Tema 2

Pârțac Laura

08.12.2021

Introducere

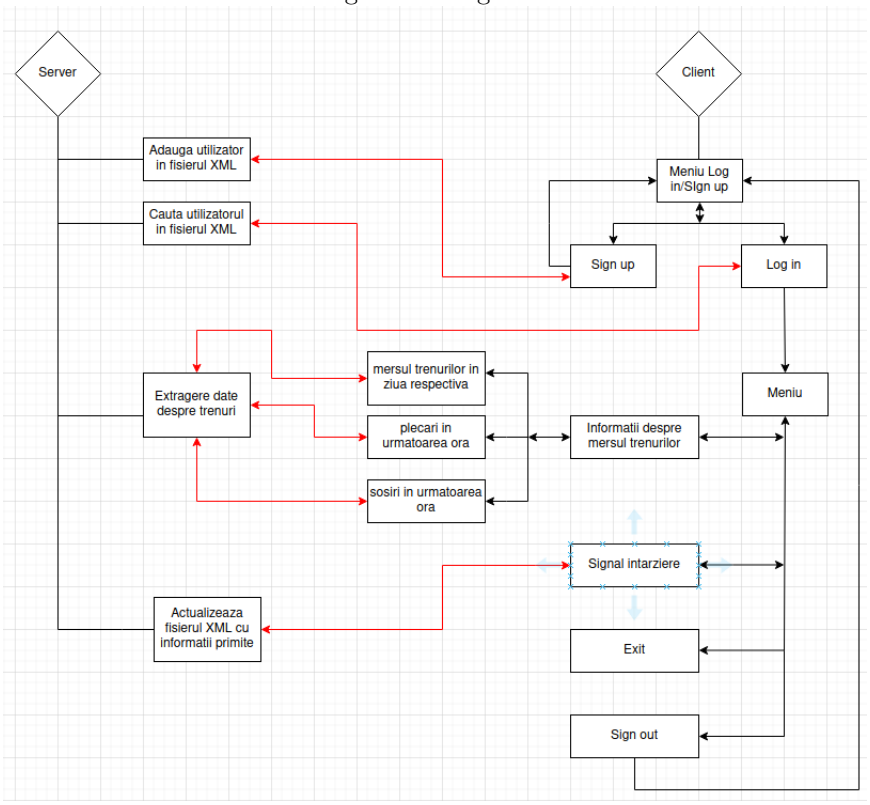
Am ales proiectul Mersul Trenurilor deoarece consider că reprezintă o aplicație folositoare prin posibilitatea utilizării sale de către o gamă largă de utilizatori, printre care se numără și studenții pentru care mijlocul de transport în comun reprezintă o mare parte din viața de zi cu zi. Un alt motiv îl reprezintă posibilitatea aplicației de a încuraja folosirea trenurilor mai frecventă, acestea reprezentând un mijloc de transport mai prietenos pentru mediul înconjurător în contextul actual al încălzirii globale. Astfel, proiectul are la bază o aplicație server/client astfel încat prin client utilizatorii înregistrați pot primi informații despre mersul trenurilor, dar pot la rândul lor să ofere informații privind întârzieri ale unor trenuri pentru a le face disponibile altor utilizatori. Pe de altă parte, serverul trimite informațiile solicitate de către utilizator către client și actualizează datele despre trenuri în funcție de semnalările primite de la client.

Tehnologii utilizate

Principala tehnologie utilizată este un server TCP/IP concurent, deoarece este un protocol care îndeplinește necesitățile aplicației care este de tip server/client. Acest lucru se datorează faptului că TCP, fiind orientat conexiune, presupune ca aplicația care primește informații primește exact aceeași secvență de octeți care a fost trimisă de cealaltă aplicație, fapt ce asigură că utilizatorii vor primi informații corecte privind mersul trenurilor.

Arhitectura aplicației

Figure 1: Diagrama



În Figura 1 se evidențiază o arhitectură preliminară a aplicației, reprezentând atât serverul cât și clientul, iar cu sageți roșii sunt evidențiate legăturile dintre funcțiile din server și cele din client.

Serverul, prin utilizarea primitivei fork(), va avea posibilitatea să preia comenzi de la mai multi clienți, fiind un server concurent. Atât clientul, cât și serverul vor fi reprezentate de programe implementate în C/C++.

Detalii de implementare

Programul aferent clientului (client.cpp) va fi mijlocul prin care utilizatorii vor interacționa cu aplicația în sine. Acesta va avea o interfața grafică ușor de utilizat, iar programul va trimite, în funcție de cererea utilizatorului, comenzi serverului.

Serverul (server.cpp) se va ocupa de executarea comenzilor și va trimite mesaje înapoi către client. De exemplu, atunci când utilizatorul se autentifică

cu un username și o parolă validă (existente în fișierul XML citit de server), serverul va trimite un mesaj corespunzător clientului pentru a lăsa utilizatorul respectiv să acceseze meniul reprezentat în Figura 1. Vor exista în programul serverului mai multe funcții care se vor ocupa de prelucrarea mesajelor primite, citirea și modificarea fișierului XML sau căutarea unor date specifice în fișierul respectiv.

În Figura 2 se poate observa un exemplu de funcție prezentă în server care caută un anumit tren dupa ID-ul său.

Figure 2: Exemplu functie in server.cpp

```
//#define XMLDOC "trenuri.xml"
int searchTrain(char* idTrain)
{
    XMLDocument doc;
    doc.LoadFile( "trenuri.xml" );//se acceseaza fisierul XML
    XMLElement * pTop = doc.RootElement();//file

    XMLElement *pTrains = pTop->FirstChildElement("Trenuri");//trenuri
    XMLElement *pTrain = pTrains->FirstChildElement("Tren");
    while(pTrain)//se ia fiecare tren pe rand si se verifica id-ul pana cand se gaseste trenul cautat
    {
        XMLElement *pID = pTrain->FirstChildElement("IDtren");
        if(strcmp(idTrain, pID->GetText()) == 0)
        {
            cout<<"Found train!\n";
            return 1;
        }
        pTrain = pTrain -> NextSiblingElement("Tren");

    }
    return 0;
}
```

Datele despre mersul trenurilor vor fi păstrate într-un fișier XML. Serverul va citi, scrie și edita informațiile despre trenuri cu ajutorul librăriei TinyXML-2 care reprezintă un simplu, dar eficient, parser pentru XML implementat in C++, ușor de încadrat în alte programe.

Figure 3: Exemplu fișier XML cu date despre trenuri

```xml
<?xml version="1.0" encoding="utf-8"?>
<file>
    <Trenuri>
        <Tren>
            <IDtren> 1234 </IDtren>
            <plecare> Roman </plecare>
            <sosire> Iasi </sosire>
            <statusPlecare> 14:00 </statusPlecare>
            <statusSosire> 17:00 </statusSosire>
            <estim> 17:00 </estim>
        </Tren>
        <Tren>
            <IDtren> 1494 </IDtren>
            <plecare> Iasi </plecare>
            <sosire> Roman </sosire>
            <statusPlecare> 18:00 </statusPlecare>
            <statusSosire> 19:20 </statusSosire>
            <delayA> 30 </delayA>
            <estim> 19:50 </estim>
        </Tren>
    </Trenuri>
</file>
```

În figura 3 se poate observa un exemplu preliminar de fișier XML care conține informații despre trenuri. Atunci când un utilizator semnalează o întârziere la sosirea unui tren, serverul va adăuga un element $< delayA >$ în care va memora întârzierea, și va actualiza și elementul $< estim >$ reprezentativ pentru estimarea sosirii. Un alt scenariu ar fi atunci când sunt semnalate mai multe întârzieri pentru același tren de către utilizatori diferiți, caz în care serverul va actualiza $< delayA >$ cu o medie a acestor întârzieri. Se vor mai adăuga și elemente reprezentative pentru zilele în care merge un anumit tren, pentru a putea fi extrase datele acestuia atunci când un utilizator cere mersul trenurilor pentru ziua respectivă.

Într-o manieră asemănătoare vor fi adăugate și informațiile despre utilizatorii înregistrați, mai exact numele de utilizator și parola.

Serverul va extrage astfel informații din fișierul XML în funcție de comanda venită de la client, însă va și modifica aceste informații atunci când este cazul, de exemplu modificarea estimării sosirii sau adăugarea unui nou utilizator.

Concluzii

Așadar, proiectul Mersul Trenurilor are scopul de a îmbunătății experiența utilizatorilor atunci când aleg acest mijloc de transport. Acesta ar putea fi îmbunătățit prin oferirea mai multor opțiuni utilizatorilor în ceea ce privește

administrarea propriului cont, de exemplu adaugarea unei adrese de email sau numar de telefon având astfel posibilitatea de a-și recupera contul. O altă îmbunătățire ar putea fi adăugarea prețului biletelor sau alte informații prin care utilizatorii să aibă posibilitatea de a intra in contact direct cu reprezentanții unei anumite gări, de exemplu.

Bibliografie

Douglas E. Comer, Internetworking With TCP/IP, vol.I: Principles, Protocols, and Architecture (2nd edition), Prentice Hall, New Jersey, 1991

https://github.com/leethomason/tinyxml2

https://profs.info.uaic.ro/ computernetworks/cursullaboratorul.php

https://tinyxml2.docsforge.com/master/api/tinyxml2/