Bradley Maness

**Java Chat Application**

**Requirements Document**

**Table of Contents:**

**1. Introduction 2**

1.1 Purpose and Scope 2

1.2 Target Audience2

1.3 Terms and Definitions 2

**2. Product Overview** **3**

2.1 Users and Stakeholders 3

2.1.1 Instructor and Assistant 3

2.1.2 Students 3

2.2 Use Cases 3

2.2.1 Use as a Basic Messaging Tool 3

2.2.2 Use as an Evaluation of Skills 3

**3. Functional Requirements 4**

3.1 Client 4

3.1.1 Registration and Login 4

3.1.2 One-to-One Communication 4

3.1.3 Multi-User Communication 4

3.1.4 Access to Personal Message Records 4

3.1.5 Inform of All Active Users 4

3.2 Server 4

3.2.1 Accept and Indefinite Number of Clients Connections 5

3.2.2 Deliver Messages Between Clients 5

3.2.3 Store User Information 5

3.2.4 Store Message Records 5

3.2.5 Inform of Other Client Connections 5

**4. Nonfunctional Requirements 6**

4.1 Use of Java Programming Language 6

4.2 Creation of User-Friendly Interface 6

**5. Milestones and Deliverables 7**

5.1 Server Implementation 7

5.1.1 User Database 7

5.1.2 Message Database 7

5.2 Single Client Implementation 7

5.2.1 Communication Between Client & Server’s User Database 7

5.2.2 Communication Between Client & Server’s Message Database 7

5.3 Multi-Client Implementation 8

5.3.1 Client Connection Alerts 8

5.3.2 Communication Between Different Clients 8

5.4 Implementation of User-Friendly Interface 8

**1 Introduction**

**1.1 Purpose and Scope**

This document discusses the requirements for a traditional computer messaging application assignment written in Java for the Portland State University course “CS300: Elements of Software Engineering” Section 003. This document examines the projects English-written premise and extracts from it a list of detailed requirements that must be implemented in order to produce a product that fulfills the needs and desires of those who presented it. A general overview of the product as well as the requirements will be presented in future sections.

Source of Premise:

http://web.cecs.pdx.edu/~linbin/cs300Spring2017/Project/ChatApp.html

**1.2 Target Audience**

The main audience for this document is the instructor of the course and their assistants who presented this project as an assignment to showcase student’s abilities to engineer software from a written premise. In more practical terms this project is for those interested in simple application of advanced Java programming concepts such as networking and graphic user interfacing.

**1.3 Terms and Definitions**

To clarify the rest of the document this section will contain definitions of terms that will be featured within:

Product: the finished project.

User: the person who uses the product.

Client: program used by primarily by the user.

Server: program that handles actions sent by the server.

**2 Product Overview**

**2.1 Users and Stakeholders**

This section will briefly explain the users and stakeholders of this product or those for whom this product is meant for. The most important entities are the Instructor & Assistants(stakeholders) and students(users).

**2.1.1 Instructor & Assistants**

The instructor is the person who assigned the project and as such has specific expectations about its functionality

and implementation. The instructor’s assistants will

examine the finished product in order to make sure it is up to the instructor’s standards.

**2.1.2 Students**

After the final product is delivered, inspected, and graded by the instructor and assistants the next entity this product is meant for are other students for either study or modification.

**2.2 Use Cases**

There are two primary use cases for this product: use as a

basic messaging program and as an evaluation of personal

programming and software engineering skills.

**2.2.1 Use as a Basic Messaging Program**

The main use of this product is as a basic tool for sending

message between people. Users should be able to use the use

the product intuitively in order to interact with others

using the same program.

**2.2.2 Use as an Evaluation of Skills**

The final use of this product is for the evaluation of personal skills by the instructor and assistants. They must be able to use the product like a regular user and then

compare the functionality against a grading metric.

**3 Functional Requirements**

This section will list fundamental requirements of the product; aspects that contribute to the underlying functionality.

The underlying model for this product is the Client-Server model. The following subsections will detail fundamental requirements for each respectively.

**3.1 Client**

**3.1.1 Registration and Login**

A new user must create a registration before being able to

use the product. The registration basics should include a

username and password. Afterwards the new user should be

able to use this information to log in and formally use the product.

**3.1.2 One-to-One Communication**

Two users using separate client programs must be able to

send messages between each other.

**3.1.3 Multi-User Communication**

Multiple users (3 or more) must be able to create and send

messages in which only users included within the

conversation can view them.

**3.1.4 Access to Personal Message Records**

Any user must be able to look up records of conversations

they have participated in. Users must not have access to

messages they have not participated in.

**3.1.5 Inform of All Other Active Clients**

Anyone using the client should be able to see all other

clients currently online using the same program. Users must

also be alerted when another client program connects to the server.

**3.2 Server**

**3.2.1 Accept an Indefinite Number of Client Connections**

The server program must be able to scale itself to accept

connections from possibly an infinite amount of clients.

**3.2.2 Deliver Messages Between Different Clients**

The server must be able to process messages between clients

for both one-on-on and multiple participants.

**3.2.3 Store User Information**

The server must be able to keep records of user information

for the purposes of logging on and directing messages.

**3.2.4 Store Message Records**

The server must store records of all messages sent across

it for the purpose of clients being able to view personal

chat records.

**3.2.5 Inform of Other Client Connections**

Whenever a client connects to or disconnects from the

server, the server must inform all other client programs

of this action. The server must also display all clients

that are currently connected.

**4 Nonfunctional Requirements**

This section will describe all the product nonfunctional requirements; aspects that do not directly contribute to the underlying workings of the product, but which are still requested as part of the product. Unlike Section 3, this section will not be divided between client and server as the following requirements are applied to both aspects of the product.

**4.1 Use of Java Programming Language**

The programming language selected for this product is Java. The

use of Java will ensure portability of the product between

different clients on different types of machine setups.

**4.2 Creation of User-Friendly Interface**

The client and server aspects of the product must have a user-

friendly interface. A new customer unfamiliar with the product should be able to set up and interact with both aspects of it with relative ease.

**5 Milestones and Deliverables**

This section will describe a basic work flow of the product.

A summary of each part of the product will be described in each

subsection in order of primacy of implementation(what should be

created first).

**5.1 Server Implementation**

The server, being the connection point between multiple clients,

will be the first aspect of this program to be constructed. Work

on this section will contain the following:

**5.1.1 User Database**

The server program is what will keep records for both user

data and message history. Before the client program can be

made storage for user data must be implemented before

construction of the client program.

**5.1.2 Message Database**

The server will also keep records of messages sent over it.

An implementation of a simple message database will be

implemented.

**5.2 Single Client Implementation**

After implementing the user and message databases in the server program, A single, simple client program will be constructed to test basic functionality.

**5.2.1 Communication Between Client & Server’s User Database**

At this stage, the basic client program must be able to

send requests to the server and the server must be able to

return information from it’s database to the client. This

will become the basis for the products login functionality.

**5.2.2 Communication Between Client & Server’s Message**

**Database**

In this next stage the client program will send data to the

server program to test its message database. Because there

are no other clients to deliver messages to at this time

this part is to test the server’s ability to receive and store records of messages being sent across it. The server

must also be able to send records of messages back to the client program.

**5.3 Multi-Client Implementation**

After successful testing of server interaction with a single

client testing with multiple clients can begin. This stage will

test messaging between different clients as well as other

requirements involving multiple client programs.

**5.3.1 Client Connection Alerts**

Implementation of client connection alerts happens here. The server will alert all other clients of logins and logouts as well as provide a list to each client program of all other client programs currently online.

**5.3.2 Communication Between Different Clients**

This stage will test the ability of the server to deliver

messages between client programs for both one-to-one and

multi-user communications. It will also continue to test

record keeping for messages.

**5.4 Implementation of User-Friendly Interface**

After the products basic functionality is completed work on

creating the user interface can begin. This process involves

translating both the client and server programs from a comm and

line interface into a graphic user interface.