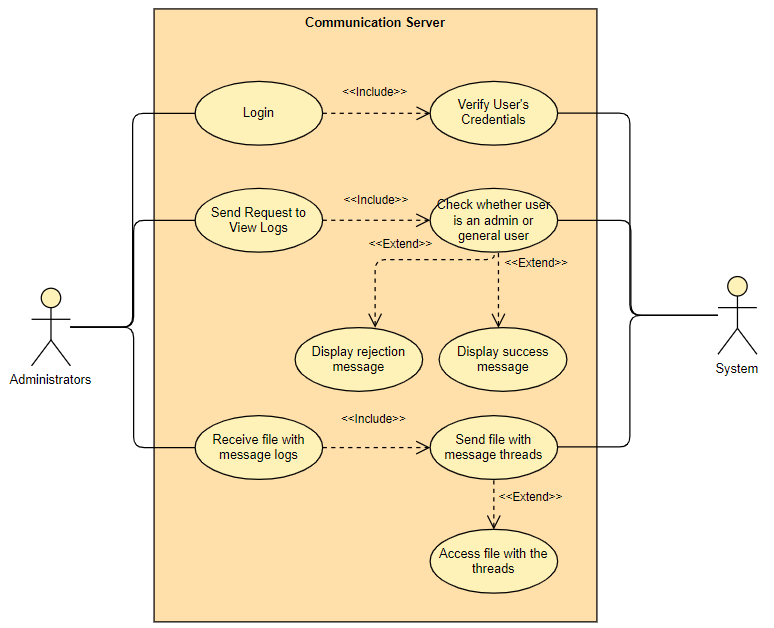
1. User enters username and password

2. System verifies credentials as user only

3. User is shown error message when trying to view the logs.

**Exceptions:** No chat logs exist, so the admin does not have anything to access.

**Related Use Cases: n/a**



**Use Case ID:** 7

**Use Case Name:** User blocks another user

**Relevant requirements: 3.1.1.4, 3.1.5.7, 3.1.6.2, 3.1.6.1, 3.2.1, 4.1.3, 3.1.2.5, 3.1.3.4, 3.1.3.7, 3.1.5.8, 3.1.6.5**

**Primary Actor:** User

**Pre-conditions:** User has an existing account. Second user has an existing account.

**Post-condition:** Second user’s status has been changed to blocked when viewed by first user.

**Basic Flow or Main Scenario:**

1. User enters their username and password.

2. Server verifies credentials and login is successful.

3. Status is updated to active once logged in.

4. User navigates to user that they want to block

5. User blocks individual and system changes their status to blocked.

**Extensions or Alternate Flows:**

n/a

**Exceptions:** If user is already blocked, display a message stating that they are.

Related Use Cases: n/a

**User Case ID:** 8

**Use Case Name:** User is flagged for spamming group chats

**Relevant requirements: 3.1.1.4, 3.1.5.7, 3.1.6.2, 3.1.6.1, 3.2.1, 4.1.3**

**Primary Actor:** System

**Pre-conditions:** User has an existing account and is logged in.

**Post-condition:** User status is changed to blocked and they can no longer message the chat.

**Basic Flow or Main Scenario:**

1. User enters their username and password.

2. Server verifies credentials and login is successful.

3. Status is updated to active once logged in.

4. User messages chat beyond the reasonable limit of messages per minute.

5. System flags user’s account

6. System updates user status to blocked and displays message.

**Extensions or Alternate Flows:**

1. User enters their username and password.

2. Server verifies credentials and login is successful.

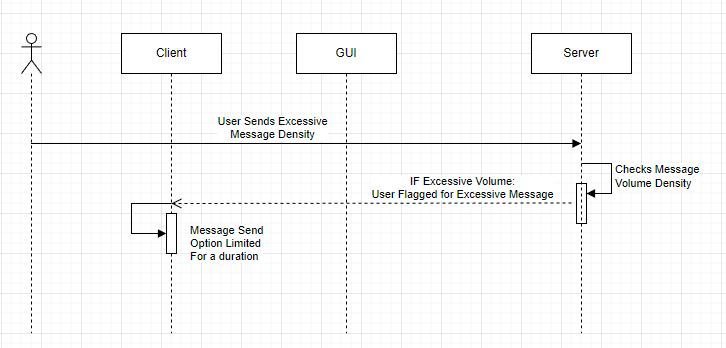
3. Status is updated to active once logged in.

4. System flags user for spamming chat

5. Admin changes user status to blocked

**Exceptions:** If user status changes to away before being blocked, user’s status will be changed to blocked when changing it back to active

**Related Use Cases: n/a**



**User Case ID:** 9

**Use Case Name:** User changes their password.

**Relevant requirements: 3.1.1.4, 3.1.5.7, 3.1.6.2, 3.1.6.1, 3.2.1, 4.1.3**

**Primary Actor:** User

**Pre-conditions:** User has an existing account.

**Post-condition:** User has a new password

**Basic Flow or Main Scenario:**

1. User enters their username and password.

2. Server verifies credentials and login is successful.

3. Status is updated to active once logged in.

4. User selects option to change password

5. User enters new password

6. System checks length and updates password.

**Extensions or Alternate Flows:**

1. User enters their username and password.

2. Server verifies credentials and login is successful.

3. Status is updated to active once logged in.

4. User selects option to change password

5. User enters new password

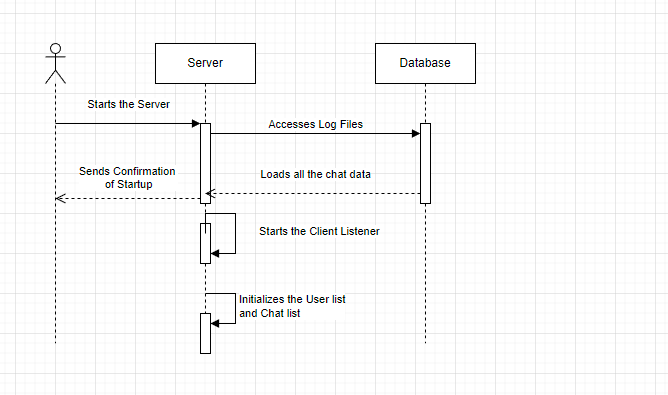
6. System finds that password exceeds the maximum password length

7. System requests new passwords.

**Exceptions:** If user enters existing password when trying to change password, password will not change.

**Related Use Cases: n/a**

**Server Startup Sequence Diagram**



**Client Startup Sequence Diagram**

