Michael Long

Term Project: Chat Application

Requirements Document

Table of Contents

1: Introduction

Chat Application is going to be a basic program to enable easy communication between two users. Each user will be able to participate in a global chat, as well as communicate personally between two people. This document will outline the functionality of the software as well as some of the details with it's implementation.

1.1: Purpose and Scope

Chat Application is going to be a first massive project in which there will be several new development challenges to tackle and prepare. Chat Application is going to be written in Java with the Java Eclipse IDE. Overall, being able to work on a two month cycle and maintain a workflow similar to that of Industry will be the main focus. Each step of the program will be following the lifeline of a program or implementation.

In terms of true software purpose, Chat Application is going to be designed to carry out basic chatting features you might expect in a chat room, or with text message. Each user will be able to participate in a "global" chat room, in which any individual using the program while being connected to the server will be able to post a message and read other messages posted by anyone else. There will also be the ability to communicate directly with a single user, in which two users can send text based messages to each other. All of these messages will be stored, so that any user logging in can see any previously posted messages either in the global chat or within each of their personal chats.

Due to a two month time constraint, the scope of Chat Application is going to be narrow and focused on implementing key features: a friendly user interface, the ability to connect to a server, the ability to send information into a server while simultaneously receiving information, recognizing if a user is online, and to be able to communicate in the global chat room or direct messages. This program is not meant to have any extravagant features, rather demonstrate the core functionality of Chat Application.

1.2: Target Audience

This document is meant to outline and communicate clearly what features and functionality can be expected from Chat Application. Overall, this document is intended for our stakeholders, to make sure that all of their expectations are met while simultaneously being able to show the capabilities of the program as well as the limitations.

1.3: Terms and Definitions

Chat Application -- The current prototype title for this program.

Message -- A single instance of text which can be sent from a user.

Global Chat Room -- A chat room in which all users connected to the server will be able to write and read messages.

User -- An individual using our software who can read or write to the global chat room and send messages to another user.

Successful Login -- When a user attempts to connect to the global chat room or their own personal account with a username that has a matching password.

Server -- An application which will handle storage and distribution of messages for any user. The Server will also be storing all usernames, passwords, and relevant information to allow the users to communicate.

Online User -- A user that has the Chat Application running on their computer, and have successfully logged in.

Offline User -- A user who is not currently using the Chat Application, but has an existing username and password.

Black Box -- A term used to describe when some input goes in, work is done, then you receive an output. The work done is shielded, as it's implementation is less important than the output when describing functionality with this term.

2: Product Overview

Chat Application, in this form, will be consisting of limited but important features. Chat Application will be prototyping the key features, and will not be adding functionality outside the scope of this project. The purpose of Chat Application is so a user can login, connect, communicate with everyone, communicate with another identifiable user, or read previously written messages. Any other feature will not be implemented due to the stakeholder's requirements as well as the overall success of Chat Application does not require extravagant or unnecessary features. When only the needed information is focused on, it makes the goal more clear, reduces the amount of time in development, and minimizes the potential for bugs which come with increasingly complex features. More information about the functionality of Chat Application will be outlined below in the ***Nonfunctional Requirements*** section.

2.1: Users and Stakeholders

A "user" is an individual or set of individuals who will be directly using Chat Application, they will not be involved in the development or testing of this program. A "stakeholder" is an individual or set of individuals who set the requirements, or have a particular investment or interest in this project, and as such outline the requirements or expectations.

2.1.1: Fei Xie

The main client, Dr. Xie has assigned the creation of Chat Application. He will not be working within development, use, or maintenance. Instead, Dr. Xie has outlined the requirements and specifications that he requires for Chat Application, and will be the primary stakeholder.

2.1.2: Ben Lin

The secondary client, Ben Lin will be in charge of lending assistance to myself or other students with development of their respective programs. For Chat Application, he will be the one to ensure that all requirements set by Dr. Xie are present. In addition, Ben Lin

will also be testing our software to make sure that Chat Application functions to the abilities expected by Dr. Xie.

2.1.3: Michael Long

Michael Long, myself, will be the developer for this project. I will be in charge of making sure that all previous stakeholder's requirements are met. In addition, I will be writing all code used in Chat Application, I will be developing both the Server Application and the Chat Application which the users can connect to the server with. I will be maintaining this software until the retirement phase towards the end of the Spring 2017 term.

2.2: Use cases

The Use Cases will be demonstrating each of the expected uses for Chat Application. Each of the main uses of the program will be outlined here.

2.2.1: User → Server

A user will be a human attempting to connect to the server through a valid username and password. The details of this connection will be outlined in ***Functional Requirements*** under Login.

2.2.2: User ←→ Global Chat Room

A user in the global chat room will be able to read any message posted, and will be able to write any message for every other user to view. The details of the global chat room will be outlined in ***Functional Requirements*** under Global Chat Messaging.

2.2.3: User ←→ User

Any user will be able to directly message another user. These messages will be private between the two users. The details of the personal messaging between two users will be outlined in ***Functional Requirements*** under User to User Messaging.

3: Functional Requirements

Functional Requirements are the essential features that allow Chat Application to be used through it's basic functionality. The main functions that are needed for Chat Application to be functional are: **Login**, the ability for a user to access the server, **Global Chat Messaging**, the ability to send and read messages in the global chat, and **User to User Messaging**, the ability to send messages between any two users.

3.1: Login

This feature is meant to be used with a human user. A new user will be able to login to the server. If the user is not registered, or would like a new account, they will have the ability to create their own username and password. This information will be stored within the server. When a client attempts to connect to the global chat room or their own personal profile, they will first have to complete a successful login in which their username matches a password in the server. Each username will need to be unique.

3.1.1: Username

Usernames will be a set of characters that will serve as the identifier for a login. It will also double as a visible component to a user while sending or receiving messages.

3.1.2: Password

A password will be a set of characters unique to a particular username, this password will be determined by the user when they create their account.

3.2: Global Chat Messaging

Each user is expected to be able to communicate within the global chat room. Any user that has successfully logged in will be able to write a message, read a message, or view previously written messages. This functionality will be expected to work whether a user is online or offline, as long as the server is running, any user can view all messages within the global chat room.

3.2.1: Reading a Message in Global Chat Messaging

When a user logins in to Chat Application, they will be immediately greeted with the global chat. All messages written into the global chat will be stored by the server and

viewable through a scrollable textbox to all users.

3.2.2: Writing a Message in Global Chat Messaging

When a user attempts to write to the global chat, they will be able to type any message they would like into a textbox. From there, that message will be sent to the server, and written to all users running Chat Application as well as to the log of the global chat so that any user that is offline will be able to view it later when they login.

3.3: User to User Messaging

An online user will be able to send any message to another online user, or to an offline user. Any message sent will be stored on the server so that the two user's involved will both be able to view messages they received while they were online, as well as when they were offline. These messages, like the global chat, will be entered through a chat box and viewed in the same scrollable window. The difference being that only the user sending, and the user receiving the message will be able to view each message sent or received.

4: Nonfunctional Requirements

Nonfunctional Requirements outline the features of a program that do more with the functionality and operation of the system as a whole. The Nonfunctional Requirements are: **Online Connectivity**, the ability for Chat Application run by the user to communicate with the server, **Server Application**, a separate application that runs and handles connecting users, **User Friendliness**, Chat Application must have a simple to use interface, and **Data Storage**, the ability to view all messages stored previously.

4.1: Online Connectivity

For a user on Chat Application to communicate with any other user, they must be able to have an internet connection to communicate between the two instances of Chat Application. Online Connectivity will allow each instance of Chat Application to communicate with a Server Application to send or receive messages, despite when either user is online or offline.

4.2: Server Application

The Server Application is an abstraction of the Chat Application so users are not hosting and sending messages on their own system. This piece of software is going to be written as it's own separate entity. The Server Application will be responsible for receiving a message, displaying it to the corresponding user, storing the information in a log to allow future access for either the sending or receiving user, then updating the user's application who received the message. This piece of software will not be accessible to the user or through Chat Application.

4.3: User Friendly

Any user on Chat Application will be able to easily use the software. For the purposes of prototyping and due to the scope of this project, the graphical interface will be minimal, but very clear and easy to identify. Additional visual components will be outlined in a separate Design Document.

4.4: Data Storage

Chat Application is expected to support being able to view all sent and received messages

for any particular time, whether it is in the global chat, or between two users. The Server Application will be responsible for writing all sent and received messages to personalized logs, to make sure that every user has access to the appropriate information. This will be done through unencrypted text files, as any more complex storage method would be outside of the scope of this project, and unnecessary.

5: Milestones and Deliverables

Milestones are objectives in which something is accomplished and steps towards finishing the current phase of the software. A Deliverable is product that is ready to be used or tested.

5.1: Constructing Chat Application

Chat Application is the main piece of software. It has several milestones which can be allocated to make sure that transitioning and tracking progress is smooth. We will not be focusing on Online Connectivity, as it will be black boxed in a separate group of functions, instead the focus will be on a graphical user interface and functionality.

5.1.1: Milestone: Establish Data Structures / Deliverable: Data Storage

Chat Application is required to allow instant access to previously viewed messages. Data Structures allow the application to allocate and store data in such a way where the program is fast and memory efficient. This will be broken up into multiple stages: Establish storage for each instance of a chat (global, between two users), and the ability to view all online users. The resulting deliverable is the ability to store any number of messages and user data.

5.1.2: Establish Data Structures: Chat

Any instance of messages are going to be loaded initially from Server Application. Given that there will be a lot of messages, the load time and access needs to be quick and efficient. A stack implementation will most likely be used, given that all data is put ontop of eachother, and once the message is written into the window, it will no longer be modified or changed. Stacks are very quick to load into memory from a text file, and as such would be ideal for Chat Application.

5.1.3: Establish Data Structures: Online Users

A user is free to go online or offline at their leisure. As such, the data structure which shows which user is online or offline needs to grow and shrink to fit this need of a constantly changing flow of users. A Linear Linked List implementation will be ideal, as it can easily grow and shrink. Although it is not the ultimate data structure for sorting or

massive traversals, it allows for an easy, bug free implementation that stays within the scope of Chat Application. This can later be expanded into a Binary Search Tree to allow for easy sorting and fast access, although these implementations are more prone to bugs, and are slower when it comes to marking a user as offline.