Universitatea “Alexandru Ioan Cuza”

Facultatea de Informatica

Retele de Calculatoare

Offline Messenger

Autor: Crucerescu Ion 2B6

Coordonator: Colab. Damian Catalin

27 ianuarie 2017

Cuprins:

1.**Introducere**

Documentatia ce va urma a fi prezentata, descrie aplicatia client/server, offline messenger, care nu este nimic altceva decat o simpla modalitate de comunicare intre doi sau mai multi utilizatori. Punctul forte al acestei aplicatii reprezinta minimul de resurse de care este necesar pentru a se efectua o comunicare cu posibilitati acceptabile pentru fiecare dintre utilizatori. Pentru ca un utilizator sa poata benificia de toate serviciile care i le ofera aplicatia acesta trebuie sa isi creeze un cont dupa care urmeaza sa se autentifice,in caz contrar acestia nu vor avea posibilitatea de a trece peste etapa de autentificare pentru a benificia de serviciile oferite, si anume:

- Inregistrare de noi utilizatori
- Autentificare in system, in cazul in care acesta dispune de un cont valid
- Vizualizarea unei liste care cuprinde toti utilizatorii din system
- Trimiterea unei cereri de prietenie unui utilizator
- Vizualizarea listei care cuprinde toate cererile de prietenie
- Acceptarea/refuzarea tuturor sau a unei cereri de prietenie specific
- Vizualizare listei care cuprinde toti prietenii tai
- Stergerea unui prieten din lista de prieteni
- Trimiterea unui mesaj catre un prieten specificat
- Stergerea unui prieten din lista cu prieteni
- Posibilitate de a oferi un raspuns (reply) la un mesaj anumit
- Trimiterea unui mesaj catre un prieten chiar daca acesta este offline
- Utilizatorul este notificat daca acesta are o cerere de prietenie/ mesaj nou
- Autentificarea aceluasi user de pe mai multe aplicatii
- Delogarea care ii permite sa se autentifice de pe alt utilizator

2.**Tehnologii utilizate**

Pentru realizarea acestui proiect a fost utilizat protocolul TCP/IP pentru stabilirea conexiunii intre clienti si server, si fisiere .config care simuleaza o baza de date.

TCP/IP (Transmission Control Protocol/Internet Protocol) este cel mai utilizat protocol folosit în reţelele locale cât şi pe Internet datorită disponibilităţii şi flexibilităţi lui având cel mai mare grad de corecţie al erorilor.
TCP/IP permite comunicarea între calculatoarele din întreaga lume indiferent de sistemul de operare instalat.

Protocolul corespunde nivelului transport din stiva TCP/IP. TCP oferă un serviciu de comunicare la un nivel intermediar între un program de aplicație și Protocolul Internet (IP). Atunci când un program de aplicare dorește sa trimită o bucată mare de date pe Internet, în loc sa fragmenteze datele în pachete IP de dimensiuni mici și sa emită o serie de cereri pentru protocolul IP, software-ul poate emite o cerere unică pentru TCP și sa lase protocolul TCP sa se ocupe de detaliile de manipulare.

Protocolul TCP este un protocol sigur orientat-conexiune care permite trimiterea unui flux de octeti de la o masina la alta fara erori. Protocolul este sigur intrucat nu au loc pierderi de informatii. Pentru fiecare pachet IP trimis se asteapta o confirmare de primire. Daca aceasta nu apare intr-un anumit interval de timp, pachetul este retransmis.

3.**Arhitectura aplicatiei**

Aplicatia cuprinde patru elemente de baza:doua servere si doi clienti. Pentru ca un client sa poata beneficia de toate serviciile care le ofera aplicatia, acesta, trebuie sa dispuna de aplicatia de tip client care cuprinde doua fisiere. Aceasta aplicatie a fost organizata astfel incat resursele clientului sa fie utilizate la un nivel minim, tot din acest motiv s-a renuntat la o interfata grafica. Aplicatia offline messenger descrisa in aceasta documentatie tinde spre o aplicatie de tip RPC cu o pondere de 90%.

3.1.**Serverul I**

Acest server reprezinta centrul tuturor interschimbarilor de informatii in aplicatie, mai putin al mesajelor dintre utilizatori.

Primul server are rolul de:

1. A efectua conexiunea cu clientii intr-un mod concurrent
2. Deserveste fiecare dintre client cu ajutorul un process auxiliar care este creat de fiecare data cand apare un client, acesta preluand toate informatiile de care dispune serverul la moment
3. Pentru fiecare client este alocata o zona de memorie aparte pentru a nu aparea conflicte pe viitor cu privire la datele personale ale fiecarui utilizator
4. Toate informatiile despre utilizator sunt stocate in fisierele serverului astfel clientul nu dispune de fisiere care ar putea fi vulnerabile unui atac cybernetic
5. Toata interfata de care dispune clientul este oferita de serverul I astfel resursele folosite de client pentru aceasta aplicatie sunt minime

3.2.**Serverul II**

Serverul II reprezinta un manager pentru fiecare mesaj si fiecare comanda care este executata in fereastra chatului.

Al doilea server fiind unul auxiliar dispune de o lista mai mica de servicii, si anume:

1. Asigura transmiterea automata a mesajului de la un utilizator specific la un destinatar
2. Asigura notificarea clientului in cazul in care acesta are mesaje necitite
3. Asigura un istoric cu conversatiile sale pentru fiecare client cu fiecare dintre prietenii sai
4. In cazul aparitiei comenzii speciale “reply” acesta raspunde la o intrebare specificata de utilizator in comanda sa.
5. Ofera posibilitatea un utilizator de a intretine mai multe conversatii in acelasi timp

3.3.**Clientul I/II**

Aplicatia client in acest proiect este foarte putin dezvoltata pentru a oferi clientului o portabilitate maxima si un consum de resurse minimal. Primul client nu reprezinta nimic altceva decat o fereastra in care acesta primeste toate informatiile de care are nevoie de la server si anume:

login	Autentificare in sistem
register	Inregistrarea unui utilizator nou
quit	Inchiderea aplicatiei
friendrequest	Vizualizarea listei de cereri de prietenie
userslist	Vizualizarea listei de utilizatori in sistem
friendlist	Vizualizarea liste de prieteni
logout	Delogarea de pe utilizatorul current
sendrequest	Trimite cerere de prietenie
back	Intoarcere la meniul precedent
accept_username	Accepta cererea de prietenie a utilizatorului
accept_all	Accepta cererea de prietenie a tutuor
refuse_username	Refuza cererea de prietenie a utilizatorului
refuse_all	Refuza cererea de prietenie a tuturor
send_message_user	Trimite mesaj utilizatorului
delete_username	Sterge din lista de prieten utilizatorul

3.4.**Conexiunea**

Oricare dintre clienti la conectare este supus unui test de validare pentru a identifica daca clientul care se conecteaza apartine aplicatiei. Aceasta validare reprezinta un cod unic care este cerut de server, in cazul in care acest cod unic nu corespunde cu codul cerut de server, clientul este refuzat de server si se inchide conexiunea., acest lucru se efectueaza din motive de securitate. Daca conexiunea cu clientul se pierde la un moment dat serverul primeste o erroare de conexiunea astfel aceesta ignora deconectarea clientului si inchide procesul care deserveste clientul deoarece toate modificarile si inregistrarile de date care sunt efectuate de client sunt aplicate in baza de date in timp real, astfel nici o informatie despre utilizator nu se va pierde. Acelasi lucru se intampla si momentul in care serverul se deconecteaza, clientul este notificat si primeste o erroare de notificare, dar toate datele despre utilizator sunt salvate.

4.**Detalii de implementare**

Serverul I ca structura este un server concurrent care deserveste clientii in mod concurent prin intermediul instructiunii fork().

- name_to_id() //convertor nume user in id user
- command_type_fr1() //tip de commanda accept/refuse in meniul pentru friendrequest
- command_type_fr2() //tip de commanda pentru all/sau un id concret din structura database
- command_type_fr3() //tip de commanda delete/refuse in meniul petru friendrequest
- command_type_fr4() //tip de commanda pentru meniul friendlist care returna id existent din database
- build_structure() //functie pentru generearea structurii din fisierul cu conturi
- n_matrix() //returneaza numarul de users din database
- build_descriptors() //flagul de online/offline pentru fiecare din useri
- read_matrix() //citeste matricea de tangente din fisier
- check_friendreq() //returneaza notificarea de friendrequest
- frql() //returneaza lista de friendrequesturi pentru userul curent
- check_messages() //verifica daca un user a trimis mesaj clientului
- friend_list_generate() //genereaza lista curenta de prieteni al clientului
- update_messages() //notifica un prieten ca ii trimiti un mesaj
- check_friendlist() //notifica client ca are un mesaj
- build_matrix() //redacteaza matricea din fisier
- update_matrix() //update complet pentru matrice
- update_matrix1() //update partial pentru menu friendrequest
- update_matrix2() //update partial pentru menu friendlist
- username_check() //test pentru username daca respecta conditiile impuse
- password_check() //test pentru password daca respecta conditiile impuse
- write_user() //scrierea userului in fisierele .config
- register_username() //protocolul register pentru un nou user
- check_login() //testul de logare pentru un client
- menu() //meniul secundar pentru client
- deserveste() //meniul principal pentru client

5.**Concluzii**

Aplicatia Offline Messanger reprezinta un proiect de tip client/server care asigura comunicarea intre doi sau mai multi utilizatori. Datorita implementarii unui server concurent aplicatia poate deservi un numar mare de utilizator in acelasi timp. Pentru a fi mai rapida si mai eficienta aplicatia dispune de doua servere care functioneaza concurent pentru o eficienta sporita. Serverele sunt sincronizate pentru a nu oferi date eronante clientului. Numarul mare de comenzi disponibile pentru client ii ofera utilizatorului o mobilitate sporita. In baza de date a serverului nu exista useri cu prioritate fata de alti clienti astfel nimeni nu are drepturi asupra celuilalt. Comunicarea intre client se efectueaza doar cu acordul partenerului astfel nu sunt intalnite situatii neplacute (ex: spamuri). Protocolul TCP este un protocol sigur orientat-conexiune care permite trimiterea unui flux de octeti de la o masina la alta fara erori, plus este sigur intrucat nu au loc pierderi de informatii. Si cum toate informatiile sunt server-sided clientul se poate simti in siguranta.

6. Bibliografie

- https://en.wikipedia.org/wiki/Transmission_Control_Protocol
- http://profs.info.uaic.ro/~adria/teach/courses/net/files/NetEx/S9/cliTcp.c
- http://profs.info.uaic.ro/~adria/teach/courses/net/files/NetEx/S12/ServerConcThread/servTcpConcTh2.c
- http://stackoverflow.com/questions/2347770/how-do-you-clear-console-screen-in-c
- http://www.cplusplus.com/reference/cstring/strstr/
- http://www.tutorialspoint.com/c_standard_library/c_function_fopen.htm
- http://stackoverflow.com/questions/24194961/how-do-i-use-setsockoptso-reuseaddr