
SISTEMA DI CHAT CLIENT-SERVER

Elaborato Programmazione di Reti

Autore

Cristina Murvai

Università di Bologna, Campus di Cesena

Anno Accademico 2023/2024

Contents

1	Introduzione	3
2	Descrizione del Sistema	3
2.1	Server	4
2.2	Client	4
3	Guida all'esecuzione del codice	5
4	Esempi di funzionamento	6
5	Considerazioni aggiuntive	8

1 Introduzione

Lo scopo di questo progetto per l'esame di programmazione di reti è quello di implementare un sistema di Chat Client-Server in Python utilizzando socket programming. Sarà necessario garantire a più utenti l'accesso ad una chatroom condivisa e la possibilità di scambiarsi messaggi broadcast.

2 Descrizione del Sistema

I sistemi client-server sono un modello architetturale in cui le risorse e i servizi sono suddivisi tra due tipi di nodi: il client e il server. Il client richiede e consuma servizi forniti dal server, mentre il server fornisce risorse e gestisce le richieste dei client. Questo modello favorisce una distribuzione efficiente delle risorse, la scalabilità e la gestione del carico di lavoro, rendendolo ampiamente utilizzato in varie applicazioni distribuite. In particolare, in questo

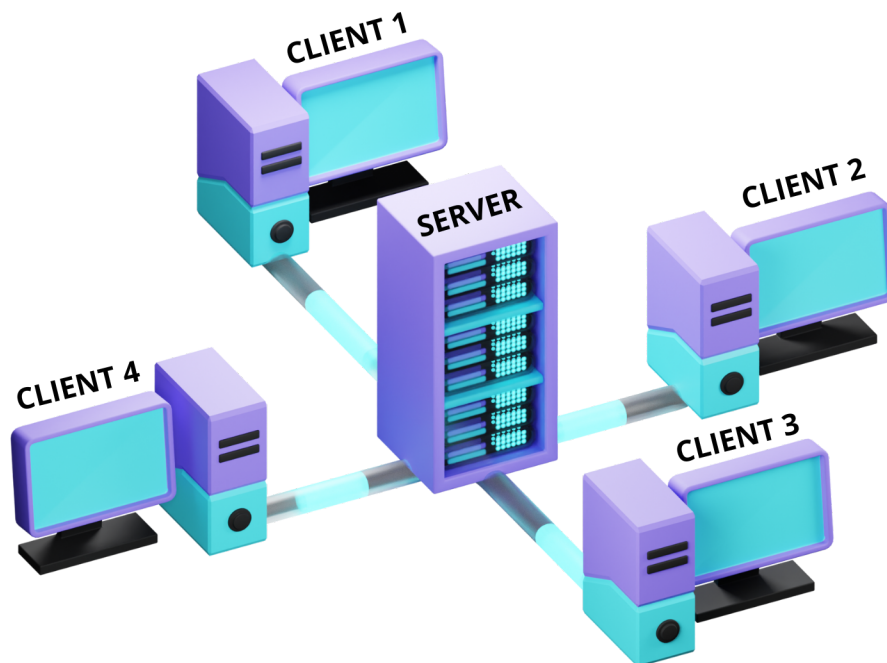


Figure 1: Esempio configurazione Client-Server

progetto Client e Server sono modellati da due distinte classi Python, sotto descritte in maniera più dettagliata.

2.1 Server

Il server è il fulcro del nostro sistema di chat client-server. È responsabile della gestione delle connessioni dei client, dell'invio e della ricezione dei messaggi tra i client e del mantenimento della chatroom condivisa.

Il server utilizza il modulo socket di Python per creare un socket TCP/IP e stabilire una connessione con i client. Ogni volta che un nuovo client si connette, il server accetta la connessione e avvia un thread separato per gestire il client. Questo consente al server di continuare ad accettare nuove connessioni senza bloccarsi.

Il server tiene traccia dei client connessi, mantenendo un dizionario dei loro nomi utente, richiesti all'inizio della connessione, e degli indirizzi IP. Quando un client invia un messaggio, il server lo trasmette a tutti gli altri client connessi in broadcast.

Tramite un meccanismo di eccezioni sono gestite tutte le condizioni che rendono il server impossibilitato ad accettare nuove connessioni e gli eventuali messaggi relativi alla perdita di connessione da parte di un client.

Si dispone infine di un'interfaccia testuale per monitorare lo stato delle connessioni e le attività dei client (entrata e uscita dalla chat).

2.2 Client

Il client è responsabile della gestione dell'interfaccia utente e della comunicazione con il server. Utilizza il modulo socket per connettersi al server tramite un socket TCP/IP.

Una volta connesso, il client, anch'esso rappresentato da un thread, può inviare e ricevere messaggi dalla chatroom condivisa. Una volta che un utente ha stabilito la connessione con il server, gli viene richiesto di impostare un username nel campo di input. Successivamente sarà possibile digitare messaggi e inviarli al server, che li distribuirà a tutti gli altri client connessi.

Il client dispone di un'interfaccia grafica basata su Tkinter che consente agli utenti di scrivere e visualizzare messaggi in modo intuitivo. L'interfaccia utente comprende un campo di input per digitare i messaggi, una lista per visualizzare i messaggi precedenti e un pulsante per inviare i messaggi.

3 Guida all'esecuzione del codice

Per eseguire correttamente il codice è necessario seguire i seguenti passaggi:

1. Assicurarsi di avere Python installato sul sistema.
2. Salvare i file `server.py` e `client.py` all'interno di una cartella.
3. Aprire due finestre della shell o del prompt dei comandi.
4. Nella prima finestra, navigare nella directory contenente `server.py` e eseguirlo con il comando:

```
python3 server.py
```

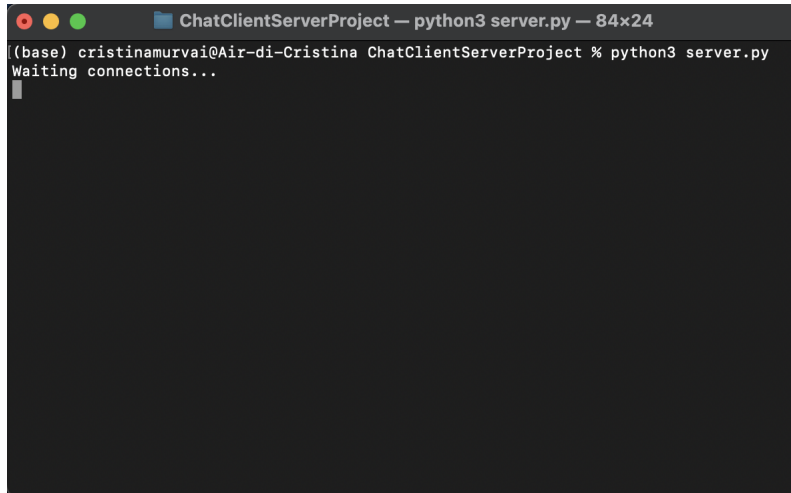
5. Nella seconda finestra, navigare nella directory contenente `client.py` e eseguirlo con il comando:

```
python3 client.py
```

6. Ripetere l'operazione precedente in un numero di volte pari al numero di client che si desidera connettere al server.
7. Seguire le istruzioni visualizzate nell'interfaccia del client per connettersi al server e iniziare a chattare.

4 Esempi di funzionamento

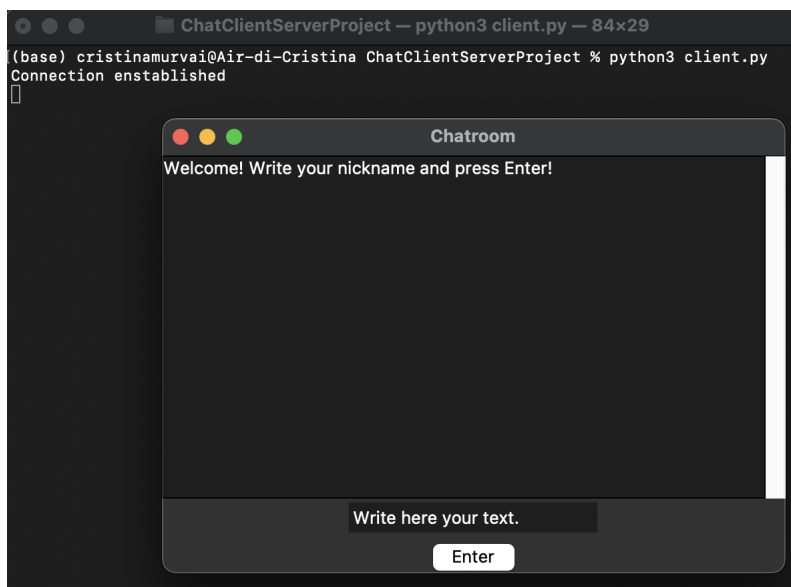
Per fare un esempio di funzionamento del sistema supponiamo di aver appena messo il server in esecuzione, ciò che vediamo sono i log del server, come mostrato in figura 2.

A terminal window titled "ChatClientServerProject — python3 server.py — 84x24". The prompt is "(base) cristinamurvai@Air-di-Cristina". The command "python3 server.py" has been executed, and the output is "Waiting connections...".

```
ChatClientServerProject — python3 server.py — 84x24
(base) cristinamurvai@Air-di-Cristina ChatClientServerProject % python3 server.py
Waiting connections...
```

Figure 2: Visualizzazione testuale stato del server in attesa di connessioni

A questo punto supponiamo di effettuare la connessione di un client al server. Ci troviamo davanti un'interfaccia come quella rappresentata dalla figura 3, dove ci viene chiesto di inserire un username.

A terminal window titled "ChatClientServerProject — python3 client.py — 84x29". The prompt is "(base) cristinamurvai@Air-di-Cristina". The command "python3 client.py" has been executed, and the output is "Connection established". Overlaid on the terminal is a "Chatroom" window. The chatroom window has a title bar "Chatroom" and a message "Welcome! Write your nickname and press Enter!". At the bottom of the chatroom window is a text input field with the placeholder "Write here your text." and an "Enter" button.

```
ChatClientServerProject — python3 client.py — 84x29
(base) cristinamurvai@Air-di-Cristina ChatClientServerProject % python3 client.py
Connection established
```

Chatroom

Welcome! Write your nickname and press Enter!

Write here your text.

Enter

Figure 3: Utente appena entrato nella chatroom

Supponiamo poi di ripetere la stessa operazione per un altro client e di scambiare alcuni messaggi. Il risultato ottenuto è rappresentato in figura 4.

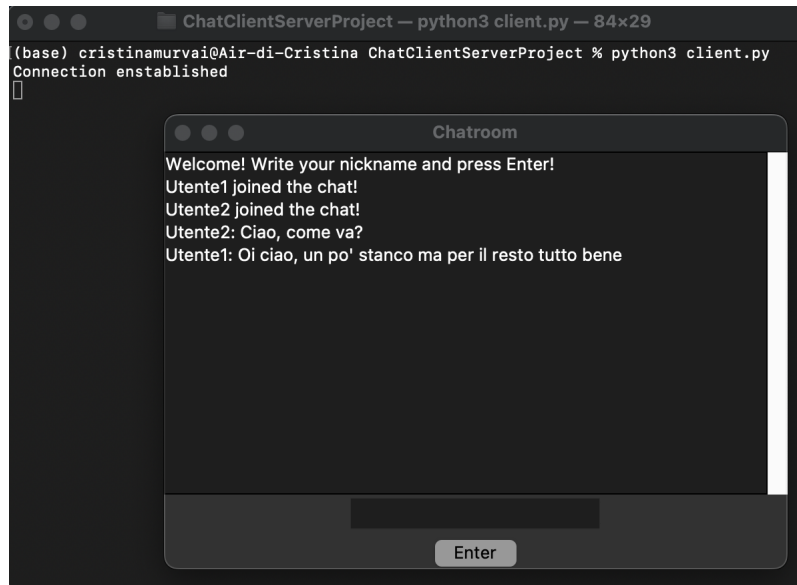


Figure 4: Estratto di chat tra due utenti

Se infine uno dei due utenti, in questo caso *User1*, si disconnette, i log del server mostrano ciò che è possibile vedere in figura 5.

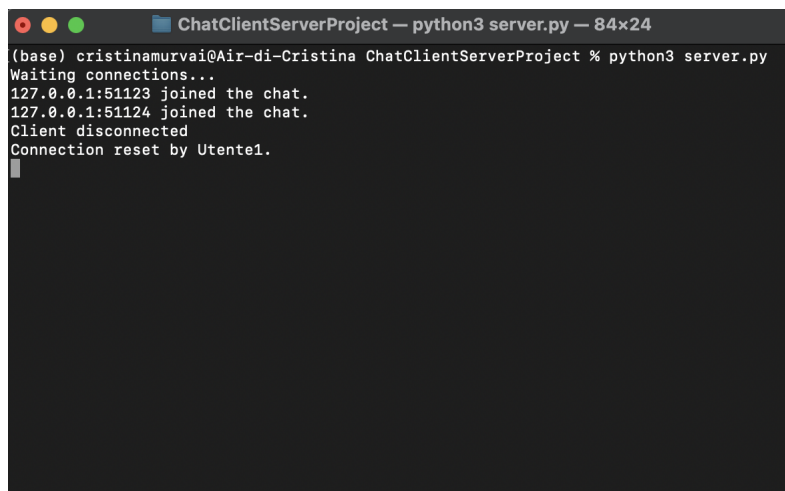


Figure 5: Visualizzazione testuale stato del server quando due utenti si sono collegati e in seguito uno di loro si è scollegato

5 Considerazioni aggiuntive

Il sistema di chat client-server implementato offre una base per la comunicazione in tempo reale tra più utenti su una rete. Tuttavia, possono essere apportate diverse migliorie per aumentare la sicurezza e le funzionalità del sistema. Ad esempio, potrebbe essere implementata l'autenticazione degli utenti, la gestione degli errori più dettagliata e la supporto per file sharing o chat private.