**Group 4 – Communications Systems**

Software Requirements Specification

Revision History

| **Date** | **Revision** | **Description** | **Author** |
| --- | --- | --- | --- |
| 02/21/2025 | 1.0 | Initial Version & Updated Section 1 | Talhah Shaik |
| 02/24/2025 | 1.1 | Added Requirements in Section 3 & 4 | Talhah Shaik |
| 2/25/2025 | 1.2 | Added UML Use Case Diagram | Kenny Kottenstette |
| 2/25/2025 | 1.3 | Added Use Cases | Zoheb Sharif |
| 2/27/2025 | 1.4 | UML Use Case Diagram Refactor | Kenny Kottenstette |
| 2/27/2025 | 1.5 | Added Requirements & Use Cases | Kenny Kottenstette |
| 2/27/2025 | 1.6 | Updated Definitions and Requirements | Talhah Shaik |
| 2/28/2025 | 1.7 | Finalized Section 2 | Zoheb Sharif |
| 3/1/2025 | 1.8 | Updated Section 1 - Definitions | Talhah Shaik |
| 3/1/2025 | 1.9 | Add UML Class Diagram | Kenny Kottenstette |
| 3/2/2025 | 2.0 | Add IT Admin Sequence Diagram | Kenny Kottenstette |
| 3/2/2025 | 2.1 | Finalized Section 3 | Zoheb Sharif |
| 3/3/2025 | 2.2 | Added Use Cases | Zoheb Sharif |
| 3/3/2025 | 2.3 | Created User Sequence Diagram | Talhah Shaik |
| 3/3/2025 | 2.4 | Inserted Diagrams in Document | Talhah Shaik |
| 3/3/2025 | 2.5 | Updated use cases 1,5 (include re-enable), 8 | Kenny Kottenstette |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[**1. Purpose 4**](#_heading=h.1fob9te)

[1.1. Scope 4](#_heading=h.3znysh7)

[1.2. Definitions, Acronyms, Abbreviations 4](#_heading=h.2et92p0)

[1.3. References 4](#_heading=h.tyjcwt)

[1.4. Overview 4](#_heading=h.3dy6vkm)

[**2. Overall Description 5**](#_heading=h.1t3h5sf)

[2.1. Product Perspective 5](#_heading=h.4d34og8)

[2.2. Product Architecture 5](#_heading=h.2s8eyo1)

[2.3. Product Functionality/Features 5](#_heading=h.17dp8vu)

[2.4. Constraints 5](#_heading=h.3rdcrjn)

[2.5. Assumptions and Dependencies 5](#_heading=h.lnxbz9)

[**3. Specific Requirements 6**](#_heading=h.35nkun2)

[3.1. Functional Requirements 6](#_heading=h.1ksv4uv)

[3.1.1. Common Requirements: 6](#_heading=)

[3.1.2. Server Module Requirements: 6](#_heading=)

[3.1.3. Client Module Requirements: 6](#_heading=)

[3.1.4. IT Module Requirements: 6](#_heading=)

[3.2. External Interface Requirements 6](#_heading=h.4i7ojhp)

[3.3. Internal Interface Requirements 7](#_heading=h.1ci93xb)

[**4. Non-Functional Requirements 8**](#_heading=h.qsh70q)

[4.1. Security and Privacy Requirements 8](#_heading=h.3as4poj)

[4.2. Environmental Requirements 8](#_heading=h.49x2ik5)

[4.3. Performance Requirements 8](#_heading=h.23ckvvd)

[**5. Use Case Specifications Document 9**](#_heading=h.62jut7x90ccq)

[**6. UML Use Case Diagram 18**](#_heading=h.kyb8e5m22308)

[**7. UML Class Diagram 19**](#_heading=h.pwen6miihlhp)

[**8. UML Sequence Diagrams 20**](#_heading=h.soy5aty0hiuz)

# Purpose

This document outlines the requirements for the Communications System.

## Scope

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

## Definitions, Acronyms, Abbreviations

Message - A piece of text which is kept as a string, with no ability to send images and videos

User - An entity that is able to log into the System and use the services of an application at a basic level

Admin (IT) - An elevated entity that can access all messages sent and control the users (create and delete)

Client - An entity which the user interacts with and has the ability to talk to the server by sending requests and information.

Server - An entity which listens a/multiple clients at the same time, and holds information given by a client

## References

Github Repository - <https://github.com/ZohebSharif/communicationsGroup4>

Use Case Specification Document – Page 9 & Attached

UML Use Case Diagrams Document – Page 17 & Attached

Class Diagrams – Page 18 & Attached

Sequence Diagrams – Page 19 & Attached

## Overview

A communications system for a very large organization which allows employees to communicate over chat both synchronously and asynchronously. Users should be able to chat privately and in groups. All conversations should be logged and viewable by the IT users. This will only be a text chat with no access to audio videos content or communication. This is a Java application with a GUI that operates over TCP/IP. This system requires a server application and client application. There is no web or HTML component.

# Overall Description

## Product Perspective

The system will be based on the client-server design. The client module is for user chat functionality, the server module is for logging messages and managing users, the IT module is for admin access to all messages and managing users

## Product Architecture

The system will be organized into 3 major modules: the Server module, the Client module, and the IT 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):

* Private chats with one user to another user
* Group chats with multiple users
* Messages are kept in a persistent log
* Admin has access to this persistent log, therefore has access to all messages
* IT users can manage accounts(create, delete, disable)
* Message delivery techniques ensure messages can be sent when users are both online, or when a user is offline

## Constraints

* Not accessible by browser
* Text is the only accepted media (no videos, pictures, gifs, etc)
* Uses TCP/IP
* Application will be made using Java programming language

## Assumptions and Dependencies

* Both client and server are Java applications
* Client must have a stable connection to the server
* The server is assumed to be stable

# Specific Requirements

## Functional Requirements

### Common Requirements:

3.1.1.1 User login is mainly handled on the system the application is running on and will be a soft login on application start

3.1.2.2 Users are given a login name that’s comprised of the first three letters of their first name, and first three letters of their last name (ex: Kenny Kottenstette → KenKot)

3.1.2.3 Users are given a password that must be 8 characters long, include 1 capital letter, 1 digit and 1 symbol

3.1.2.4 Authentication is required during login to restrict authority according to user roles

3.1.2.5 Client will be used by both admin and general users

3.1.2.6 Maintain storage of login information on a plain text document

### Server Module Requirements:

3.1.2.1 Messages are logged in terms of user, time, and to whom it is sent to

3.1.2.2 All messages logged will be accessible to IT Users with minimal restrictions

3.1.2.3 Messages that are not able to be given to a client, should send it in order of when the client re-connects to the server

3.1.2.4 Support both real-time messaging alongside asynchronous messaging

3.1.2.5 Give an error message when messages fail to send

### Client Module Requirements:

3.1.3.1 Users should be able to create their own group or private chats

3.1.3.2 All messages sent are saved with no ability to edit or remove it

### IT Module Requirements:

3.1.4.1 IT Users are able to access all messages no matter where they are from

3.1.4.2 Messages that are removed from the chat are still existing in both saved history and visible to the IT User

## External Interface Requirements

3.2.1 The system will be split into different sections - A private chats sections, a group chats section, and a message history section where you can see the current history of a selected chat.

3.2.2 Client will be responsive

3.2.3 Users must be able to clearly see and select recipients

3.3.3 Chat will have a field to type, as well as view the recipient’s and user’s previous chats

## Internal Interface Requirements

3.3.1 The system must be able to handle multiple different clients at the same time while also saving all of its actions in logged form

3.3.2 System must have a persistent log of previous chats

# Non-Functional Requirements

## Security and Privacy Requirements

* + 1. The messages sent from clients are supposed to only send it to the recipients that the client sends them too.
    2. System access requires valid credentials
    3. IT Admins can disable accounts, invalidating those login credentials
    4. Communication between client and server will be unencrypted.

## Environmental Requirements

* + 1. Client and Server applications must be standalone Java applications
    2. Client and Server must communicate via TCP/IP
    3. Display connection error if no network is found or if network connection is lost

## Performance Requirements

* + 1. No HTML or web components will be a part of this application, it will strictly be Java.
    2. The server will use TCP/IP based server to handle communication between users.
    3. The server should be able to handle a large number of concurrent users effectively.

# Use Case Specifications Document

**Use Case: 01**

**Use Case Name:** User Login

**Relevant Requirements:** Users must exist before logging in, users must have valid

credentials that are saved on system

**Primary Actor:** User, IT User

**Pre-Conditions:**

* Server is online
* User’s credentials is in data file
* Client has connection to server
* Client has user list and all their chat message data

**Post-Conditions:**

* The user is now logged into the system

**Basic Flow or Main Scenario:**

1. The user logs into the client
2. The user is prompted with a login screen
3. The user enters their login info
4. The system validates the login info
5. If valid, the user is given access as well as user list and all chat message data

**Extensions or Alternate Flows:**

* Invalid Credentials - If users provide invalid credentials, an error message will pop up and the user will have to provide new login details again.
* Server Outage - If the user is trying to login while the server is down, an error message will pop up saying that the server is down.

**Exceptions:**

* User has connectivity issues

**Related Use Cases:** N/A

**Use Case: 02**

**Use Case Name:** Sending Message

**Relevant Requirements:** User must be logged in, system ensures message delivery

**Primary Actor:** User

**Pre-Conditions:**

* User has successfully logged into the system
* User must have a recipient to send the message to
* System is online

**Post-Conditions:**

* The message has been sent successfully

**Basic Flow or Main Scenario:**

1. After login, user is in messaging interface
2. User selects a recipient or group chat
3. User types a message
4. User sends message
5. System sends message

**Extensions or Alternate Flows:**

Message Content - Messages can only be text, no images or video can be sent

**Exceptions:**

* Recipient does not exist

**Related Use Cases:** N/A

**Use Case: 03**

**Use Case Name:** Create Group Chat

**Relevant Requirements:** User must have logged in successfully, user must select recipients

**Primary Actor:** Users

**Pre-Conditions:**

* User must be successfully logged in

**Post-Conditions:**

* A new group chat is made with the new participants

**Basic Flow or Main Scenario:**

1. User has logged into the system
2. The user creates a group chat
3. The user selects the recipients to add to the group chat
4. The user gives the chatroom a name
5. System creates group chat with all added recipients
6. The user can now send messages in this new group chat

**Extensions or Alternate Flows:**

* None

**Exceptions:**

* To make a group chat, you must have at least two recipients. If two recipients don’t exist for the user to add, an error message is thrown
* If the chatroom’s name is already taken, an error message is thrown

**Related Use Cases:** N/A

**Use Case: 04**

**Use Case Name:** Creating User

**Relevant Requirements:** IT User must be logged in

**Primary Actor:** IT User

**Pre-Conditions:**

* IT User has successfully logged into the system

**Post-Conditions:**

* The new user has been successfully created

**Basic Flow or Main Scenario:**

1. After login, IT user navigates to admin interface
2. Admin enters users first name, last name and username
3. Admin receives a success message of the new user being created

**Extensions or Alternate Flows:**

* If User’s first name and last name are under 3 characters in length, just use the the 1 or 2 characters (Ex: Kenneth Ko → KenKo)
* Update Users - change name or details

**Exceptions:**

* If username already exists, IT User should add a numeric postfix (ex: KenKot2)

**Related Use Cases:** N/A

**Use Case: 05**

**Use Case Name:** Disabling/Re-enabling User

**Relevant Requirements:** IT User must be logged in

**Primary Actor:** IT User

**Pre-Conditions:**

* IT User has successfully logged into the system
* The User to be disabled/re-enabled exists
* The User to be disabled isn’t an IT User themselves

**Post-Conditions:**

* The target user is disabled/re-enabled and their login credentials are invalid for future system access

**Basic Flow or Main Scenario:**

1. After login, IT User navigates to admin interface
2. IT User selects target User and changes their account to disabled/re-enabled
3. Admin receives a success message of the new user being disabled/re-enabled

**Extensions or Alternate Flows:**

* If target User is an IT User, the option shouldn’t be listed in the UI

**Exceptions:**

**Related Use Cases:** N/A

**Use Case: 06**

**Use Case Name:** Receive Message

**Relevant Requirements:** User must be logged in, system must be online, received message has recipient and time stamp

**Primary Actor:** User

**Pre-Conditions:**

* User has logged into the system with verified credentials
* System is online

**Post-Conditions:**

* The message is received by the user
* The message will be recorded in the persistent log

**Basic Flow or Main Scenario:**

1. The user logs in
2. The system constantly checks if a message was sent to the user until the user logs out again
3. When a message is received, it will be shown in the chat log with the user and the recipient

**Extensions or Alternate Flows:**

* If the user is offline before being sent a message: The system sends a message with the recipient's name and the timestamp of the message

**Exceptions:**

* If the server fails to send the message, on which every user has the outgoing message, explain the error

**Related Use Cases:** User login, Sending Message

**Use Case: 07**

**Use Case Name:** Logout

**Relevant Requirements:** User must have successfully logged in with valid credentials

**Primary Actor:** User

**Pre-Conditions:**

* User must have successfully logged in with valid credentials

**Post-Conditions:**

* Client closes
* The user is logged out and the session is terminated

**Basic Flow or Main Scenario:**

1. Use selects the logout option from interface
2. The system asks for confirmation of logout
3. System terminates user session
4. User is logged out, is now in

**Extensions or Alternate Flows:**

* If the user cancels the logout confirmation, their login session continues

**Exceptions:**

* If the session termination fails, the system maintains the session and retries the termination

**Related Use Cases:** User Login

**Use Case: 08**

**Use Case Name:** Write group chat log to file

**Relevant Requirements:** User must have IT Admin credentials

**Primary Actor:** ITUser

**Pre-Conditions:**

* IT User must have successfully logged in with valid credentials
* IT User must be currently viewing the group chat to log

**Post-Conditions:**

* Text file of the chat is written to a .txt file on the client

**Basic Flow or Main Scenario:**

1. IT Admin views a group chat
2. IT Admin in the GUI clicks button to log the chat
3. The chat conversation is written to .txt on the client

**Extensions or Alternate Flows:**

* If the group chat

**Exceptions:**

* If file writing operation fails, the user is notified

**Related Use Cases:**

**Use Case: 09**

**Use Case Name:** Manage Group Chats

**Relevant Requirements:** Group chat must exist, users must be logged in.

**Primary Actor:** User

**Pre-Conditions:**

* Group chat must have already been created.
* Users must be logged in.
* Must have other user accounts to add to the group.

**Post-Conditions:**

* User will be added from group chat.
* User will be deleted.
* Group chat will be renamed.

**Basic Flow or Main Scenario:**

1. User selects the manage icon.
2. Select one of the available manage options.
3. User will be prompted to add/delete user, or rename chat.

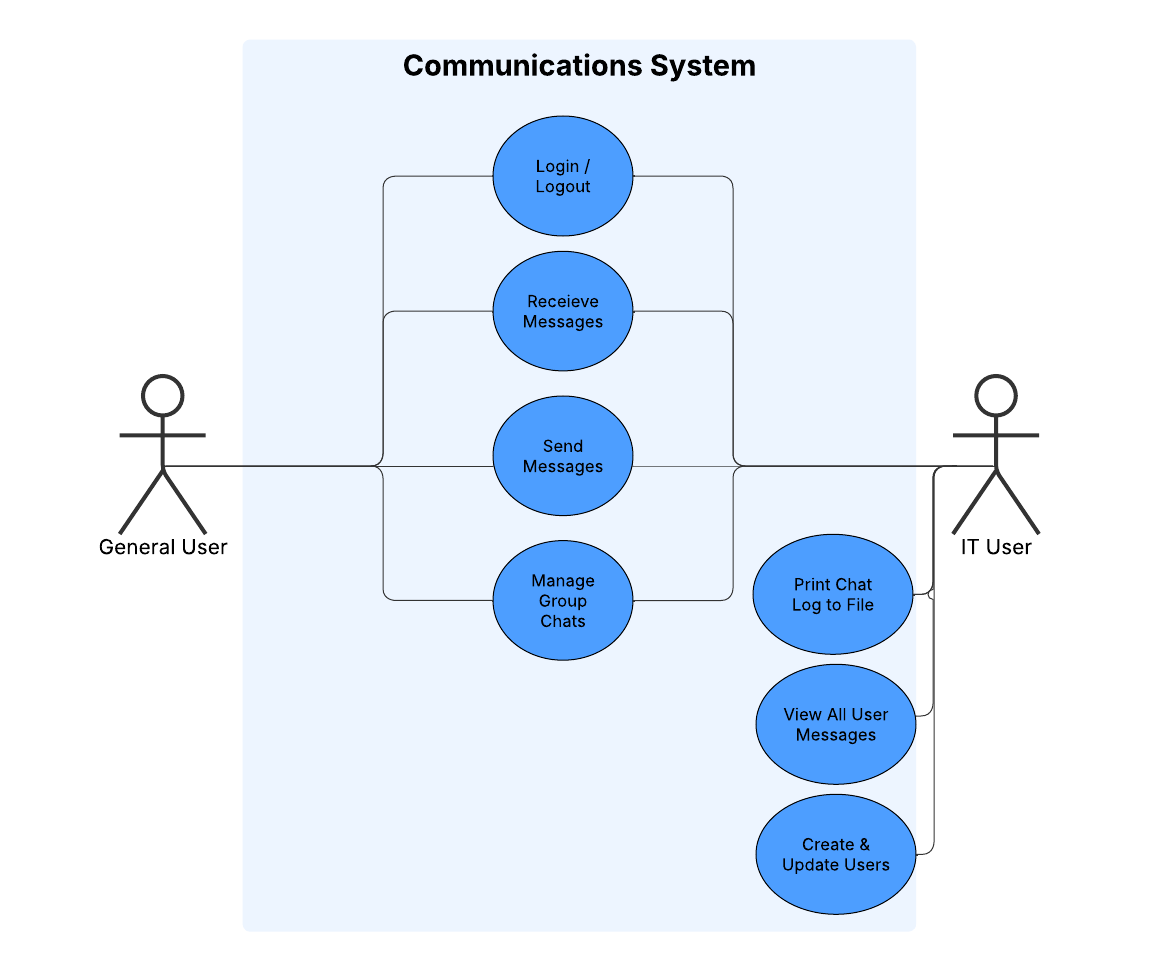
**Extensions or Alternate Flows:**

* Add user to group chat
* Delete user from chat
* Rename chat

**Exceptions:** Group chat doesn’t exist.

**Related Use Cases:** Create Group Chat

# UML Use Case Diagram



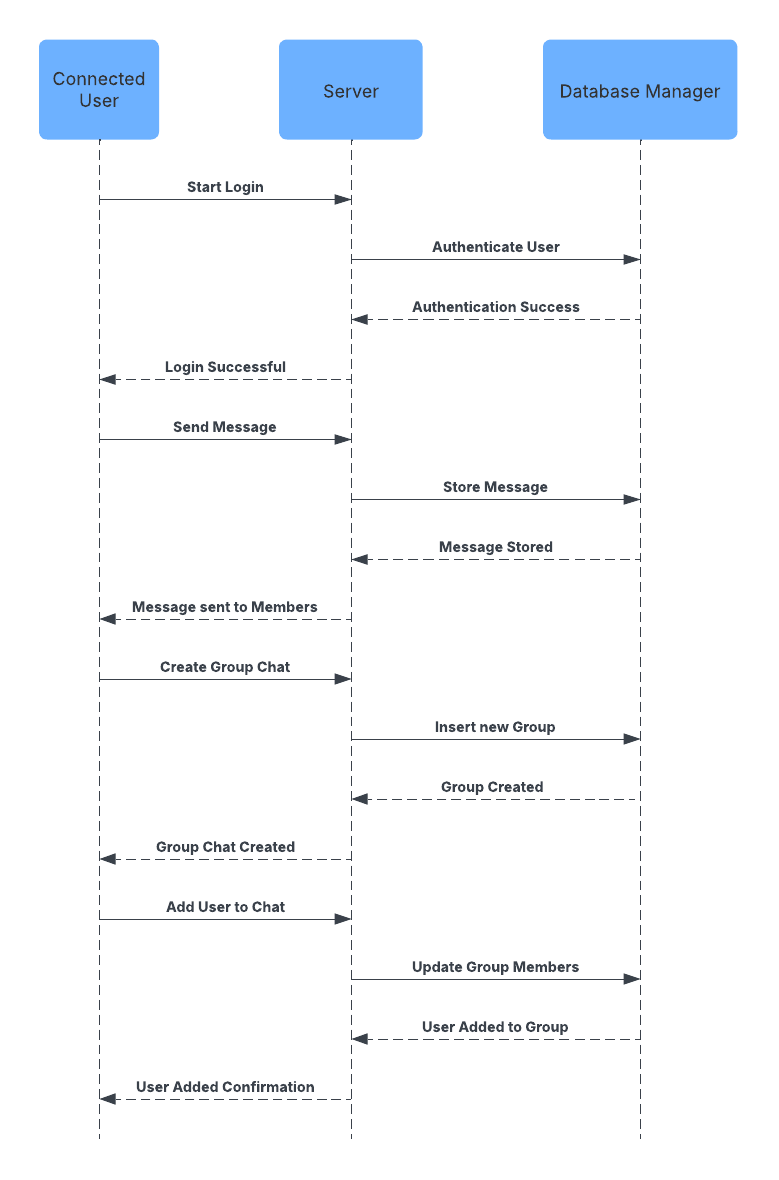
<https://lucid.app/lucidchart/79f676a1-a0df-4445-9a19-45afb0fd6478/edit?invitationId=inv_fa09cacd-d60e-4a1c-8470-3d7e731b3e7b&page=.Q4MUjXso07N#>

# UML Class Diagram

<https://lucid.app/lucidchart/319df949-aa53-452e-b47e-1c06d6b0a7ab/view>

# UML Sequence Diagrams

<https://lucid.app/lucidchart/ae1e5cda-b11f-404e-9534-9c744039bd6e/edit?viewport_loc=922%2C374%2C2520%2C1287%2C0_0&invitationId=inv_b60cba0e-cb72-467f-9096-0eab7284f70f>



<https://lucid.app/lucidchart/d043f260-d5f1-4bff-b40c-4f9442b363c8/edit?invitationId=inv_347f206c-61e5-4500-96c8-1329e1003d49&page=0_0#>