Local Area Network Office Messaging System

Specifications and Requirements

Ali Cheddadi
Nikhil Kodakandla
Shreejan Paudel

October 16th, 2020

COSC/ITEC 3506

Dr. Amandeep Patti

Fall 2020

Algoma University

Table of Contents

1 Introduction

1.1 Purpose

The purpose of this Specifications and Requirements document is to provide comprehensive information about our LAN chat software.

1.2 Scope

This document features LAN chat software, its requirements, its clients, and our timed deadlines for completion of this project.

1.3 Abbreviations and Acronyms

GUI	Graphic User Interface
LAN	Local Area Network
IP	Internet Protocol
JVM	Java Virtual Machine
OS	Operating System
ROM	Read-only Memory
CLI	Command Line Input
API	Application Programming Interface
LANOMS	Local Area Network Office Messaging System

2 General Descriptions

2.1 Product Functions

The product will store information about office workers, and allow them to communicate as a team. Conversations should happen privately between two or more coworkers, or publicly as a general message destined to the entire office. The software will allow the user to add or remove coworkers from conversations, thus changing their access level to a certain chain of messages.

2.2 Similar System Information

The product will be developed with standard Java libraries, and Modules, such that any device where a JVM is installed would be able to run the application. Portability will be the main characteristic of the software, as well as native OS integration, since modules such as JavaFX abstract GUI logic from the system specific interfaces.

2.3 User Characteristics

Office workers will be provided user accounts from the system administrator, and be required to input their user credentials in order to access certain software features. It will be up to the user to remember their credentials, and in the event that they no longer have access to their account, it will be up to the system administrator to provide them with new user credentials. Once logged in, the user need only be familiar with basic windowing features common to most computers, and would not require any additional training.

2.4 User Problem Statement

Communication software intrudes on office workers' personal time, as they usually find themselves replying to work related messages after work hours. Security also becomes an issue if important company information becomes accessible outside of physical office spaces. The limitless reach of the internet makes it impossible to monitor every access point for potential security breaches, and illegal third party access to private company information.

2.5 User Objectives

Office workers require a communication platform which does not follow them home, quickly and safely gets them in contact with their work peers, and does so within the confines of their office building. The reach of information should be carefully monitored by the system administrator as to limit the spread of discrete information, and any security breach should be manageable as it could only happen through a rogue agent.

2.6 General Constraints

The product should be designed in a way which abstracts all of its functionality through a simple and intuitive interface, which would allow users to access all of its features with little to no training. The application must be portable, as it should be able to run on every computer machine inside of the company. All user and messaging information should be stored within the server package, and would be easily accessible by the system administrator.

3. Functional Requirements

3.2 Database Connection

a) The application always begins with a login page and must make a seamless transition into the main page.
b) Different pages must be defined clearly for the users.
c) Features must be clearly visible to users.
d) Available options on each page should be clearly visible to users.

3.2 Database Connection

a) Database must constantly monitor user activity and inactivity (Offline, Online).
b) Database must be constantly available for all user activity.
c) Database must be updated upon new user registration, password changes.
d) Database must store all existing messages and create backups.

3.3 Messaging

a) Users are able to update their current status (Away, busy etc.)
b) Users should be able to send short messages (Need to decide on Character Limit) to other users in the Database.
c) Message recipients do not have to be online to receive messages.
d) The user may be able to select a group or individual to message to.
e) Incoming Messages are delivered in pop up style. If a pop-up is ignored by the recipient, unread message can be found in users inbox.
f) Incoming and Outgoing messages are automatically sorted into Inbox messages. Users can delete messages individually or in groups.
g) Users are able to restore deleted messages through administrator.
h) Users have an ability to set customized auto responses if they are going to be away for a long period of time.

3.4 Profile:

a) The user profiles are created by the administrator (Based on users First & Last Name)
b) The user shall be able to create a personalised password.
c) The user shall be able to personalize profile if needed (profile picture, status message)

4. Non-Functional Requirements

4.1 Scalability

a) The application is intended to have a limited number of users (10 on average) and all client machines must be connected over a local area network.

4.2 Privacy/Security

a) The application is designed to run on a local network. Privacy/Security is not a concern as any breach of information can be traced back to machines on the local area network (within walking distance.)

4.3 Robustness

a) In case a user's device crashes, a backup of their chat history must be stored on remote database servers to enable recoverability. All user data must be stored on the server machine, and can be retrieved from in the event of a failure on a client’s machine.

4.4 Performance

a) Application must be lightweight and must send and receive messages with absolute minimal delays.

b) Server must be able to serve and communicate with several client machines at once.

4.5 Ease of Use

a) The application is to be designed to be intuitive. No formal training is necessary for operation

4.6 Compatibility

a) The application is designed to run on a JVM. It can be installed on any compatible machine.

5. Interface Requirements

5.1 User Interfaces

5.1.1 GUI

For the client application, the windowing environment and event management system will be handled by JavaFX and as such should match the OS environment in which the software is run. The content itself will be spread throughout several panels, which will all reside within the main window. Panels will allow users to interface with them on any sized display, whether the window is maximized or not, as the graphics will scale accordingly. The user will be required to type out their messages inside of input forms, as well as interact with various buttons and menus with their mouse.

5.1.2 CLI

For the server application, a command line interface will be used by the system administrator to display important server information, and issue various administrative commands. The choice of using command line instructions will allow the administrator to have more control over the server and to perform more complicated tasks (they would have been overly simplified, were they implemented through a graphical user interface.)

5.1.3 API

A set of methods will be created in order to interact with abstract data types and objects within the backend of both the server and the client software.

5.1.4 Diagnostic or ROM

All troubleshooting should be done by the system administrator through the server software, or within the server database.

5.2 Hardware Interfaces

The software runs within an operating system and does not have any direct control over the system hardware. The chosen operating system is left in charge of hardware interfacing.

5.3 Communication Interfaces

Communication protocols go through several layers of abstraction, but will ultimately be handled by the operating system of choice; as such, the software has no direct control over network protocols.

5.4 Software Interfaces

The software packages will allow data to be sent to a main server, from one or more client applications. All interfacing will be done through a GUI for the client application, and a CLI for the server application.

6. Sample scenarios

Billy wants to contact his co-worker Jane:

Billy opens the LANOMS application by double-clicking on a desktop icon; a window appears and prompts him to enter his user credentials. Since this is not Billy's first time using the application, he enters his username in the top text field, and his password in text field directly below it, then clicks on the "Sign In" button. After successfully signing in, the window closes and a new, larger window pops up. The window displays a navigation icons and a list of all conversations Billy has had in the past on the left side, and on the right side, highlighted conversations are shown in greater detail.

Since Billy has never contacted Jane before, he clicks on search. This replaces the list of past conversations with a list of all of his coworkers, and clears out any messages that was displayed to the right. Billy searches Jane, clicks on Janes name and clicks on create new message, which brings him back to the previous screen he was on, and adds a conversation with Jane to his inbox of messages. To bottom right of the window, Billy starts typing his message into a text field, then hits the Enter key to send it to Jane; his message appears on the bottom of the screen. Jane replies to him, his message immediately below his. The more the two coworkers talk, the more the right side of the screen fills, eventually scrolling the conversation down to accommodate new messages.

Kim logged in for the first time

Kim just received her username and temporary password from the system admin. When Kim opens the application for the first time. She is presented with a login screen containing two text input fields labeled “user id” & “password”. Under the password input, there is a “Login button”. The User-ID was given to her by the IT department. When she first logs in, she can put in any password she wants. She creates her new password as she logs in.

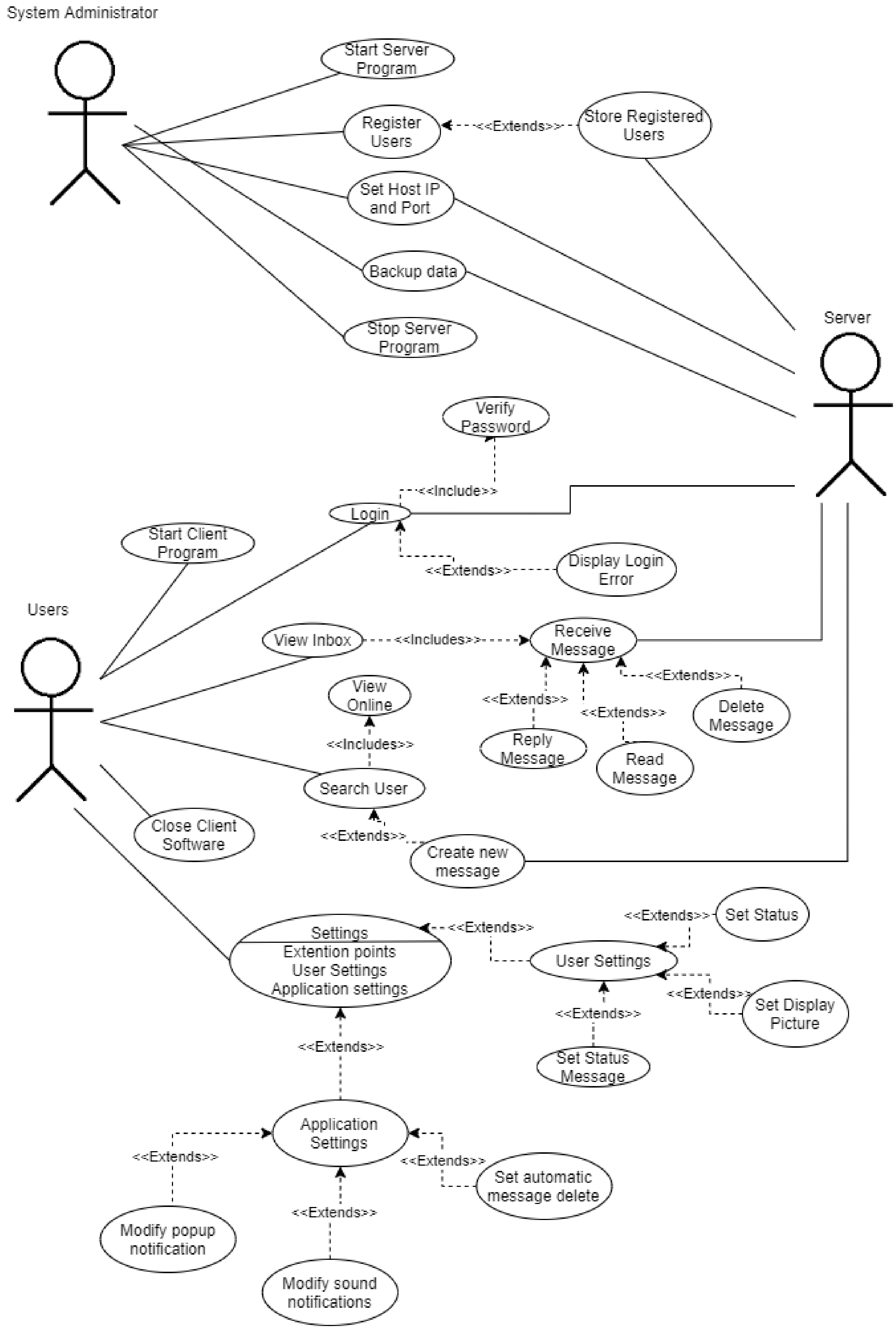
Jane wants to see an offline message she received

Jane double clicks the LANOMS from her desktop. This opens up a login window. She types her username. She types her password. Even though she has logged in many times, the GUI gives an error saying wrong password!! Oops she forgot to capitalize her password again!?! She types her password again. She clicks Log In. This closes the login window and opens a larger window showing various features. A popup appears saying “You received 2 messages while offline”. After few seconds the popup closes by itself. There are two tabs on the left of the window, Inbox and Search, with inbox highlighted. She clicks on the new message she received from Kim. It changes color from yellow to grey, representing that she read the message and it’s no longer new. She replies to the message and logs off for the day.

7. Use case model design table

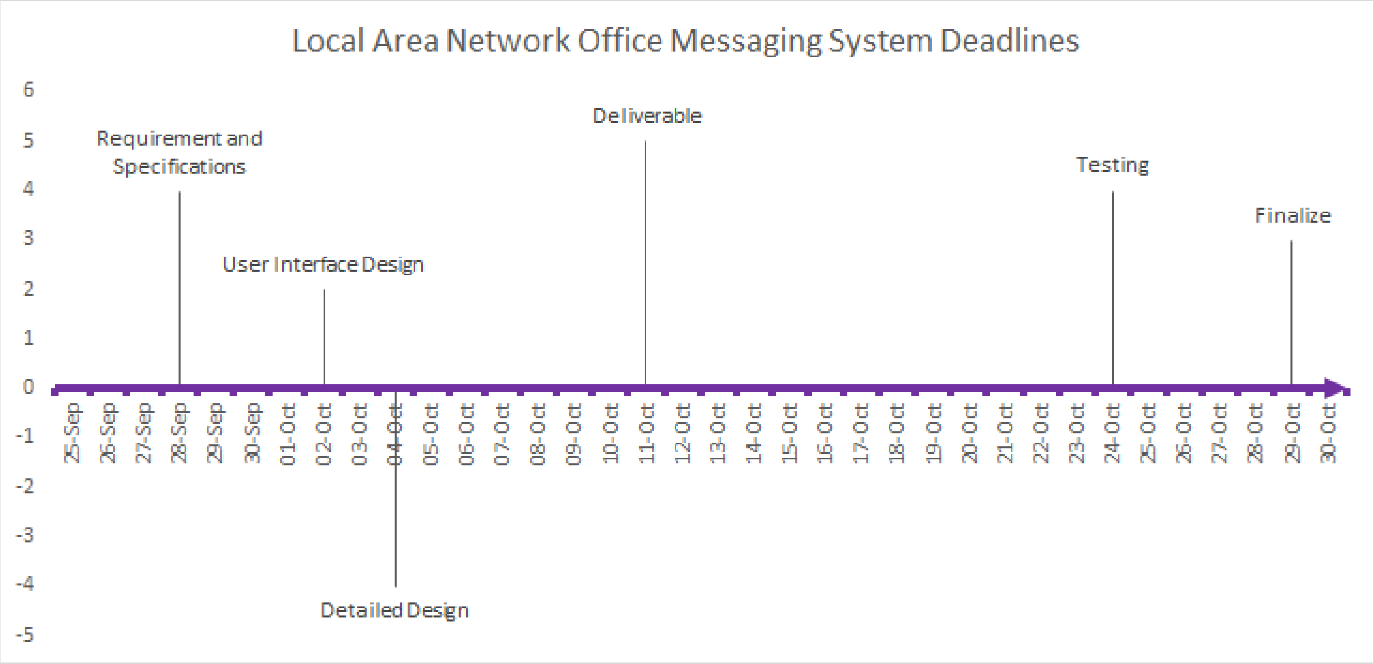
Actors	Goals
System Administrator	Start the server program Register users Set Host IP and Port Backup all existing conversations Stop the server program
User	Start client program Login View Inbox Write messages Read messages Delete messages Add users to messages Search users Create new message View online users Settings User Settings: Change status Change status message Change display picture Application Settings: Modify sound notifications Modify popup notifications Set automatic delete Close client program
Server	Validate Login Forward messages Save and restore data Set server IP and Port

8. Use Case Diagram



9. Schedule

Our goal is to complete the project by October 30, 2020. The due dates for individual portions were calculated based on their percentage marking scheme, with 100% completion being October 30, 2020. Once completed, the product will be showcased to stakeholders and improved further.



The lines represent the start date of mentioned implementation, and completion of the one before it.