A Messaging App

Requirements Document

Table of Contents

1 Introduction

The project that this document discusses is a project to create a messaging app. This document is the requirements document which will go over the requirements for the project. The document will discuss meta aspects of the project at a high level while also elaborating on necessary details; please see the table of contents for a full illustration of topics.

1.1 Purpose and Scope

The purpose of this document is to outline the requirements for the messaging app, the technology stack it will be using, and technicalities to be taken under consideration in its design. However, the document will not go into much detail about the design of the app, or the actual implementation of features; it is a schematic, not a blueprint.

1.2 Target Audience

The target audience of this app will be people who prefer a very minimalist approach to messaging.

1.3 Terms and Definitions

- Bus Factor: The higher the bus factor, the more probable it is that a bus running over the lead developer of a project will devastate the progress of the project and cripple its ability to be completed. Often used as a slightly morbid way of determining how robust a project's state of documentation and organization is. That is, it should not be dependent on the people as that indicates a lack of necessary documentation and paperwork.
- Version Control: Version Control (VCS) is a system that stores and logs code for organization and retrieval.
- Commits: A way of adding in and storing changes to a codebase over time into a VCS database.
- Client: A client is a single instance of the app that a user will use to log into the server.
- Server: A server is a single instance of a server app that will be used as the main interface between all clients.
- Client-Server Architecture: This is a style of designing interaction between multiple clients where many clients talk to one server.
- I/O: Input/Output is all of the input and output into and from a system.
- Asynchronous: The concept that something can happen in the background without the system paying explicit attention to it.
- Non-blocking: An activity has the ability to perform in the background and have other tasks start and finish while it runs.

2 Product Overview

The project here is a messaging app that acts, in short, as a stripped down IRC client. Users log into the app and chat either globally or one-on-one or in group chats. The project itself will involve a client-server architecture model with a many clients to one server relationship. The UI will be graphical and user-friendly in nature. The program will not deal with extensive user customization, nor arbitrary execution of user-designed scripting, macros, or aliasing of commands.

Definitions:

- User: A person who has an account in the app, logs in, and uses the app.
- Account: An object which holds User data and information.
- Stake Holder: An entity or entities that have a vested interest in the result of the project.
- Use Case: An intended and planned for path of usage for the app, including User demographics.

2.1 Users and Stakeholders

This section goes over the stakeholders and the users of the application. It details who will use the app, and who has a vested interest in the result of the project.

2.1.1 User: Professor/TA

The professor and, primarily, the TA will be the main users of the app as it's not intended for actual use outside of the class. That being said, were this a "real" project, the user group would additionally include the targeted demographics for the application. As messaging apps are a very saturated market, a niche is required to make an entryway into the market and to differentiate from the competition. In this case, the niche would be people who want a very simple and feature-lite experience.

2.1.2 Stakeholder: Me

The first stakeholder is me, as I have a vested interest in seeing this application succeed. It is directly tied to my final grade, so I have a strong need to do well with it. My role in development, deployment, maintenance, etc., is being the architect, designer, and creator.

2.1.3 Stakeholder: Professor/TA

The professor and TA are also stakeholders in the app as they have a vested interest in seeing it succeed. Their knowledge of how well they've succeeded in teaching concepts rests, in part, on seeing how well I deliver. They will use the application and will provide some guidance but are otherwise not involved in its creation.

2.2 Use Cases

In all use cases, their experience should be seamless and easy; they will only need a limited subset of the features, so their UI should not be overly cluttered nor should it be complicated to use rarely needed features.

2.2.1 Tester

The tester, either by manual testing or using automated tests, will be using this app in an attempt to break it; by doing so, they will hope to ensure as much as possible that the app is bug free and is well defined for all inputs.

2.2.2 One-on-One User

The One-on-one user is a subset of the users who are primarily interested in engaging in one-on-one chats with other users.

2.2.3 Group Chat

The group chat user is a subset of the users who are primarily interested in engaging in group chats with groups of users.

2.2.4 Global Chat

The global chat user is a subset of the users who are primarily interested in engaging in global chatting, irc-like.

3 Functional Requirements

A functional requirement is a requirement that is absolutely essential for the application to be considered working.

3.1 Functional Requirement 1: UI

There must be a UI (User Interface). This UI will be user-friendly in all aspects including, but not limited to, accessibility, input agnostic, window size/shape agnostic, intuitive, easy to use, easy to read, and unobtrusive.

3.2 Functional Requirement 2: Client-Server Architecture

The design of the app necessitates a client-server architecture so that multiple clients can connect to one server in order to send messages back and forth.

3.2.1 Functional Requirement 2 Aspect 1

The client itself will require authentication by a user before being able to join the server. This will necessitate account creation and storage.

3.3 Functional Requirement 3: Chatting

The experience will be such that the user of the application will be able to chat globally, in groups, or one-on-one.

3.3.1 Functional Requirement 3 Aspect 1

Further, all of this chatting will be stored to disk in an appropriately safe way so that the user can seamlessly look it up again.

4 Non-functional Requirements

A non-functional requirement is a requirement that, while not absolutely essential for the application, needs to be done before the application is considered finished and maintainable.

4.1 Non-Functional Requirement 1: VCS

Version control should have a sane history, with clear commits and a coherent story being told through commit messages.

4.2 Non-Functional Requirement 2: Documentation

Should have comprehensive documentation. The documentation should be clear enough in all aspects of the program that the bus factor of the project approaches zero.

4.3 Non-Functional Requirement 3

The I/O of the entire system, client and server, must be asynchronous and non-blocking. This will allow for messages to be sent simultaneously without error and will allow for concurrent chatting being done by different users on the system.

5 Milestones and Deliverables

This section goes over what I will deliver and the checkpoints on which certain stages of the project will be done at.

5.1 Milestone/Deliverable 1: Requirement Document

This milestone is the submission of this particular document. The deliverable is this document and nothing else at this point in time. It is due on April 25.

5.2 Milestone/Deliverable 2: Design Document

This is the design document. The design document is a detailed document going over, in greater depth, the exact design and specifications for how the project will be put together and organized. It is due May 9.

5.3 Milestone/Deliverable 3: Test Plan

The test plan is a comprehensive document detailing all of the test suites, the test cases, testing methodology, and all other factors related to testing the program and ensuring its correctness. It is due May 30.

5.4 Milestone/Deliverable 4: Project Report & Final Deliverables

This will have multiple things in its Deliverables. Firstly, there is the project report, which is the main comprehensive report detailing all of the project's documentation, the progress report, a "journal" for the duration of the project, and so on. Secondly, there is the project itself which will be submitted at this time. This will be the complete project, ready to be deployed. It is due June 8.