PyChat

Alex Guerrini; 0001090495

26 maggio 2024

Indice

1 Introduzione

L'obiettivo del progetto è realizzare un'applicazione di chat che consente a più utenti di connettersi a un server centrale per comunicare tra loro in una chatroom condivisa.

Il progetto è composto da due parti principali: il client e il server. Il server gestisce le connessioni dei vari client e distribuisce i messaggi in modalità broadcast all'interno della chatroom. Il client permette agli utenti di collegarsi al server tramite il protocollo TCP, inviare messaggi e visualizzare quelli ricevuti dagli altri utenti. È importante notare che i messaggi inviati prima della connessione di un nuovo client non sono disponibili per quest'ultimo.

Nel corso dello sviluppo del progetto sono state effettuate diverse scelte architetturali. Ad esempio, è stato scelto il protocollo TCP per la comunicazione tra client e server per la sua affidabilità nella trasmissione dei dati. Inoltre, l'uso dei thread permette di gestire in modo concorrente l'invio e la ricezione dei messaggi su entrambi i lati della connessione, migliorando l'esperienza utente.

Nel codice del client è stata implementata una logica per gestire l'invio e la ricezione dei messaggi tramite thread separati, in modo che l'utente possa scrivere e leggere contemporaneamente senza bloccare l'esecuzione del programma. Analogamente, il server gestisce le connessioni multiple dei client e distribuisce i messaggi a tutti i client connessi.

In sintesi, il progetto mira a fornire una soluzione stabile e affidabile per consentire a più utenti di comunicare in tempo reale attraverso una chatroom condivisa.

2 Client

Il client utilizza un approccio connection-oriented IPv4 tramite il modulo socket di Python per stabilire una connessione con il server della chatroom. Prima di avviare l'applicazione, l'utente deve specificare come parametri l'indirizzo IP del server e la porta.

Una volta avviato, il client crea un socket e tenta di connettersi al server utilizzando l'indirizzo IP e la porta specificati. Se la connessione ha successo, il client invia il nome utente al server per identificarsi.

Dopo la connessione, il client avvia due thread separati per gestire l'invio e la ricezione dei messaggi.

Il thread di invio consente all'utente di digitare un messaggio e inviarlo al server tramite il socket. Prima di inviare il messaggio, viene applicata la

codifica UTF-8 per garantire la compatibilità con caratteri speciali, come le lettere accentate.

Il thread di ricezione rimane in attesa di messaggi dal server e li decodifica usando la codifica UTF-8 per una corretta visualizzazione. Quando arriva un nuovo messaggio, il client lo mostra sulla console.

L'interfaccia utente fornisce le istruzioni necessarie per interagire con l'applicazione, come i comandi per inviare messaggi.

Una volta che l'utente vuole terminare la connessione, gli basta digitare il comando `quit` e ogni altro utente verrà notificato della sua uscita attraverso un messaggio in chat e l'aggiornamento della lista utenti presente a sinistra della chat box.

3 Server

Il server della chatroom è progettato per permettere agli utenti di connettersi e comunicare utilizzando il protocollo TCP/IP. Il server mantiene una lista dei client connessi e gestisce la trasmissione dei messaggi tra di essi.

Il server è implementato per essere sempre pronto a ricevere pacchetti da ogni dispositivo collegato. Per gestire contemporaneamente le connessioni multiple, il server utilizza un approccio basato su thread, con un thread separato per ogni connessione attiva.

Il server della chatroom offre le seguenti funzionalità principali:

- Gestione delle connessioni dei client.
- Trasmissione di messaggi broadcast a tutti i client connessi.
- Fornitura della lista degli utenti connessi.

Il codice del server è suddiviso nelle seguenti sezioni:

- Gestione delle connessioni: Il server accetta le connessioni dei client e li aggiunge alla lista dei client connessi.
- Trasmissione dei messaggi: Il server gestisce la trasmissione dei messaggi broadcast tra i client connessi.

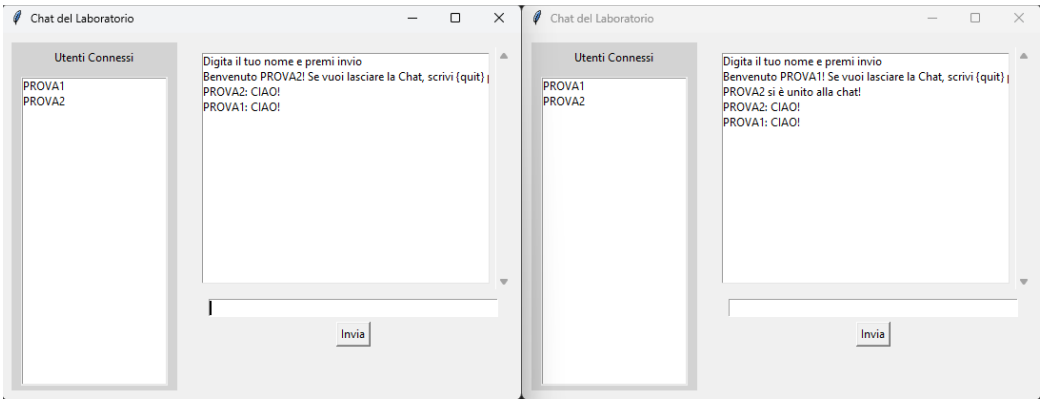

Figura 1: GUI della chatroom