**Group 7 Software Design Document** *WeDiscuss Communications Application  
  
  
  
Github:* [*https://github.com/rmncbr/cs401-group7-communications*](https://github.com/rmncbr/cs401-group7-communications)

### Revision History

| **Date** | **Revision** | **Description** | **Author** |
| --- | --- | --- | --- |
| 10/17/2024 | 1.0 | Initial Release | Roman Cabrera |
| 10/21/2024 | 1.5 | Redesigned Document Layout | Joseph Dominguez |
| 10/29/2024 | 1.6 | ClientUI and AdminUI class design | Roman Cabrera |
| 10/29/2024 | 1.7 | Client, Server, Message, MessageCreator, LogManager class | Joseph Dominguez |
| 10/29/2024 | 1.8 | User & UserManager Class | Salvador Alvarez |
| 10/29/2024 | 1.9 | Chatroom & ChatroomManager Class | Isaiah Meza |
| 10/30/2024 | 2.0 | Use Cases Added | All Members |
| 10/30/2024 | 2.1 | Section 4: All Sequence Diagrams Added | Marcos Gomez |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**ii**

### Table of Contents

**1. PURPOSE................................................................................................................................................4**

1.1. SCOPE ....................................................................................................................................................4 1.2. DEFINITIONS, ACRONYMS, ABBREVIATIONS .........................................................................................4 1.4. REFERENCES......................................................................................................................................4

**2. USE CASE OVERVIEW.......................................................................................................................5**

2.1. SERVER/CLIENT DIAGRAM ....................................................................................................................5 2.2. USER/CHAT DIAGRAM............................................................................................................................5 2.3. LOG DIAGRAM........................................................................................................................................5 2.4. CONSTRAINTS.........................................................................................................................................5 2.5. ASSUMPTIONS AND DEPENDENCIES..................................................................................................5

**3. OVERALL CLASS DESIGN.................................................................................................................6**

3.1. TOP-LEVEL UML CLASS DIAGRAM DESIGN DESCRIPTION....................................................................6 3.2. INDIVIDUAL CLASS DESIGNS .............................................................................................................6

3.2.1 SERVER/CLIENT MODULE......................................................................................................7  
3.2.2 USER MODULE......................................................................................................................7  
3.2.3 LOG MODULE........................................................................................................................7  
3.2.4 CHAT MODULE......................................................................................................................7  
3.2.5 GUI MODULE........................................................................................................................7

**4. SEQUENCE DIAGRAMS......................................................................................................................9**

4.1 LOGIN/LOGOUT SEQUENCE DIAGRAM................................................................................................9  
4.2 SEND MESSAGE TO USER SEQUENCE DIAGRAM..............................................................................9  
4.3 SEND MESSAGE TO CHATROOM SEQUENCE DIAGRAM..................................................................9  
4.4 CREATE CHATROOM SEQUENCE DIAGRAM..............................................................................................9  
4.5 INVITE USER TO CHATROOM SEQUENCE DIAGRAM.......................................................................9

4.6 LEAVING CHATROOM SEQUENCE DIAGRAM.....................................................................................9

4.7 ADMIN ACTIONS SEQUENCE DIAGRAM..............................................................................................9

**iii**

## 1. Purpose

The Group 7 Software Design Document outlines the architecture and system design for WeDiscuss, a communications system project. WeDiscuss is designed to help employees of an organization communicate effectively both synchronously and asynchronously, enabling private and public messaging. This document is intended for Project Managers, Software Engineers, and anyone else who will be involved in the implementation of the system.

#### 1.1. Scope

This document describes the implementation details of WeDiscuss Communications Application. WeDiscuss will consist of five major components: The Server/Client module, the User module, the Log module, the Chat module, and the GUI module. Each of the components will be explained in detail in this design document.

#### 1.2. Definitions, Acronyms, Abbreviations

* UTU - User to User message
* UTC - User to Chatroom message

#### 1.3. References

Group 7 *Software Specifications Document*

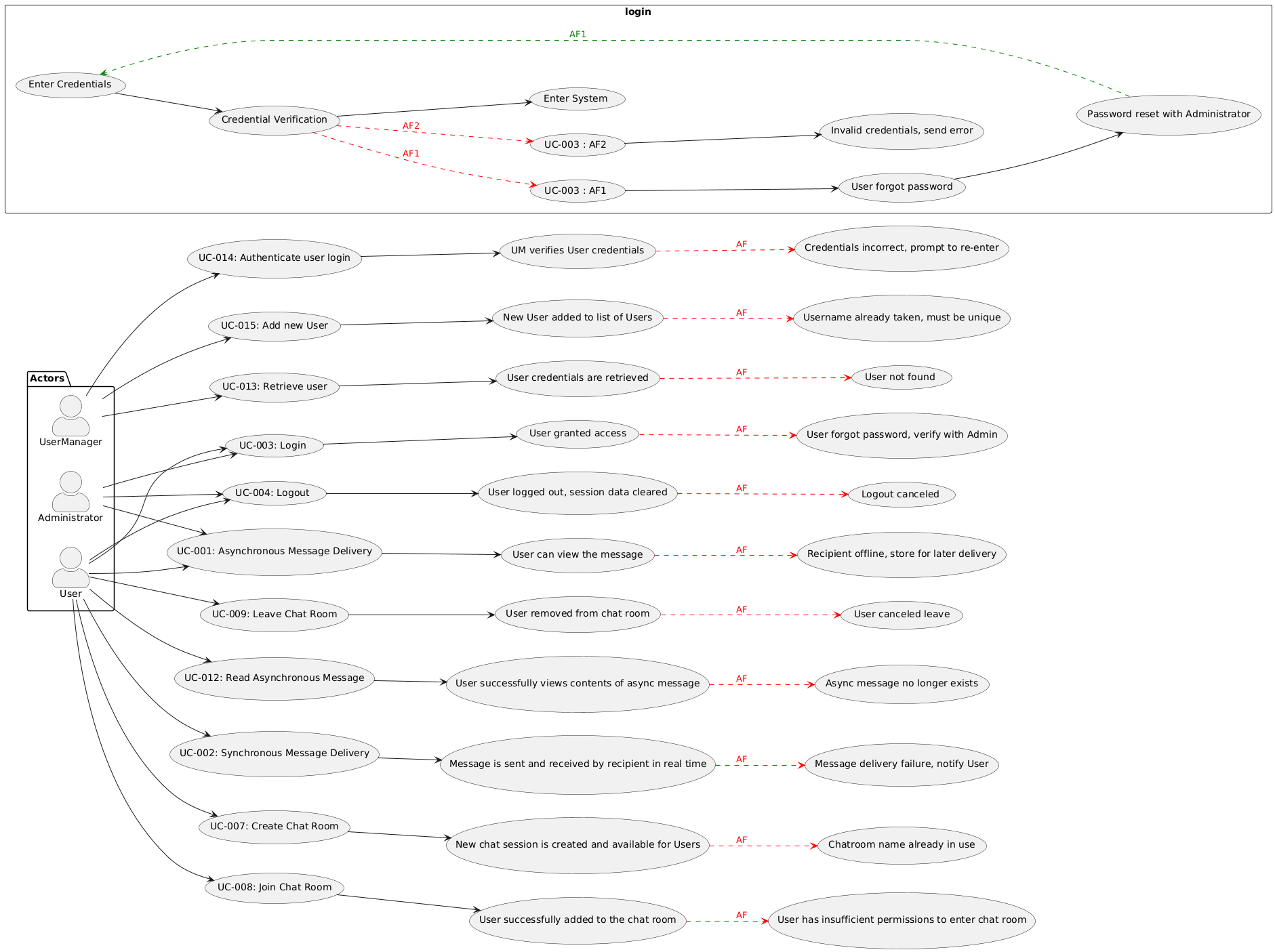
## 2. Use Case Overview

#### 2.1 Server/Client Module Use Case Diagram

#### 

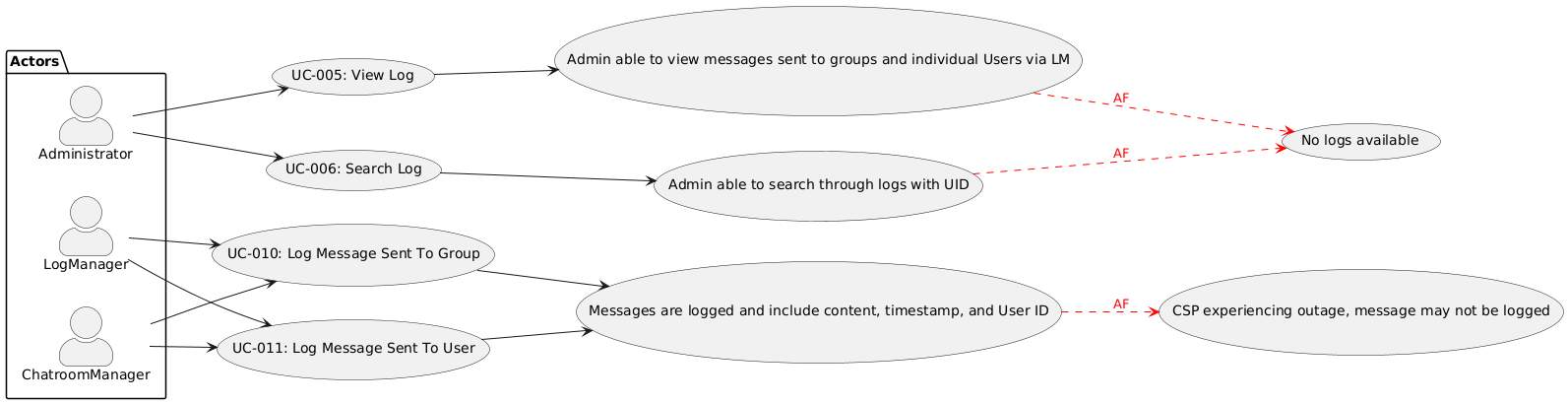
#### 

#### 2.2 User/Chat Module Use Case Diagram



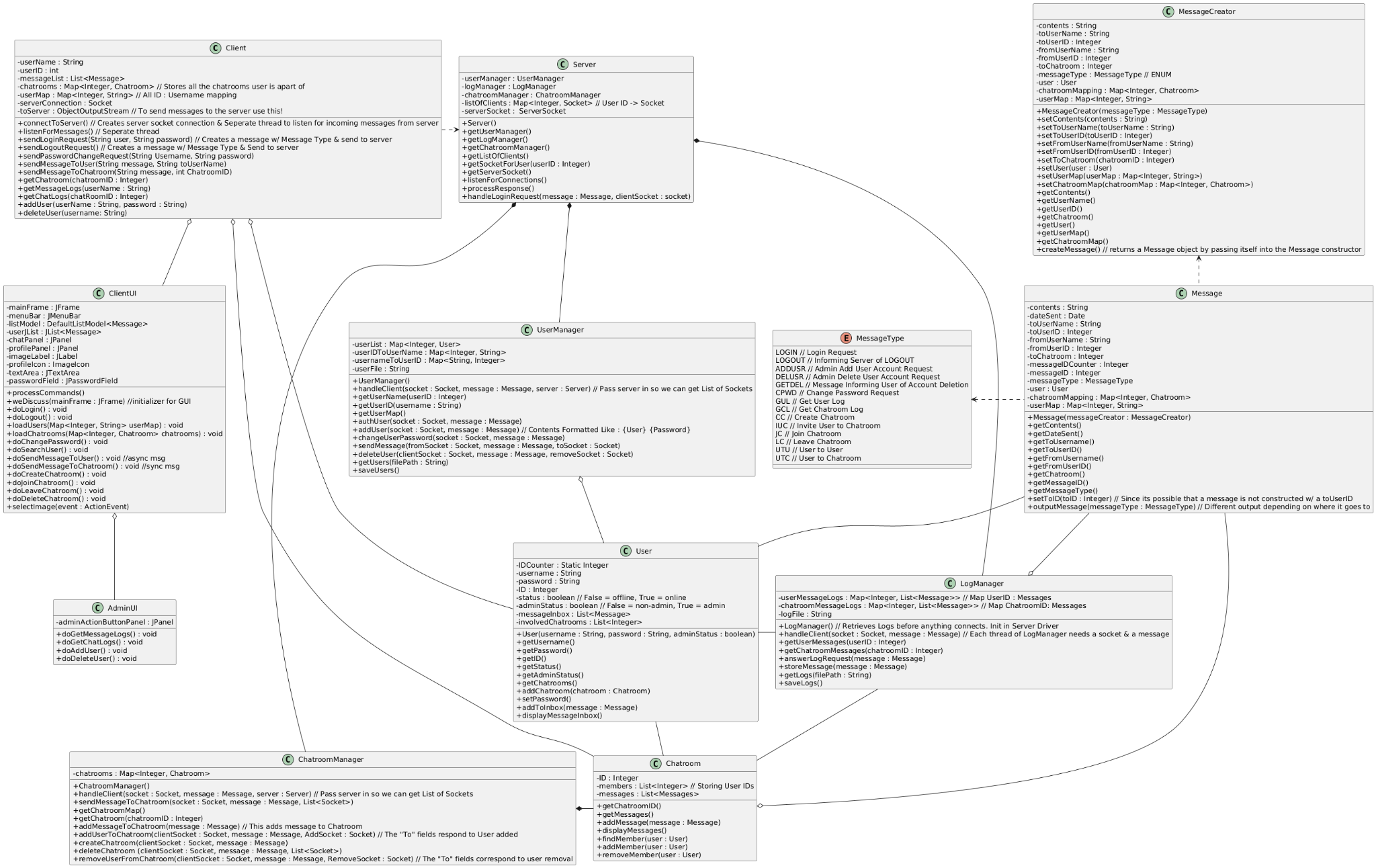
#### 

#### 2.3 Log Module Use Case Diagram



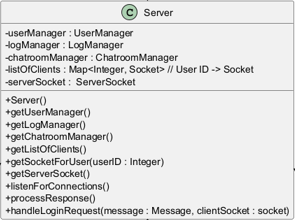
## 3. Overall Class Design

#### Top-Level UML Class Diagram Design

****

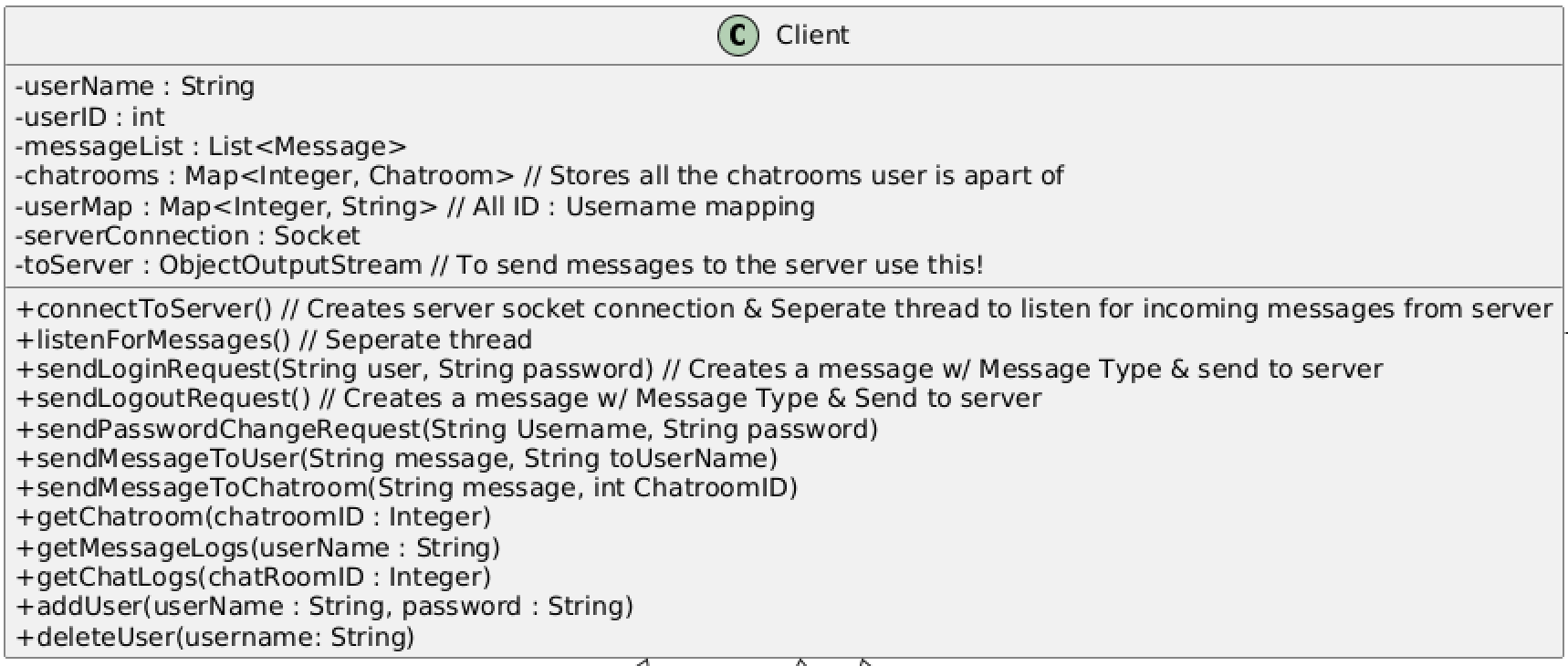
#### Individual Class Designs

* 1. **Server/Client Module**
     1. **Server Class**

****

The server class is an integral part of the client-server design pattern. The server’s job is to listen for connections from clients and handle those incoming requests by creating a thread to process it and passing it to the appropriate handler based on the message type. Our design of the server keeps track of every connected client by mapping their socket to their user ID. Keeping track of this mapping aids the thread runners in functionalities that require broadcasting messages to clients other than the one who sent the original request.

* + 1. **Client Class**

****

The Client class is responsible for managing the interaction between users and WeDiscuss’ core functionalities, which include user authentication, user and communication management, data handling, and user interface integration. The Client class will store maps of users and chatrooms for easy retrieval and display when a client connects to the server. The Client stores a copy of the list of messages directed to the Client as well as the involved chatrooms so they can update both on their end when updates are broadcasted by the server.

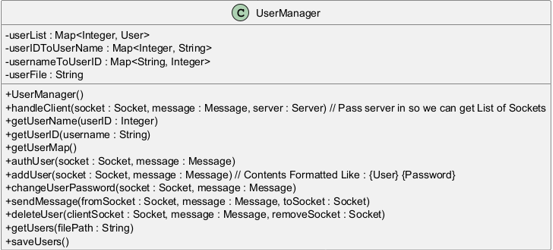
#### User Module

* + 1. **User Class**

****

The User class is an encapsulation of all the user data pertaining to each client of the communications system. Each user object is given a unique user ID for identification purposes, stores the client’s login credentials, tracks the user’s online and admin status, holds a list of all the messages that have been sent to the user so that asynchronous communication is possible, and holds a list of chatroom IDs that the user is involved in. The client side caches a list of involved chatrooms upon login so it is important that a lookup of every involved chatroom for a user is easy to find.

* + 1. **UserManager Class**

****

The UserManager class is responsible for all operations that involve any user objects. These operations include user lookup, user authentication, an addition of a new user, any alteration of user data including the modification of passwords, the deletion of a user, or messages passed from one user to another.

#### Log Module

* + 1. **Message Class**

****

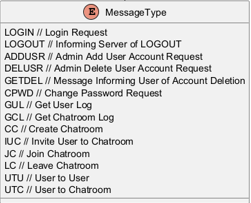
The Message class is the encapsulation of all data that is going to be sent between the client & server, as well as between the internals of the server. Every message is designed to be immutable with the exception of toUserID as it is possible that a message could be sent from a client without that field filled out. Every message is also associated with a messageType to aid the server and client in determining the correct processing mechanism to employ.

* + 1. **MessageCreator Class**

****

The MessageCreator class is responsible for building the attributes that are going to be set inside of a message object. These setters were separated from the Message class to emphasize and ensure the semi-immutability that the Message class is designed to have. To create a message, the attributes of the MessageCreator class are set and then the MessageCreator object is passed into the Message class’s constructor.

* + 1. **MessageType Enum**

****

The MessageType enumeration is useful for helping to identify the type of processing that a transmitted message object should go through when received on both the client and server side.

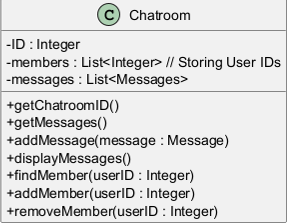
* + 1. **LogManager Class**

****

The LogManager is responsible for anything involving data logs for messages sent by users and within chatrooms. Every message processed on the server-side that involves sending a message is forwarded to the LogManager, which records it in the appropriate log. Other than reading or writing to logs, the LogManager is also responsible for responding to any log requests from an admin on the client side.

#### Chat Module

* + 1. **Chatroom Class**

****

The Chatroom class is the encapsulation of all the data that would pertain to a created chatroom in the communications system. This class holds the list of User IDs that have joined the chatroom and the list of messages that have been sent within the chatroom. Encapsulating this data makes it easier to manage user participation and message history.

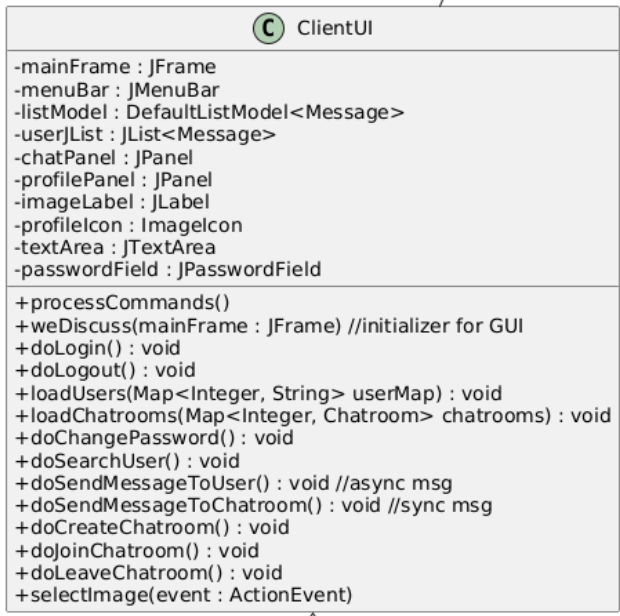
* + 1. **Chatroom Manager Class**

****

The ChatroomManager class is responsible for all operations that involve any chatroom objects. These operations include chatroom lookup, creation, and deletion, as well as adding or removing users from chatrooms. The ChatroomManager is also responsible for the broadcasting of messages to all participants within a chatroom.

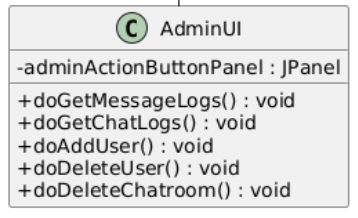
#### GUI Module

* + 1. **ClientUI Class**

****

The ClientUI class is the base front-end user interface for the WeDiscuss application. It will provide functionalities for all users of the application, such as logging in and out, chatroom management, searching for users, and selecting profile pictures. The ClientUI will refresh objects and components throughout the activity of the application.

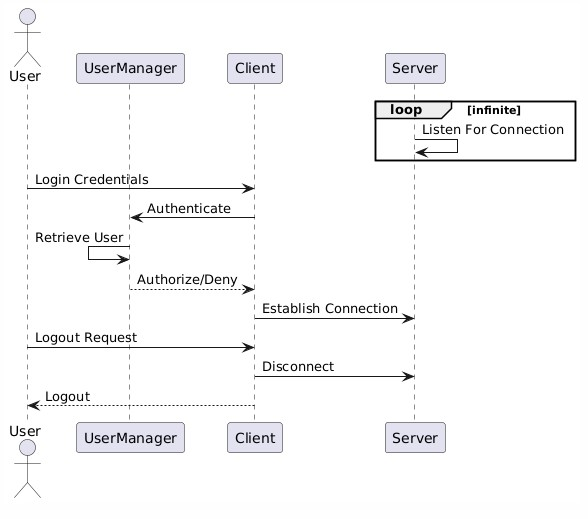
* + 1. **AdminUI Class**

****

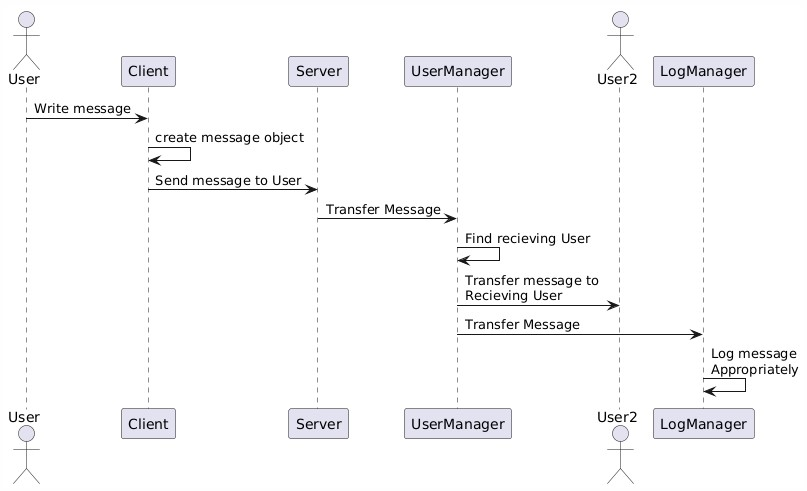
The AdminUI class provides extra functionality to the ClientUI class, given that the user is an Admin. Extra functionalities include viewing message logs and user management.

## 4. UML Sequence Diagrams

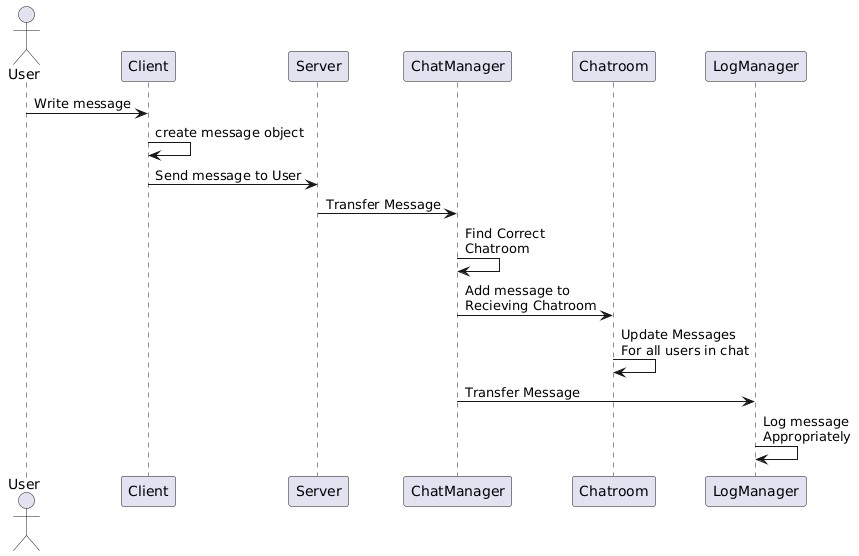
### Login/Logout



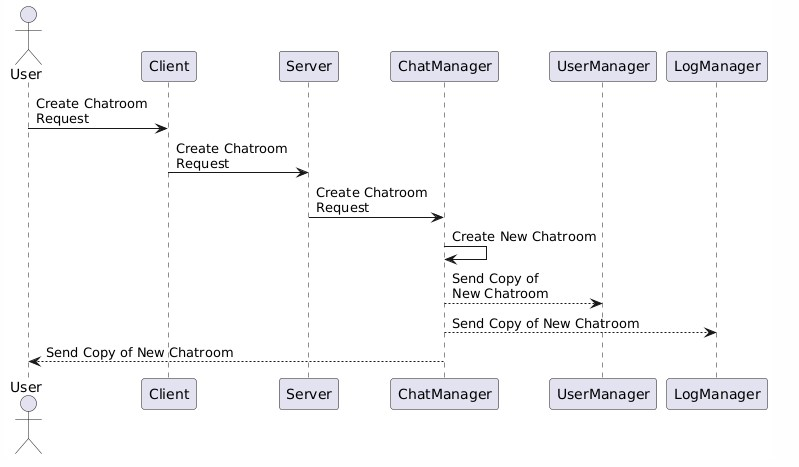
1. **Send Message To User**



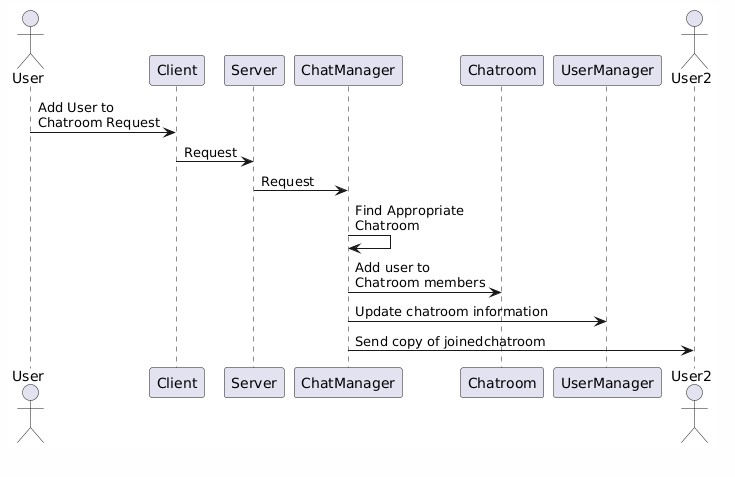
1. **Send Message To Chatroom**



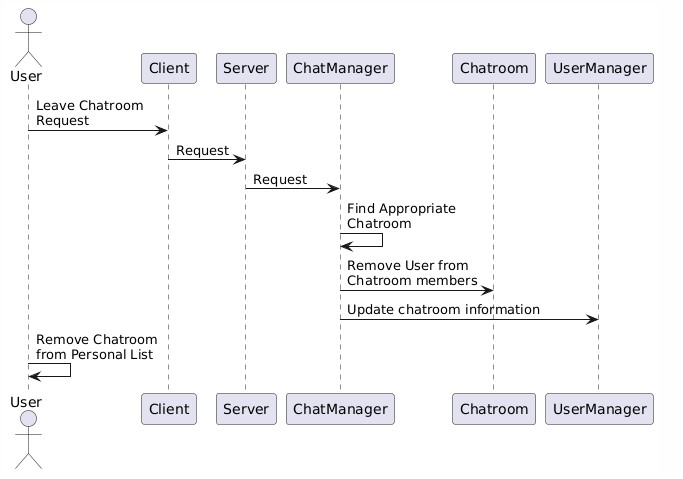
1. **Creating Chatroom**



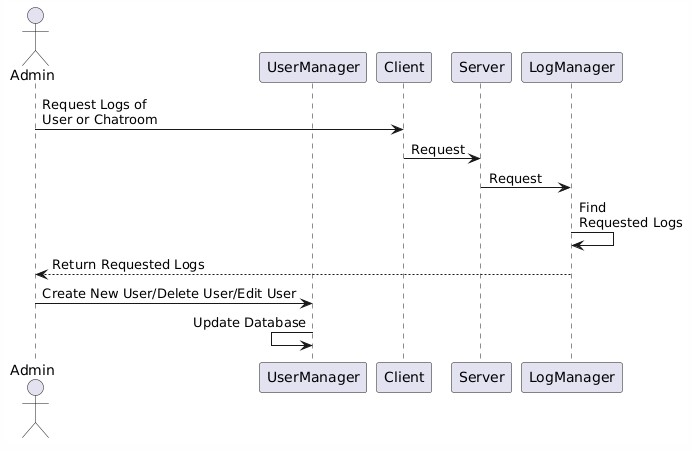
1. **Inviting User To Chatroom**



1. **Leaving Chatroom**



1. **Admin Actions**



### 