PSUEDOCODE

## Server

# Initialize GUI

Set up GUI using QT Framework

# Create Socket

Create a stream socket with AF\_INET, SOCK\_STREAM, O parameters

Check for any errors on socket call

Go to **Bind Address** State

# Bind address

Bind an address to the socket

Allocate memory for server struct

Initialize server struct with AF\_INET, port specified by user, and to accept connections from any client

Call bind on ListenSocket

Check for any errors on bind call

Go to **Listen On Socket** State

# Listen on socket

Listen for connections, queue up to LISTENQ connect requests

Call listen on ListenSocket with LISTENQ

Check if listen call failed

While true

Call select()

Check if there was a new client connection

Go to **Accept New Connection** State

# Accept new connection

Call accept(ListenSocket)

Check if it accept call failed

Save client’s descriptor

Go to **Update List of Connected** Clients State

Go to **Check clients for data** State

# Check clients for data

loop through all the clients

Check if client has data

Go to **Read Data** State

write the data read from the client socket to all other sockets except the one that sent it

# Read data

While (read data from the client socket)

Check if read call failed

Update bytes read

Go to **Echo Data to all other clients** State

# Echo data to all other clients

Write data read from the client socket to all other sockets except the one that sent it

If no more readable descriptors

Go to **Listen on Socket** State

## Client

# Initialize GUI

Set up GUI using QT Framework

Go to

Go to **Wait for IP and Host Info** State

# Wait for IP and Host Info

Forever loop:  
 If invalid IP or port entered

Print error message

Else if no port entered

Use default port

Else

Go to **Create Socket** state

# Create Socket

Create a stream socket with AF\_INET, SOCK\_STREAM, O parameters

Check for any errors on socket call

Go to **Bind Address** state

# Bind Address

Bind address to the socket

Allocate memory for server struct

Initialize server struct with AF\_INET, port specified by user, and to accept connections from any client

Call bind(ListenSocket)

Check for any errors on bind call

Go to **Connect to Server** state

# Connect to Server

Call connect()  
If error  
 Print message

Go to **Create** **Data Receiving Process**

# Create Data Receiving process

Fork new process

If error

Print error message

Go to **Wait for User Input**

# Wait for User Input

Forever loop  
 If received keyboard input  
 Print to window  
 If received <Enter>  
 Go to **Transmit Data through Socket** state

If “Save chat session” toggled

Open file for writing  
 Go to **File Writing Process** state

Else if untoggled

Close file

Close process

# Transmit Data through Socket

Get text from user input  
Add to buffer  
Write buffer to socket  
Go to **Wait for User Input** state

# Wait for Incoming Server Data

Call select()

Forever loop

If received data

Store data in buffer

Go to **Print to Window** state

If “Save chat session toggled”

Go to **Write to File** state

# Print to Window

Get text in buffer  
Print text to screen  
Go to **Wait for Incoming Server Data** state

# Write to File

Get text in buffer  
Write text to opened file  
Go to **Wait for Incoming Server Data** state