Team 15

Report by: Jeremy Polansky

Problem Statement

This program’s purpose is to develop a system in which a server is used to allow clients to relate by sending messages and associating themselves with certain other clients.

Approach to Solution

This program was written using C++, and then later would be executed in a Linux environment. There is code written for the server side and for the client side, of which the client code will be used for any client that would talk with the server. From the client side, it will talk with the server and send over authenticity information (user name and password) and send commands to do certain things, including sending messages to other users. On the client side, it will, from startup, take in credentials from the client to “login” and authenticate a certain user, then take in commands to run on behalf of that certain user, such as sending messages to certain users on behalf of the user sending the message. It will save information about how the users are related so that it will be able to relay the information later as needed. Because the server and client sides are not able to communicate directly but only can send binary messages between them, a specific order of communication is used so that each set of code (for server side or client side) is pre-programmed to follow suit with what the other is asking/giving across the connection. In other words, from the main menu on the client side, the client must send information to the server depending on what the user selects because the client is able to send information to the server or else the server would not know what that information applies to (if it’s a username for login credentials or to request to dictate to whom to send a message). The tool used was g++ version 5.2.1 for Linux and the code was written in Visual Studio 2013.

Solution Description

To build the solution, simply execute

“make”

To remove the generated .o object files after compilation, simply execute

“make clean”

To remove all files generated after compilation, and simply execute

“make cleanall”

To run the build program execute

“./Program5.out”

This program meets all course learning outcomes. Here is an explanation of how each are met in our program:

1.Ability to create classes of abstract data consisting of variables and functions

Trivial; Used on client and server side

2.Ability to utilize C++ constructors, copy constructors, and destructors

Trvial: Used most places

3.Ability to utilize C++ OOP features using static member data and member functions

Used when…???

4.Ability to utilize C++ File and Stream Input/Output Processes

Used when prompting user for input and receiving input from user

5.Ability to generate reusable code using inheritance

Used when…???

6.Ability to use polymorphism and virtual member functions

Used when…???

7.Ability to generate reusable code using templates

Used when….???

8.Ability to create and utilize dynamic data structures such as linked lists

Used when dealing with the information on the server

9.Ability to create and utilize recursive functions

Used in some method…???