Documentazione Progetto Programmazione di Reti

Alessio Bifulco

June 10, 2024

Traccia del Progetto

Progetto di Programmazione di Reti: 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.

1 Descrizione del Sistema

Il sistema di chat client-server implementato consente a più client di connettersi a un server centralizzato e di scambiare messaggi all'interno di una chatroom condivisa. Ogni client può inviare e ricevere messaggi in tempo reale, rendendo possibile la comunicazione tra tutti i partecipanti connessi.

2 Funzionamento del Sistema

2.1 Server

Il server è progettato per accettare connessioni da più client contemporaneamente e per gestire la trasmissione dei messaggi all'interno della chatroom. Quando un messaggio viene inviato da un client, il server lo riceve e lo inoltra a tutti gli altri client connessi, escluso il mittente.

Il server utilizza i seguenti componenti principali:

- **Socket**: Utilizzato per la comunicazione di rete.
- **Threading**: Utilizzato per gestire le connessioni multiple contemporaneamente.
- **Lista dei Client**: Mantiene traccia dei client connessi.

2.2 Client

Il client permette agli utenti di connettersi al server, di inviare messaggi e di ricevere i messaggi inviati dagli altri utenti nella chatroom. Il client dispone di una semplice interfaccia grafica (GUI) che facilita l'interazione con il sistema.

Il client utilizza i seguenti componenti principali:

- **Socket**: Utilizzato per la comunicazione di rete.
- **Threading**: Utilizzato per ricevere messaggi dal server in modo asincrono.
- **Tkinter**: Libreria utilizzata per creare l'interfaccia grafica.

3 Requisiti per Eseguire il Codice

Per eseguire il sistema di chat client-server, è necessario disporre di:

- Python 3.x installato sul sistema.
- La libreria Tkinter, che è generalmente inclusa con Python.
- Connessione di rete stabile per la comunicazione tra server e client.

4 Gestione degli Errori e delle Eccezioni

Il codice è progettato per gestire vari tipi di errori ed eccezioni:

- **Errore durante l'invio dei messaggi**: Se un client non è più disponibile, il server lo rimuove dalla lista dei client.
- **Errore di connessione**: Se un client non riesce a connettersi al server, viene visualizzato un messaggio di errore.
- **Errore nella ricezione dei messaggi**: Se un client non riesce a ricevere messaggi, il thread di ricezione termina e viene visualizzato un messaggio di errore.

5 Considerazioni Aggiuntive

- **Scalabilità**: Il sistema può essere scalato per gestire un numero maggiore di client, ma potrebbe essere necessario implementare ulteriori ottimizzazioni per garantire prestazioni ottimali.
- **Sicurezza**: La sicurezza non è stata considerata in questa implementazione di base. In un'applicazione reale, sarebbe necessario implementare meccanismi di autenticazione e crittografia per proteggere i dati trasmessi.

6 Conclusione

Il sistema di chat client-server implementato soddisfa i requisiti di base per la comunicazione in una chatroom condivisa. La gestione delle connessioni multiple e la trasmissione dei messaggi avvengono correttamente, e il client fornisce un'interfaccia grafica semplice per l'interazione con il sistema. La gestione degli errori è implementata per garantire che eventuali problemi vengano gestiti in modo appropriato.