**Instant Messenger Application**

*Software Requirements Specifications*

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Revision** | **Description** | **Author** |
| 2/29/24 | 1.0 | Initial version | Caleb Fringer |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. **Purpose**
   1. Scope
   2. Definitions, Acronyms, Abbreviations
   3. References
   4. Overview
2. **Overall Description**
   1. Product Perspective
   2. Product Architecture
   3. Product Functionality/Features
   4. Constraints
   5. Assumptions and Dependencies
3. **Specific Requirements**
   1. Functional Requirements
   2. External Interface Requirements
   3. Internal Interface Requirements
4. **Non-Functional Requirements**
   1. Performance Requirements
   2. Environmental Requirements
   3. Timeline Requirements
5. **Purpose**

This document outlines the requirements for the Instant Messenger Application.

* 1. **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.

* 1. **Definitions, Acronyms, Abbreviations**
* TCP: Transmission Control Protocol, the standard network protocol over which Instant Messaging communications will be transmitted.
* Conversation: a sequence of text messages between two or more users.
  1. **References**
     1. Use Case Document
     2. Class Diagrams Document
     3. User Sequence Diagram
     4. IT User Sequence Diagram
     5. Gantt Chart
     6. Github repository, <https://github.com/Sepulchre49/instant-messenger>
  2. **Overview**

This system provides an instant messenger chat application to organizations. Its goals are persistence of conversations, asynchronous messaging, auditability, and scalability.

1. **Overall Description**
   1. **Product Perspective**

This product will provide large organizations with an internal communications system that prioritizes administrative control over all communications on the network. Organizations are more productive when they have good communication; this system will provide those organizations who adopt it with a means of organizing and retaining many channels of communication. It will also preserve organizational interests in auditing communications occurring in their network by providing administrative users with read access to all conversations on the system.

* 1. **Product Architecture**

This system will operate as a distributed application. There will be a server that handles authentication, logging, and connecting users. There will also exist a client application providing a GUI interface for users to login. They will then be able to send and receive messages through the client application.

* 1. **Product Functionality/Features**

The core feature of this application is its messaging system. The system will support both direct and group messaging between users of the system. Messages will be received synchronously and asynchronously. Furthermore, the system will support logging by writing all conversations to a persistent file on the server. Other features include username/password authentication, a graphical user interface for users, and conversation auditing for ITUsers.

* 1. **Constraints**

All communications must happen over TCP/IP. Network communications must be implemented using Java sockets. Both client and server applications must be written in Java.

* 1. **Assumptions and Dependencies**

It will be assumed that user accounts will be created directly in the code; we will make no attempt at implementing an account creation feature besides direct instantiation of accounts in the server code.

This project will depend on JDK 21 to implement all its features.

1. **Specific Requirements**
   1. **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 with which the user can interact the system.
         3. A user can log in to the instant messenger service by inputting their username and password.
            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 disconnect from the system.
         5. Authenticated users will be able to see all conversations of which they are a participant.
         6. A user can select one of the conversations in their conversations list to view the conversation.
         7. A user can view all messages sent in a conversation since the conversation began.
         8. A user can type and send a new message to all conversation participants.
         9. When a new message is sent in the conversation, all connected users will receive the new message.
         10. 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.
         11. Conversations may be either direct conversations (two participants) or group conversations (many participants).
         12. A user will be able to create new conversations.
         13. When creating a new conversation, the user will be able to select participants from all users on the system.
         14. In addition to their own private messages, authenticated IT users will be able to view 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.
         3. The server will use username and password for authentication.
         4. The server will store the passwords of all users for authentication.
         5. The server will maintain a list of all registered users in the system.
         6. Each user will be classified as either a regular user or an IT user.
         7. The server will keep all currently signed in users as part of its state.
         8. 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.
         9. The server will maintain logs of all conversations in an associated text file.
         10. When a user sends a new message, the server will make the message available to all participants of the conversation.
         11. The server will provide IT users with access to all conversations in the system.
         12. IT users will also be able to act as normal user, sending messages and participating in conversations.
2. **External Interface Requirements**
   1. **Client GUI**
      1. Users will interact with the client via a graphical user interface.
      2. The GUI will have a login screen with text inputs for username and password as well as a button labeled “Log In” to implement requirement 3.1.1.3
      3. The GUI will display an error message on the login screen from requirement 4.1.2 in response to a failed authentication as described in 3.1.1.3.2.
      4. Upon logging in, users will be taken to a home page displaying a list of conversations of which they are a participant.
      5. From the page described in 4.1.4, users will have a button labeled “Create” which will allow them to create a new conversation.
      6. On the page described in 4.1.4 will be a button labeled “Log Out” which will allow users to log out of the system, implementing 3.1.1.4
      7. While creating a new conversation, users will have a context menu allowing them to select one or more participants from the list of users on the system.
      8. From the page described in 4.1.4, users will be able to select one of the conversations in their conversation list, which will take them to a new page where they can view the entire conversation’s history.
      9. From the page described in 4.1.4, users will be able to type a new message into a textbox.
      10. Next to the textbox described in 4.1.8 will be a button that once pressed will send the message.
   2. **Server Interface**
      1. The server will accept connections from any device that requests a connection on the designated port.
3. **Internal Interface**
   1. The server will listen for connections on a specified port that will be agreed upon with the client application.
4. **Non-Functional Requirements**
   1. **Performance Requirements**
      1. The number of simultaneous users supported should only be limited by the hardware running the server.
   2. **Environmental 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 using Java sockets.
      4. Unit testing should be implemented using the JUnit test.
   3. **Timeline Requirements**
      1. A full design should be completed by March 26th, 2024.
      2. The final implementation of the application should be completed by April 30th, 2024.