Offline Messenger

Hoamea Vanessa-Mihaela
Facultatea de Informatică, Universitatea "Alexandru Ioan Cuza" din Iași

Rezumat

Acest proiect constă în dezvoltarea unei aplicații client-server în limbajul C, care permite schimbul de scurte mesaje între utilizatorii online în timp real, asemănător cu un chatroom. Utilizatorii au de asemenea posibilitatea de a trimite mesaje private și utilizatorilor offline – acestora din urmă apărându-le mesajele noi atunci când se vor loga în aplicație. Alte funcționalități includ: trimiterea unui răspuns la anumite mesaje, vizualizarea istoricului conversațiilor pentru și cu fiecare utilizator în parte.

1 Introducere

Aplicația le permite utilizatorilor logați să trimită mesaje către alți utilizatori. Pentru a avea acces la aplicație, utilizatorii trebuie să se logheze într-un cont unic, cu username și parolă (în cazul în care nu au încă cont, își pot crea unul). După ce s-a conectat, utilizatorul poate vedea mesajele trimise public de către ceilalți utilizatori online, precum și trimite propriile sale mesaje sau verifica lista utilizatorilor logați la momentul dat. De asemenea, poate trimite mesaje private către utilizatori offline, verifica istoricul conversațiilor pe care le-a avut, răspunde la anumite mesaje, afișa mesajele primite cât timp era offline (dacă a primit mesaje). Pentru a ieși din aplicație, se introduce comanda `/quit`.

2 Tehnologii utilizate

Aplicația este bazată pe modelul client-server, și este scrisă în limbajul C.

Folosește **protocolul TCP**, care permite transferul de date fără pierdere de informații. Deși UDP este un protocol mult mai rapid, simplu și eficient față de TCP, aplicația noastră necesită confirmarea primirii datelor și posibilitatea detectării erorilor pentru a ne asigura că funcționează corect, acest lucru fiind posibil doar prin utilizarea TCP.

De asemenea, programul operează pe principiul de **multithreading** (ce constă în execuția mai multor fire de execuție în același pipeline). Avantajele aduse de acest tip de implementare în comparație cu crearea de noi procese prin `fork` sunt viteza execuției și intercomunicației. Serverul creează un thread pentru fiecare client și îi tratează în mod concurent.

3 Arhitectura aplicației

Protocolul ce stă la baza aplicației este TCP, care oferă posibilitatea comunicării bidirecționale între server și client.

Conexiunea între client și server se realizează prin apelarea funcției `socket` și completarea structurii `sockaddr_in` cu datele corespunzătoare.

Pe partea de server, apelul `bind` asignează adresa serverului la socketul reprezentat de descriptorul `sd`. Pregătirea socketului pentru a accepta conexiuni de la clienți se face prin apelul funcției `listen`, a cărui parametru, 3, reprezintă numărul maxim de clienți pe care îi putem ține în coada de așteptare. Extragerea unui client din coada de așteptare și stabilirea unei conexiuni se va realiza cu ajutorul funcției `accept`.

Pe partea de client, funcția `connect` conectează descriptorul de socket la adresa serverului.

Odată ce se stabilește conexiunea și clientul reușește să se logheze cu succes într-un cont, comunicarea client-server devine posibilă. În momentul conectării la server, clientului i se va atribui un thread

prin intermediul căruia se va efectua această comunicare. În mod specific, trimiterea/recepționarea mesajelor se va realiza cu ajutorul funcțiilor `send` și `recv`.

Închiderea unei conexiuni se întâmplă în momentul în care clientul trimite comanda `/quit`, care închide execuția programului.

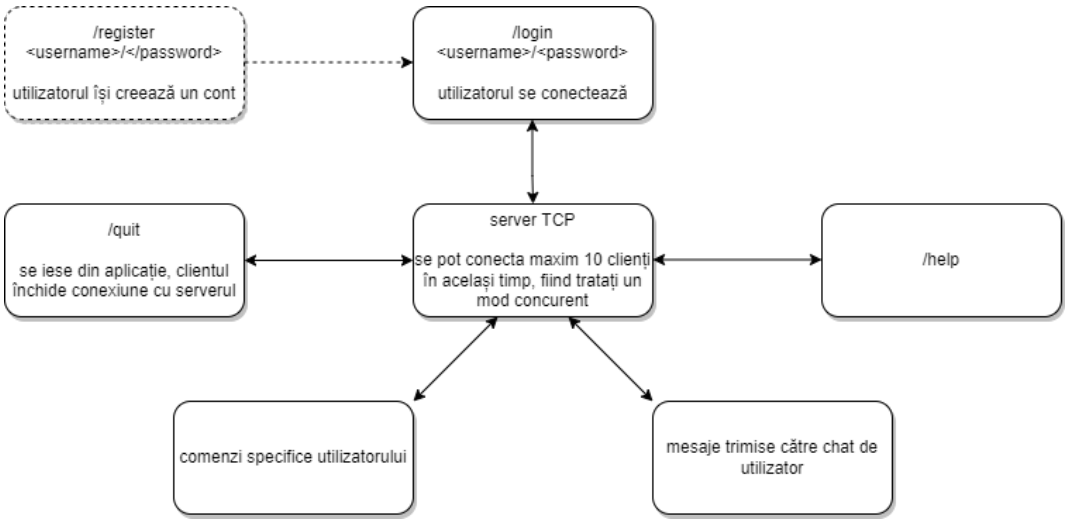

Figura 1: Diagrama aplicației.

4 Detalii de implementare

Programul funcționează conform schemei de mai sus. Pentru a avea acces la funcționalitățile oferite de aplicație, utilizatorul trebuie neapărat mai întâi să se logheze într-un cont valid. Dacă utilizatorul nu reușește să se logheze din prima, programul îi va permite să încerce din nou până când fie introduce date corecte, fie închide aplicația. Dacă s-a efectuat logarea cu succes, va începe comunicarea cu serverul.

Comenzile disponibile odată ce un utilizator devine logat sunt următoarele:

- `/help` – detalii despre modul de utilizare al aplicației
- `/register <username>/<password>` – creează un cont nou
- `/login <username>/<password>` – logare într-un cont existent
- `<text>` – trimite mesaj către toți utilizatorii care sunt online
- `/to <username>: <text>` – trimite mesaj privat către utilizatorul identificat prin `<username>` (acesta trebuie să fie offline)
- `/reply <message id>: <text>` – răspunde la mesajul cu ID-ul <message id>
- `/view` – afișează mesajele primite în timp ce utilizatorul era offline (dacă există)
- `/history [<username>]` – dacă nu se specifică un utilizator, va afișa istoricul utilizatorului care execută comanda; altfel, va afișa istoricul dintre utilizatorul care execută comanda și utilizatorul specificat în cel de-al doilea parametru
- `/chat-history` – afișează istoricul chat-ului public
- `/online-users` – afișează toți utilizatorii care sunt online în acel moment
- `/quit` – închide aplicația

5 Concluzii

Câteva moduri în care aplicația poate fi îmbunătățită ar fi: utilizarea unei baze de date pentru a stoca conturile existente, criptarea parolelor pentru a îmbunătății securitatea, adăugarea unui timestamp la fiecare mesaj în parte pentru a putea ține cont de când au fost trimise.

De asemenea, consider că prin adăugarea unor noi funcționalități, precum posibilitatea de a crea groupchat-uri și vizualizarea tuturor utilizatorilor care sunt online la un moment dat, calitatea aplicației și satisfacția utilizatorilor ar crește.

6 Surse

- https://profs.info.uaic.ro/~computernetworks/
- https://ro.wikipedia.org/wiki/Transmission_Control_Protocol
- https://ro.wikipedia.org/wiki/Multithreading
- https://www.lifesize.com/en/blog/tcp-vs-udp/
- http://www.geekride.com/fork-forking-vs-threading-thread-linux-kernel/
- https://man7.org/linux/man-pages/index.html
- Pentru diagramă: https://app.diagrams.net/