DEPARTMENT OF COMPUTER SCIENCE

FORMAN CHRISTIAN COLLEGE

(A CHARTERED UNIVERSITY),

LAHORE

Sharing is Caring

A P2P Chat and File Sharing Application using TCP

Instructions:

- Do not zip the files. Just submit three files; namely *TCPAssignment.doc*, *sc_client.py* and *sc_server.py*
- Assignment can be done in groups; only 2 students per group are allowed.
- Plagiarism will result zero marks in the assignment.
- Submit your work before the deadline.

Overview

A peer-to-peer (P2P) chat and file sharing server is a type of network architecture in which individual computers, or "peers," can connect to each other and share information without the need for a central server. In a P2P system, each computer acts as both a client and a server, allowing users to communicate and share files directly with one another without going through a central server.

In order to use a peer-to-peer chat and file sharing system, users must first connect to the network and establish a direct, reliable and stable connection with one another. Once connected, users can begin sending and receiving messages and files directly to and from one another.

You are required to create a chat and file sharing application with graphical user interface in python which enables clients to chat and share files between each other. As name of the assignment suggests, you will build two features. One is sharing which refers to file sharing facility among registered clients (peers) and other is caring where multiple users can also chat with one another. The client makes a direct connection to the opposite client using the information provided by the server, and the server maintains a log of chat sessions.

Assignment Walkthrough

A peer-to-peer chat and file sharing service is built where multiple clients will be able to log into a central server, and subsequently share files with one another directly. The chat system consists of a central server that manages multiple chat sessions, and users can participate as chat clients. Clients connect to the central server on its universally-known port. Upon connection, the server asks each client if it is a new client or a returning one. In case it is a

new client, it is asked to register its username and password with the central server. If it is a returning client, the server authenticates them using the existing username/password combination.

Once this initial process is done, clients request for a list of clients that are currently connected to the server. Clients are then given the listening socket of the client they want to connect to, and it then attempts to make a direct connection to the opposite client using the information from the server. The server keeps a track of all the active chat sessions. Through this direct connection, clients share messages and files with each other. After communication is complete, the clients indicate to the server that the session is finished, and the server removes that session from its database.

We need to do more than one job at the same time. Therefore we are implementing multi-thread system. Use of an appropriate database is highly recommended for storing username, password and chat logs.

```
#========================
# Name:
# Roll no:
# Section:
# Date:
#========================
#sc_server.py
#---------------------------------------
# Paste your solution below:


#========================
#sc_client.py
#---------------------------------------
# Paste your solution below:
```