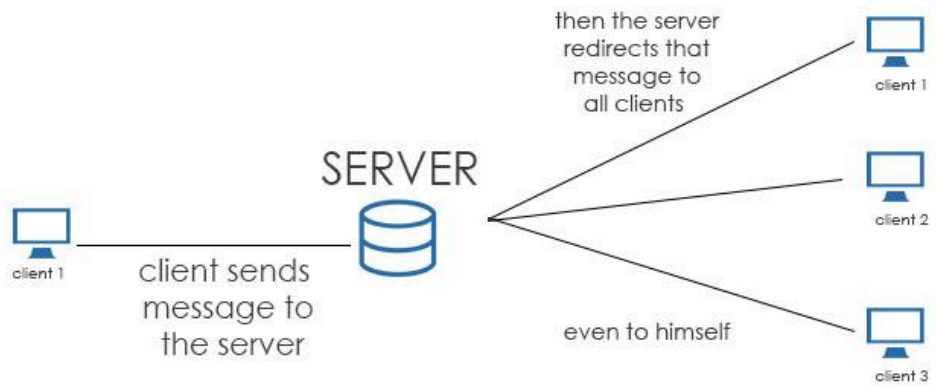
RELAZIONE ELABORATO PROGRAMMAZIONE DI RETI:

-Realizzato da Brighi Federico-



Traccia 1: Sistema di Chat Client-Server

“Implementare un sistema di chat client-server in Python utilizzando socket programming. Il server deve essere in grado di gestire più client contemporaneamente e deve consentire agli utenti di inviare e ricevere messaggi in una chatroom condivisa.Il client deve consentire agli utenti di connettersi al server, inviare messaggi alla chatroom e ricevere messaggi dagli altri utenti.”

Descrizione dei due codici :

Questa documentazione fornisce una panoramica dei miei codici di client e server implementati in linguaggio Python per gestire una chat room condivisa di utenti.
Il server è progettato per consentire a più client di connettersi simultaneamente e comunicare tra di loro attraverso lo scambio di messaggi di testo.

Funzionamento del Sistema:

Il sistema del server si basa su socket di rete e threading per gestire le connessioni multiple dei client. Ecco una panoramica delle principali funzionalità:

-Accettazione delle Connessioni In Entrata: Il server utilizza la funzione “incoming_connections” che attende le connessioni in entrata dai client utilizzando un socket di tipo TCP.

-Connessione al Server: Il client si connette al server utilizzando un socket di tipo TCP/IP e specificando l'indirizzo IP del server e la porta di accesso.

-Gestione dei Client: Ogni volta che un client si connette, il server avvia un thread separato (handling_client) per gestire la comunicazione con quel client. Il client è invitato a inserire il proprio nome e riceve un messaggio di benvenuto.

-Invio di Messaggi: I client possono inviare messaggi di testo attraverso la chat. I messaggi sono trasmessi a tutti gli altri client connessi (compreso il mittente), creando una così una chat room virtuale.

-Ricezione dei Messaggi: Il client utilizza un thread separato per ricevere i messaggi della chat e visualizzarli nella finestra GUI.

-Gestione della Chiusura: Il client gestisce la chiusura della finestra GUI e la comunicazione di disconnessione al server quando l'utente decide di chiudere il client.

-Disconnessione dei Client: Il server gestisce le disconnessioni dei client in modo sicuro, rimuovendo il client dalla lista dei partecipanti e notificando agli altri la sua uscita.

-Gestione delle Eccezioni: Il codice include gestione delle eccezioni per gestire eventuali errori come "OSError" e "ConnectionResetError" durante la comunicazione con i client o la chiusura del server.

-Protocollo dei Messaggi: Il server utilizza un protocollo semplice per il messaggio di uscita ("{quit}") per consentire ai client di lasciare la chat.

-Concorrenza e Multithreading: Il server implementa la gestione concorrente delle connessioni client utilizzando il threading, consentendo la connessione simultanea di più client.

-Interfaccia Grafica: Il client utilizza Tkinter per creare un'interfaccia grafica semplice che consente all'utente di scrivere e inviare messaggi.

Requisiti di Esecuzione:

Per eseguire con successo il codice di client e server, è necessario soddisfare i seguenti requisiti:

Python 3.x: Assicurarsi di avere una versione di Python successiva alla terza installato sul proprio sistema.
Librerie Standard di Python: Il codice utilizza le librerie standard di Python come socket, threading, e tkinter. Assicurarsi che queste librerie siano disponibili nell'ambiente Python.

Alcune considerazioni aggiuntive per testare il codice:

1 - Dal proprio terminale, recarsi nella directory (CODE) che contiene i due file "server_fede.py" e "clients_fede.py".

2 - Avviare in questa console il codice del server digitando "py server_fede.py".Una volta avviato il server sarà in ascolto.

3 - Aprire altre console in base a quanti client si vuol far comunicare, recarsi nella stessa directory e avviare il codice del client digitando "py clients_fede.py".

4 - Una volta lanciato il comando verrà richiesto di inserire HOST e NUMERO DI PORTA,quindi digitare rispettivamente 127.0.0.1 per l'host e 2003 per la porta.

5 - Verrà aperta una finestra che rappresenta la chatroom,qui dentro ogni client dovrà prima registrarsi indicando il proprio nome e poi potrà scambiare messaggi con gli altri client.

6 - Per uscire sono disponibili 2 modalità: o si invia tramite la chatroom il messaggio "{quit}" e così solo il client corrente verrà terminato,oppure se viene chiusa forzatamente la console del server da parte dell'utente,appena si prova a scrivere sulla chat room si viene espulsi,con scritta "Server has closed".