Raport tehnic pentru proiectul „Offline Messenger(B)”

Blîndu Andrada Mihaela

1. Introducere:

Proiectul Offline Messenger constă în crearea unei aplicații de tipul clienți/server prin care mai mulți clienți au posibilitatea de a trimite mesaje între ei, cu condiția de a fi logați, dar și de a trimite mesaje către un client offline. Mesaj care va rămâne in așteptare într-un istoric până când clientul se va conecta din nou. Procesul de autentificare va fi efectuat prin trimiterea unei comenzi care va prelua username-ul si parola clientului, dupa care comunicarea poate incepe.

1. Tehnologii folosite:

* Programul este scris in limbajul C;
* Serverul este un server TCP bazat pe socket-uri care utilizeaza multithreading pentru a gestiona clientii;
* Pentru a asigura comunicarea intre client si server am folosit protocolul TCP(**T**ransmission **C**ontrol **P**rotocol ), deoarece:

-putem trimite „pachete” de dimensiuni mari, fara a se pierde informatii pe parcurs, deci este un protocol sigur, atat pentru server cat si pentru client;

-se bazeaza pe mesajul de confirmare din partea receptorului, pentru a fi siguri ca mai putem continua trimite pechete in siguranta;

-folosim un socket pentru a realiza conexiunea dintre server si client. Realizarea socketului este posibila datorita perechei de forma adresa IP : port, care face legatura.;

-in server, pentru fiecare client conectat, serverul creaza un socket, iar pentru a gestiona clientii, comunicarea si preluarea datelor, vom folosi threaduri;

* Pentru stocarea datelor tirmise de clienti folosesc fisiere text, in care voi salva username-urile, parolele, si istoricul mesajelor;

1. Structura aplicatiei:

Aplicatia se bazeaza pe schimbul de comenzi prestabilite dintre client si server in terminal, fara a avea momentan o interfata grafica. Pe server se vor putea conecta mai multi clienti deodata.

Arhitectura aplicatiei va fi compusa din mai multe comenzi:

1. Autentificarea: fiecare client va trece printr-un proces de autentificare format din 2 etape:

-register: in care clientul isi va pune un unsername si o parola;

-login: daca username-ul exista deja, atunci acesta se va loga doar;

1. Trimiterea de mesaje: Dupa logare clientul va fi considerat online, iar acesta va putea alege sa scrie un mesaj unei persoane din lista de useri;

3)Afisarea istoricului: Clientul va avea optiunea, dupa conectare, de a vedea mesajele primite cat timp a fost offline. In perioada in care clientul este offline, mesajele primite vor fi salvate intr-un fisier specific pentru fiecare user, in care se vor salva toate mesajele primite de la useri separat, care mai apoi vor putea fi vizionate dupa conectare;

4)Lista de useri online si offline: Clientii vor putea scrie aceasta comanda pentru a vedea cine din lista de useri este online la acel moment, sau cine este offline;

5)Logout: clientul se poate deconecta de la aplicatie, chiar daca intre timp altii clienti inca sunt conectati, deoarece serverul va fi deschis in continuare;

6)Exit

Diagrama pentru vizionarea comenzilor schematic:

A group of rectangular white rectangles

Description automatically generated

Diagrama interna a aplicatiei pentru comunicarea client-server:

A black background with white rectangles

Description automatically generated

1. Aspecte de implementare:

Comunicarea client-server se realizeaza prin intermediul unui socket. Un socket accepta comuniarea bidirectionala intre server si client folosind un protocol bine stabilit. Protocolul creaza un socket la un anumit port, si asteapta pana cand un client solicita o conexiune. Odata stabilita conexiunea serverul creaza fluxuri de intrare si iesire catre socket si incepe sa trimita si sa primeasca mesaje.

A computer screen with text

Description automatically generated

TCP ul ruleaza pe un port specific. Cand un client se conecteaza la server, serverul accepta conexiunea si lanseaza un nou thread pentru a gestiona cererile acestui client. In timp ce threadul proceseaza cererile clientului, serverul continua sa asculte si sa accepte noi clienti, adica noi cereri de conectare. Fiecare conexiune noua creaza un thread nou, permitand serverului sa gestioneza multiple cereri de client in paralel.



Pentru executarea comenzilor, mesajele care ajung in server vor fi prelucrate si in functie de cuvantul cheie, se vor duce catre functia care se va ocupa de ea. De exemplu daca avem „Register”, „Login”, „Scrie mesaj” si tot asa. Datele importante, cum ar fi, username-urile, parolele, mesajele pentru clientii ofline, vor fi salvate in fisiere pentru a putea fi folosite ulterior.

**Scenarii pentru utilizarea aplicatiei in situatii de zi cu zi:**

In ziua de azi aplicatiile de messenger si comunicare intre clienti sunt foarte folosite si au ajuns la un nivel foarte securizat si sigur. Conceptul de inceput a avut intentia de a face comunicarea intre retele diferite posibila, intrucat atunci era posibila comunicarea doar celor din aceeasi retea. In ziua de astazi toata lumea a auzit de WhatsApp, fiind totodata si cea mai folosita platforma pentru comunicare, urmat de Messenger-ul detinut de Facebook si multe altele. Fiecare platforma de social media, are si sectiunea de chatting, pana si site-urile pentru promovare au sectiunea in care poti pune intrebari in chat pentru a discuta cu vanzatorul.

Scopul acestui proiect este de a intelege mai bine conceptele unei comunicari, si procesele care au loc „behind” la tot ce se vede la suprafata, cat si partea de server si protocoale folosite care fac posibil tot ce se intampla.

Imbunatatiri: In primul rand partea de interfata grafica ar schimba proiectul mult mai mult si l-ar face mai user-friendly si mai usor de folosit decat sa scriu fiecare comanda in terminal. Totodata ar mai fi:

-posibilitatea de a banna un cont, sau a bloca trimiterea de mesaje catre cineva;

-crearea unui grup de 3 sau mai multi clienti care pot comunica odata;

-posibilitatea de a recupera parola daca ai un cont dar ai uitat parola;

Bibliografie:

<https://profs.info.uaic.ro/%7Ecomputernetworks/index.php> (in general)

<https://eng.libretexts.org/Bookshelves/Computer_Science/Programming_Languages/Java_Java_Java_-_Object-Oriented_Programming_(Morelli_and_Walde)/15%3A_Sockets_and_Networking/15.06%3A_Client_Server_Communication_via_Sockets> (documentatie socket)

<https://app.diagrams.net/> (pentru diagrame)

<https://ro.wikipedia.org/wiki/Transmission_Control_Protocol> (pentru TCP, informatii suplimentare)