**CSCI 3010**

**Spring 2023**

**Chat Application**

In this project, you will develop a chat application that enables text messaging between a server and a client. The application should be written in Python using the socket module.

**Server:**

The server should ask the user for an IP address (or to press enter for localhost) and should ask the user for a port number (or to press enter for default port). The server should also ask the user for their username. Display the server and client usernames next to their respective messages in the conversation. Ensure that both server and client can view the entire conversation.

After establishing the necessary connections, the server waits for an incoming connection and message from a client.

When a message is received, the message is printed to the screen and the user should be prompted to send a message back to the client. If an exit keyword, such as “end” is entered on either side of the application, the program should then exit.

**Client:**

The client should establish a connection with the server using the IP address and port number entered on the server side. The user should be prompted to enter this information for connection with the server.

The user should then be prompted to enter a message to send to the server and wait for a response, or the program exits depending on the input given.

If a message is received from the server, this message is printed to the screen and the user on the client side is prompted to send a message back. This continues until a user on either side enters the exit keyword of your choice.

**Error Handling:**

Your program (server and client) should handle various errors that may occur such as errors encountered during the bind, connect, accept, send, or recv operations. Also consider cases such as when an invalid IP address, port number, or certain keypresses are entered, such as an empty string is returned. You are given creative freedom on how you would like your program to handle these errors, i.e. exiting the program or prompting the user for new input, etc.

**What to turn in:**

ReadMe

Server script

Client script

**Rubric:**

1. Functionality 60%
   1. Server prompts for IP address, port number, and username
   2. Client prompts for server’s IP address, port number, and username
   3. Server and client can send and receive messages
   4. Program exits when either side enters the exit keyword
   5. Entire conversation is visible to both server and client
2. Error Handling 20%
   1. Program handles errors in bind, connect, accept, send, or recv operations
   2. Program handles invalid IP address, port number, or empty string input
3. Code Quality 20%
   1. Code is well-structured and easy to understand
   2. Proper use of functions and modules
   3. Comments and documentation