David Kim

Java Messenger: A Simple Java Chat App

Requirements Document

Table of Contents

1. Introduction

This document describes in detail the software requirements for Java Messenger, a simple online instant messenger program. This document will describe the problems Java Messenger intends to address and the functional/non-functional requirements of the proposed system. This document is intended for the stakeholders of the application and to assist in the development process of Java Messenger as well as serve as a reference to clarify any future issues the stakeholders may encounter.

1.1 Purpose and Scope

The purpose of this Software Requirements Document is to layout the foundation for Java Messenger, a chat application that consists of a chat server and an indefinite number of chat clients (users). The following developer-oriented requirements describe the system from a software developer's perspective. These requirements include a detailed description of functional, data, performance and other important requirements.

1.2 Target Audience

Java Messenger intends to appeal to all demographic. Social media appeals to anyone that wants to communicate thoughts and ideas within a broader context, and Java Messenger assists in this process.

1.3 Terms and Definitions

Term	Definition
User	Someone who interacts with the messaging service
Stakeholder	Any person who has interaction with the system who is not a developer
Redundancy	The duplication of critical components or functions of a system with the intention of increasing reliability of the system, usually in the form of a backup or fail-safe, or to improve actual system performance
Server	A computer or computer program that manages access to a centralized resource or service in a network
GUI	The graphical user interface is a type of user interface that allows users to interact with electronic devices through graphical icons and visual indicators such as secondary notation, instead of text-based user interfaces, typed command labels or text navigation.

2. Product Overview

The purpose of this section is to give a high level description of the functionality of the project as well as stakeholders and users. This includes four main sections. The first section covers users and different types of stakeholders and their interaction with the system, as well as use cases. The second section provides an overview of the system functionality. The third section provides an overview of the system's non-functional requirements. The fourth section deals with the milestones and release plan throughout the development process.

2.1 Users and Stakeholders

This section lists the users and stakeholders and their needs and expectations for the project.

2.1.1 Developer

The developer's sole responsibility encompasses the entire software development life cycle and it includes development, maintenance, testing and quality assurance.

2.1.2 Users

Users preferences in terms of GUI layout and design are crucial to the app design. The use of beta testers should be employed to ensure maximum user satisfaction upon product delivery.

2.2 Use cases

The purpose of this section is to outline use cases that offer the developer insight into how someone procedurally interacts with Java Messenger.

2.2.1 User wants to send a message

A user who wants to send a message will first select a recipient. The user will then compose a message within the app and press send. The message will be encrypted and sent to the server upon which the message will be available to read by the recipient.

2.2.2 User receives a message

A user who receives a message will be notified by a red dot. Upon receiving the message from the server, the client application will proceed to decrypt the message and it will be displayed for the user to read.

3. Functional Requirements

3.1 User Registration

Users must be able to register for the application using a unique identifier name. Upon installing the application, the user must first be prompted to register their name. If the user refuses, the application should close. If a name is already taken by another user, the user should be given another opportunity to choose a unique name.

3.2 Adding/Deleting New Contacts

The application should allow the user to add and delete new contacts based on a unique identifier name. If the user closes and reopens the application, the user's contact list should be stored in memory.

3.3 Send Message

Users should be able to send instant messages to any contact on his or her contact list. Users should be notified when a message is successfully delivered to the recipient by displaying a green dot next to the message sent.

3.4 Message Status

Users must be able to receive information on whether the message sent has been read by the intended recipient. If the recipient reads the messages, an orange dot should appear next to the message.

4. Nonfunctional Requirements

4.1 Scalability

Java Messenger should be able to provide instant messaging services to +100,000 users at any given time.

4.2 Privacy

Messages shared between users should be encrypted to maintain privacy.

4.3 Redundancy

In case a client's device crashes, a backup of their chat history should be stored on a remote database server to ensure persistence and robustness of data.

4.4 Performance

Java Messenger should be portable, lightweight and must send messages instantly.

5. Milestones and Deliverables

The purpose of this section is to show how requirements are divided into different stages of development and the types of modifications the software is expected to undergo. The requirements are prioritized based on how application features are dependant on certain components of the app.

5.1 GUI for Messaging App

A basic design for the GUI will be developed and made available to beta testers. The beta testers will then provide feedback to the developer so that he or she can make any necessary design changes in order to ensure maximum user satisfaction.

5.2 Database Server

A server implementation that includes SQLite to persist user data and messages will be included in the package.

5.3 User Login

The feature that allows a client to communicate with the server while simultaneously reading messages from the server should be developed and delivered. The user login feature will check each user's account information and password to prevent unauthorized access.