Offline Messenger

Grigoriu Ştefan-Cosmin, an 2, grupa E2

Universitatea "Al. I. Cuza", Informatică Iaşi, RO
https://profs.info.uaic.ro/ computernetworks/
stefan.grigoriu@info.uaic.ro

Abstract. În această documentaţie vom introduce subiectul abordat, vom discuta despre motivaţia alegerii proiectului propus, implementarea lui, caracteristicile utilizate în cadrul acestuia, conceptele, precum şi detalii de implementare şi scenarii de utilizare, iar în final vom concluziona cu ajutorul motivaţiei alegerii proiectului, precum şi cu îmbunătăţirile viitoare posibile.

1 Introducere

Proiectul ales, **Offline Messenger**, cere dezvoltarea unei aplicaţii ce se bazează pe tipul client/server ce permite schimbul de mesaje între doi, trei sau mai mulţi utilizatori conectaţi, dar în acelaşi timp să ofere aceeaşi funcţionalitate şi către utilizatorii ce încă nu sunt în momentul trimiterii mesajelor conectaţi la server. Astfel, utilizatorilor **offline** le vor fi arătate mesajele primite în momentul conectării la server. O altă funcţionalitate propusă proiectului prezentat este aceea ca utilizatorii să aibă posibilitatea de a trimite un răspuns în mod specific la anumite mesaje primite de la alţi utilizatori, lucru ce implică necesitatea stocării unui istoric al mesajelor transmise tuturor utilizatorilor, acest istoric fiind accesibil de către clienţi, la cerere, pentru şi cu fiecare utilizator în parte.

1.1 Motivaţia

Apariţia internetului a produs schimbări majore în viaţa oamenilor, fiind un mediu prietenos pentru implementarea aplicaţiilor de orice fel, de la o simplă comunicare via text sau video, pâna la business-uri online, cloud computing, dezvoltarea inteligenţei artificiale, etc. Cu ajutorul internetului reuşim să comunicăm cu persoane aflate la distanţe foarte mari de noi, în timp real, lucru ce ne uşurează viaţa de zi cu zi şi ne menţine aproape de cei dragi. Astfel, pentru a putea înţelege mai bine şi a acumula cunoştinţele necesare programării cu ajutorul protocoalelor de tip TCP/IP, precum şi pentru a ne dezvolta experienţa programării cu ajutorul proceselor-copil/threadurilor şi socketurilor, am ales crearea unei aplicaţii de tip **Messenger**, astfel punând bazele acesteia şi confruntându-ne cu problemele apărute pe parcurs.

2 Tehnologiile utilizate

Dat fiind faptul că această comunicare prin intermediul aplicaţiei **Messenger** va trebui realizată cu ajutorul internetului pentru a conecta clienţii la un anumit server, am ales folosirea protocolului de tip TCP/IP. Deoarece prin intermediul serverului vor fi conectaţi mai mulţi utilizatori ce vor efectua operaţiuni în paralel, serverul va trebui să fie unul concurent.

2.1 Protoclul TCP/IP

Protocolul TCP/IP a fost ales pentru a fi folosit în acest proiect din mai multe motive, acestea fiind următoarele:

- Deoarece acesta este un protocol orientat pe conexiune, el se ocupă de transmiterea în siguranţă, fără erori, a tuturor octeţilor trimişi de la maşina sursă.

- Protocolul se ocupă de asemenea şi de controlul fluxului de informaţii pentru ca destinatarul să nu fie inundat cu mesaje mai multe decât poate procesa, iar astfel octeţii sunt transmişi în ordinea exactă trimiterii, fie că mesajul a fost transmis în unul singur de dimensiune mare, sau mai multe de dimensiuni mici.

- Dat fiind faptul că în aplicaţia noastră este necesară o comunicare bidirecţională, cu ajutorul nivelului reţea ("IP"), pachetele sunt trimise de la o maşină la alta cu ajutorul adresei IP si a unui PORT, comunicarea fiind una full-duplex.

2.2 Baza de date

Dată fiind necesitatea stocării informaţiilor de tip user, parola, istoric mesaje, etc. pentru fiecare client în parte, o bază de date va trebui folosită. Astfel, la momentul conectării unui client, acesta se poate fie înregistra, fie poate intra în contul său folosind un user şi o parolă înregistrate în tabelul din baza de date. Mai apoi, toate mesajele necitite de acesta îi vor fi arătate, iar noile mesaje ale clientului vor fi memorate în baza de date în istoricul acestuia, precum şi al destinatarilor.
Pentru o uşurinţă în implementarea unui SGBD, am ales folosirea unui tip de bază de date embedded, ce nu necesită un anumit server sau o altă configurare, denumit **SQLite**, fiind una dintre cele mai folosite din lume şi având librării native C.

3 Arhitectura aplicaţiei

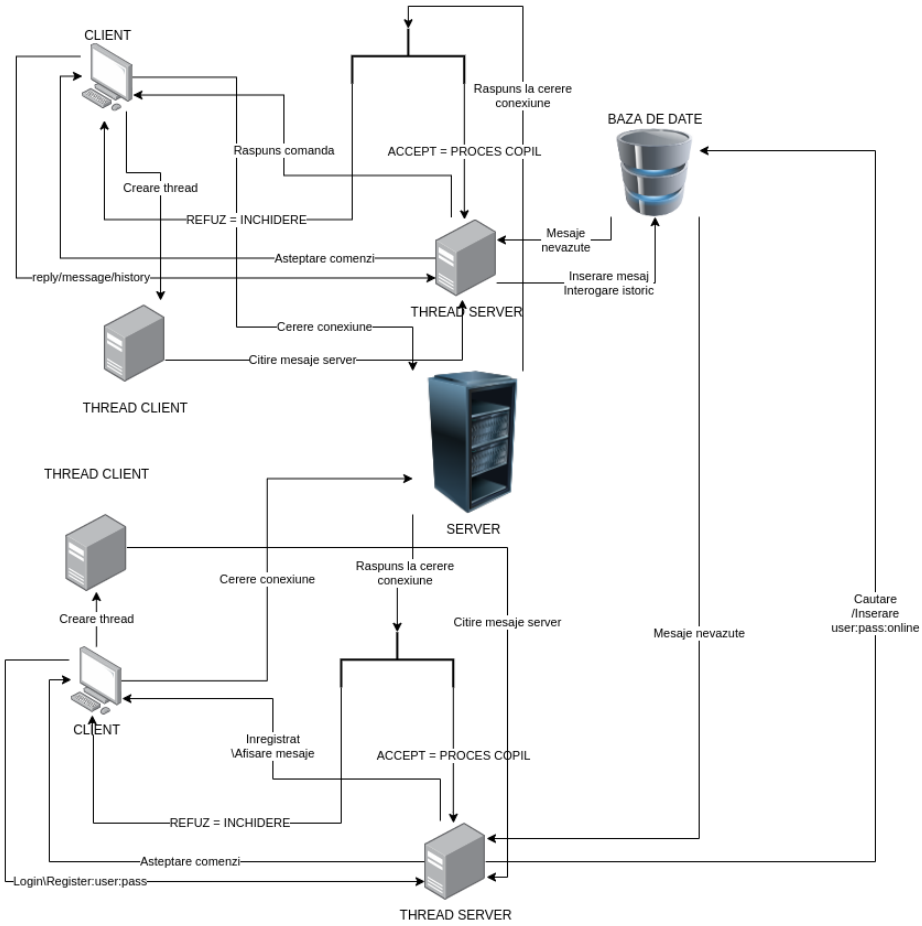

Fig. 1. Diagrama cu arhitectura aplicaţiei

Aplicaţia prezentată în diagrama de mai sus ilustrează comunicarea între doi clienţi, comunicare posibilă prin intermediul serverului şi a proceselor-copil create de acesta. În practică, în momentul cererii de conectare utilizând protocolul TCP/IP si socketurile, serverul, în caz de acceptare a conexiunii, se foloseşte de procese-copil sau threaduri pentru a îndeplini cererile fiecărui client, iar tot prin acest proces-copil/thread facilitează comunicarea clienţilor şi stocarea mesajelor cu ajutorul unei baze de date.

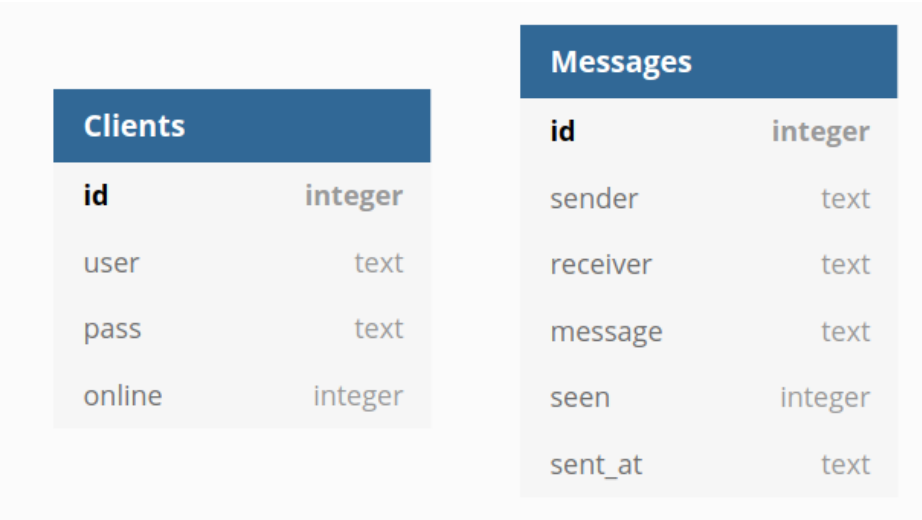

Fig. 2. Tabele baza de date

4 Detalii de implementare

Paşii pentru a utiliza aplicaţia propusă, **Offline Messenger**, sunt următorii:

1. Serverul este pornit şi aşteaptă conexiuni la IP-ul propriu şi PORT-ul desemnat;

2. Clientul este pornit, iar adresa şi PORT-ul sunt specificate;

3. Clientul trimite o cerere de conectare către server, cerere ce va fi acceptată, iar serverul va crea un thread pentru comunicarea cu clientul;

4. Clientul poate folosi doar comanda 'login:user:pass' sau 'register:user:pass';
   (a) Dacă clientul a ales comanda **register**, câmpurile user şi pass vor fi introduse în baza de date, cu condiţia ca userul să nu fie deja existent, iar mai apoi userul va fi online;
   (b) Dacă clientul a ales comanda **login**, câmpurile user si pass vor fi verificate cu toate conturile existente în baza de date, iar în caz de potrivire, userul va fi online, iar daca nu, atunci va putea încerca din nou;

5. Dupa ce a fost logat, clientul va crea un thread ce se va ocupa de citirea mesajelor de la server şi i se vor afişa pe ecran toate mesajele primite cât timp a fost offline. Acesta poate scrie în fereastră pentru a vedea comenzile disponibile sau poate folosi direct comenzile **'history:all'** sau **'history:user'**, **'reply:[data timp]'**, **'msg:user'** sau **'quit'**;

(a) La comanda **'history:all/user'**, clientul va alege fie să vadă tot istoricul conversaţiilor lui cu toţi utilizatorii cu care a comunicat, fie poate alege să vadă doar istoricul mesajelor cu un anumit utilizator;
(b) La comanda **'reply:[data timp]'**, clientul va răspunde la mesajul selectat cu ajutorul datelor trimiterii acestuia, ce apar pe ecran, iar răspunsul va fi stocat în baza de date şi trimis persoanei ce a scris mesajul;
(c) La comanda **'msg:user'**, clientul va trimite un mesaj privat userului specificat, iar cel din urmă va fi notificat, mesajul apărându-i pe ecran împreună cu persoana data, ora şi numele utilizatorului ce a trimis mesajul;
(d) La comanda **'quit'** clientul este delogat, threadurile din client se vor închide şi userul va fi trecut offline în baza de date, iar mesajele noi vor fi salvate şi afişate la următorul login;

La final, după închiderea clientului, threadurile se vor opri şi nu vor exista procese zombie. În caz că serverul va avea probleme(pană de curent, etc.), operaţiunile neterminate nu vor fi trecute în tabelurile din baza de date.

5 Concluzii

În concluzie, proiectul propus, **Offline Messenger**, este unul ce necesită bazele programării deprinse din reţelele de calculatoare, mai exact de ajutorul proceselor-copil, comunicării cu ajutorul socketurilor, protocoalelor de comunicare, threadurilor, dar care în acelaşi timp vine cu noutăţi precum implementarea unei baze de date pentru stocarea mesajelor şi utilizatorilor, în acelaşi timp învăţându-ne bune practici de programare.

5.1 Îmbunătăţiri

În cazul unei aplicaţii de tip **Messenger**, foarte multe îmbunătăţiri îi pot fi aduse, începând de la posibilitatea trimiterii diferitelor fişiere între utilizatori precum documente sau fotografii, implementarea unui *voice-chat* şi a unei interfeţe grafice sau folosirea unei camere web, pana la criptarea tuturor conversaţiilor pentru o siguranţă şi intimitate sporită a utilizatorilor.

Codul proiectului se găseşte pe GitHub, la link-ul nr 3. din Bibiliografie.

References

1. TCP/IP, https://ro.wikipedia.org/wiki/TCP/IP
2. Computer Networks, https://profs.info.uaic.ro/ computernetworks/
3. Github Repository, https://github.com/grigoriucosmin/CN_Project/