TopMusic

Facultatea de Informatica Iasi

Manoleasa tudor

13-12-2018

Abstract

Raportul presupune o analiza amanuntita a proiectului "TopMusic", a carui pasi vor fi descrisi in profunzime conform structurii ce se regaseste pe site-ul facultatii.

Contents

Chapter 1

Introducere

Proiectul se numeste ***"TopMusic"*** iar in vederea realizarii sale s-au luat in considerare urmatoarele functionalitati specifice unei aplicatii muzicale:

•*Inregistrarea si autentificarea utilizatorilor*
• *Adaugarea unei melodii in top*
• *Votul unei piese*
• *Vizualizarea unui top general*
• *Vizualizarea unui top particular*
• *Postarea unor comentarii*
• *Posibilitatea adminului de a elimina o piesa*
• *Posiblitatea adminului de a restrictiona un utilizator*

1.1 Cum functioneaza concret aplicatia

Utilizatorul se conecteaza mai intai, drept client, la server.Odata ce conexiunea a fost stabilita un meniu interactiv apare in fereastra-terminal. Aceasta interfata ii permite sa se logheze/autentifice iar in cazul unui succes ii ofera posibilitatea de a alege din 8 optiuni specifice tipului de user pe care il detine. In momentul in care user-ul doreste sa se delogheze, comanda **exit** ii va facilita acest lucru. Mai multi useri se pot conecta in acelasi timp intrucat posibilitatea rularii paralele este implementata iar topul se actualizeaza in timp real.

1.2 Comenzi & Fisiere

Acesta reprezenta "Dictionarul" comenzilor pe care un client le poate folosi in cadrul aplicatiei.

Optiune	Tip user	Efect	Repetabilitate	Fisier utilizat
1	admin+standard	adaugare piesa	da	songs.txt
2	admin+standard	votare piesa	da	songs.txt
3	admin+standard	top general	da	songs.txt
4	admin+standard	top specific	da	songs.txt
5	admin+standard	adaugare comentariu	da	comentarii.txt
6	admin+standard	vizualizare comentarii	da	comentarii.txt
7	admin	stergere piesa	da	songs.txt
8	admin	blocare utilizator	da	restrictii.txt

Utilizatorii inregistrati vor fi pusi intr-un fisier numit **users.txt**.
Piesele si toate informatiile cu privire la acestea se afla in fisierul **songs.txt**.
Comentariile referitoare la piese sunt in fisierul **comentarii.txt**.
Utilizatorii ce au restrictie la votare sunt in fisierul **restrictii.txt**

Chapter 2

Tehnologii utilizate

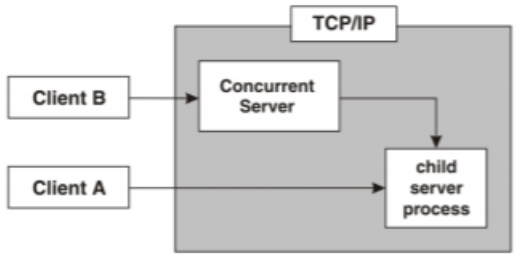
2.1 Fundamente teoretice

Intregul proiect este realizat prin intermediul protocolului ***TCP*** de la ***Nivelul de Transport***. Am ales TCP intrucat, spre deosebire de **UDP**, avem o conexiune, un transport *"in-order"*, notificari (in ambele capete) la trimiterea de date si retransmisia datelor (unul dintre factorii decisivi in vederea alegerii tehnologiei TCP intrucat, nu pierdem date). ***Internet protocol****-ul* folosit este ***IPv4***, din motive de convenienta. ***Socketul*** reprezinta principalul ***API*** din cadrul programului, facilitand comunicarea in ambele directii prin interfata de programare ***I/O*** deja cunoscuta. De asemenea, au fost folosite principalele primitive:

- *bind() - ataseaza o adresa locala la un socket*
- *listen() - permite unui socket sa accepte conexiuni*
- *accept() - blocheaza apelantul pana la sosirea unei cereri de conectare(utilizata de serverul TCP)*
- *connect() - tentativa (activa) de stabilire a conexiunii (folosita de clientul TCP)*
- *read() - citirea de date*
- *write() -scrierea de date*

2.2 Caracteristici

Paradigma de comunicare in retea: **Modelul Server/Client** →
Modul de interactiune : **Orientat conexiune, bazat pe TCP** →
Implementare concurenta



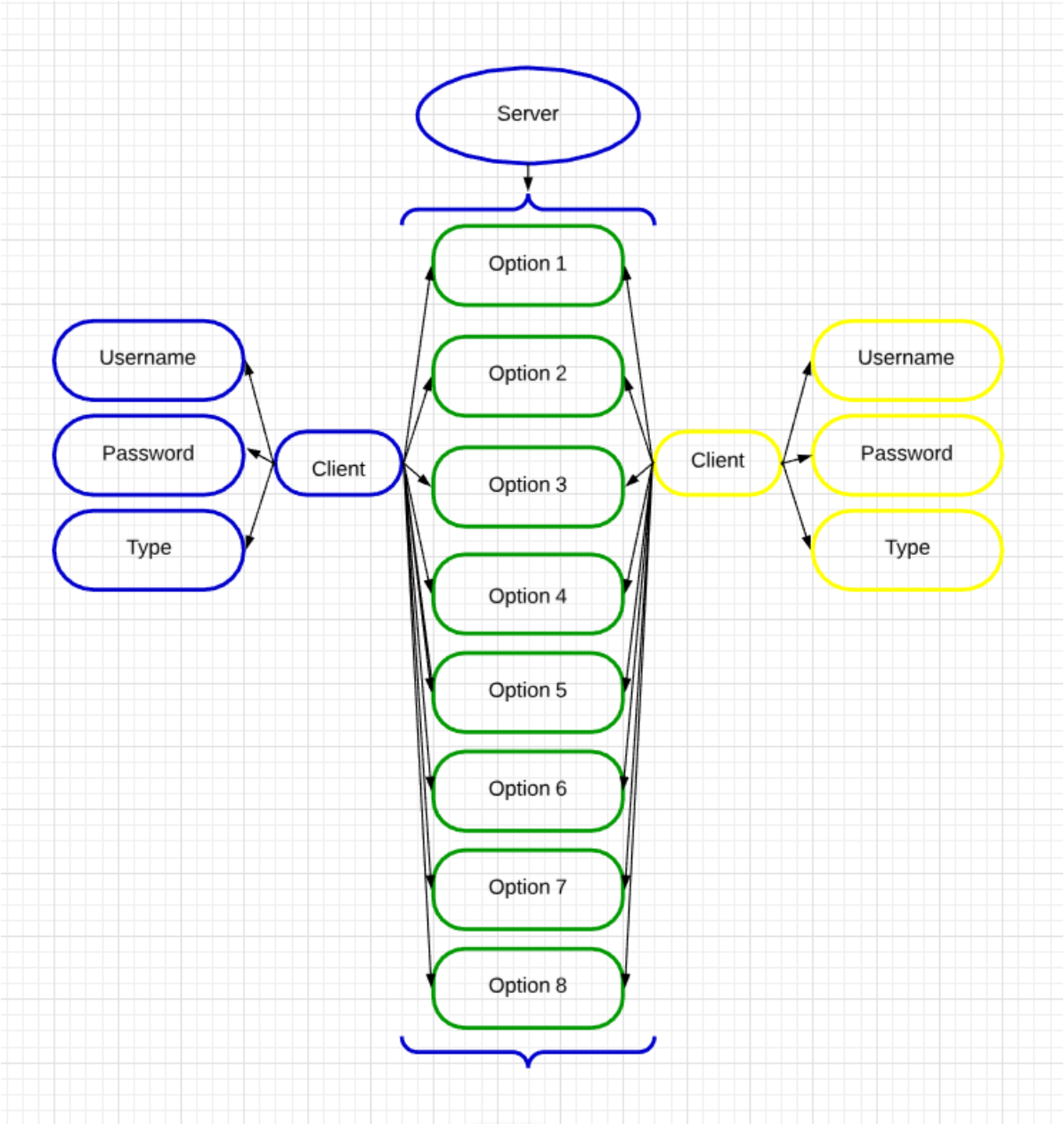
Cum tratam **Multitaskingul**?
Prin ***servere concurente pre-forked.*** Se creeaza un numar de procese copil imediat la initializare, fiecare proces liber interactionind cu un anumit client. Mai concret, in cazul nostru, avem un fork intr-o bucla *while* care trateaza toti clientii. Alegerea fork-ului in favoarea *select*-ului sau a *thread*-ului se datoreaza in principal functionalitatii. Un server cu *select* sau cu *thread*-uri nu ar fi functionat in aceasta situatie.

Chapter 3

Arhitectura aplicatiei

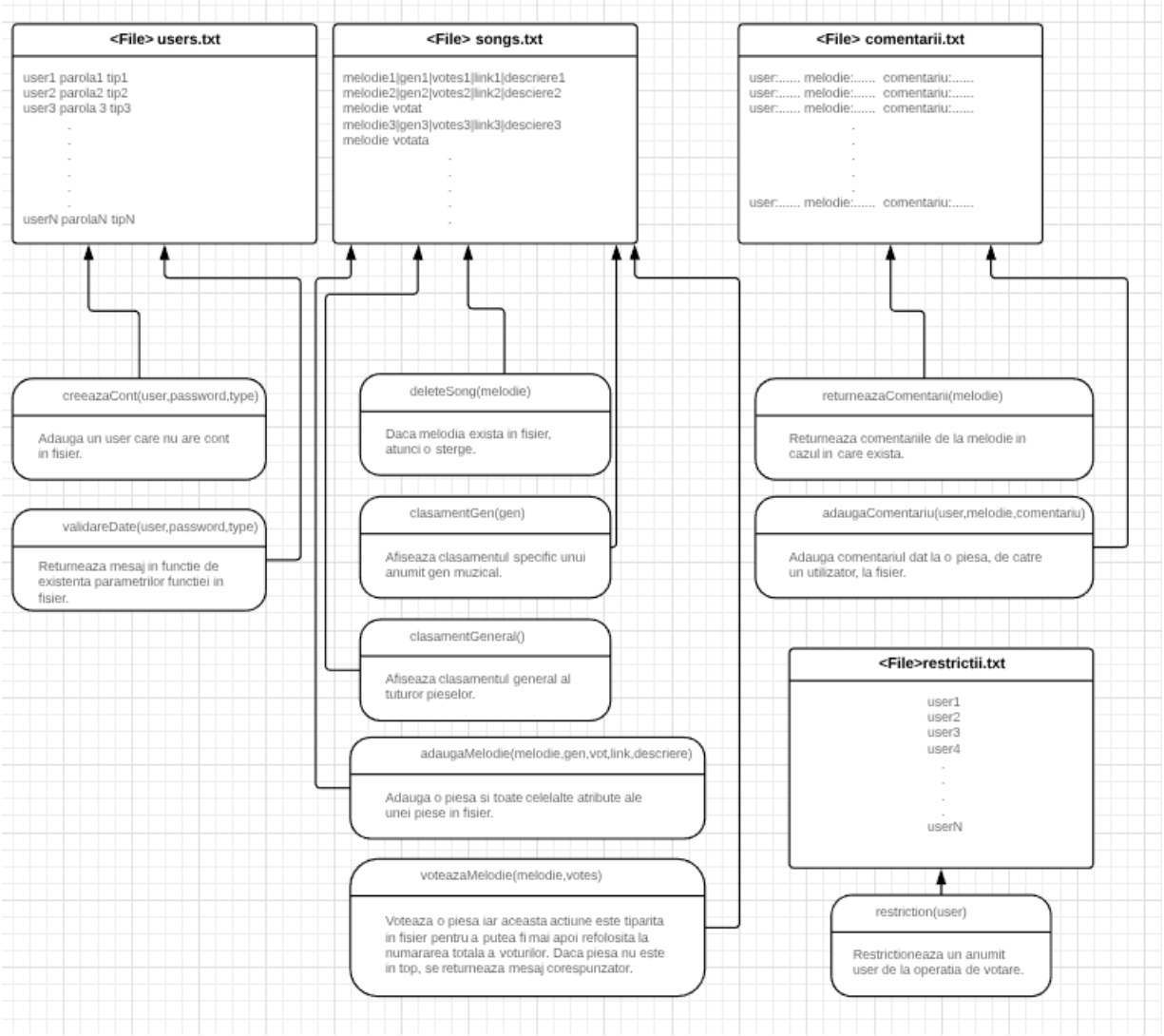
3.1 Diagrama server/client

In continuare, am realizat o scurta diagrama mai detaliata a modelului ***Server/Client***.

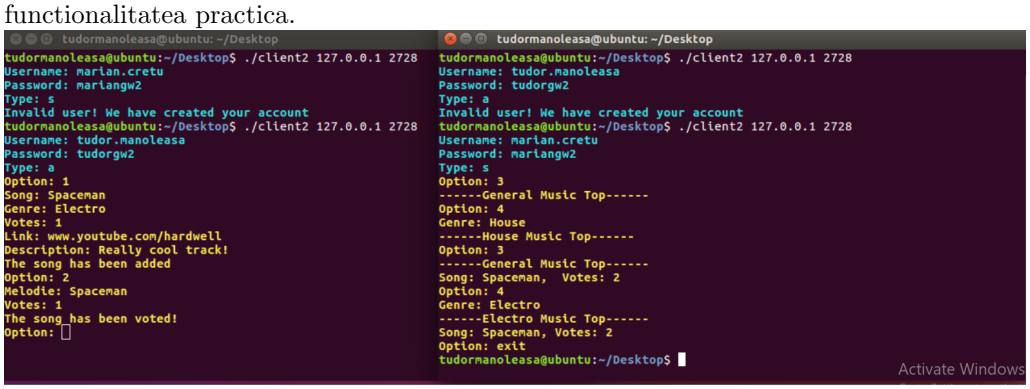


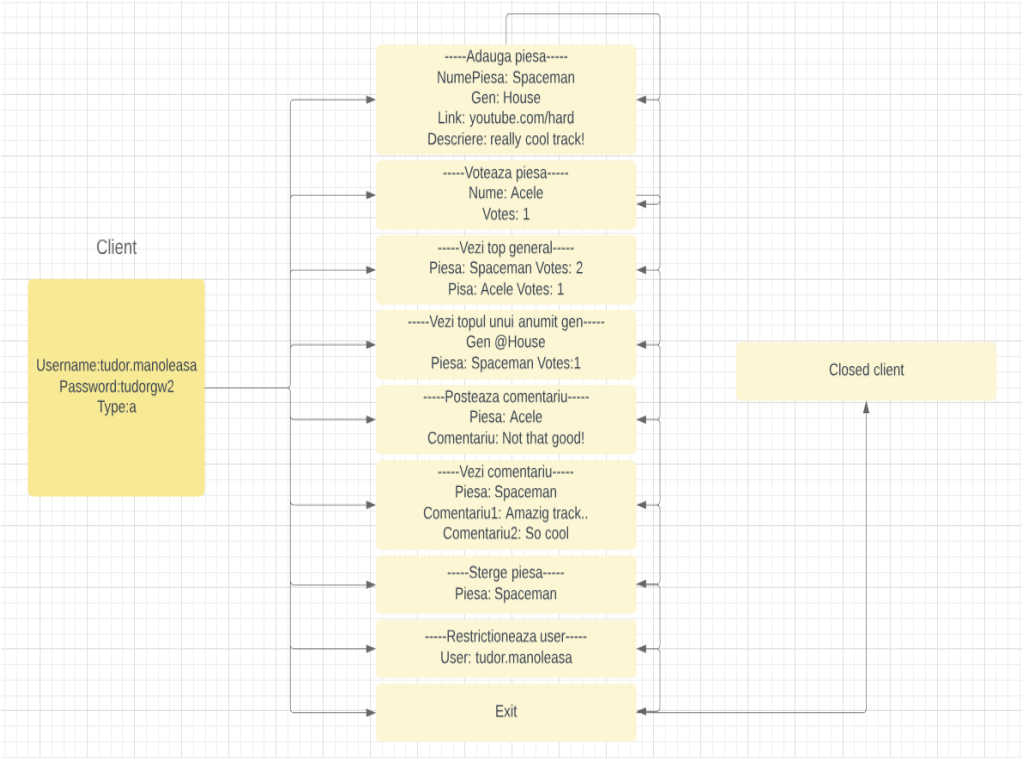
3.2 Diagrama aplicatiei

In urmatoarea imagina este realizata diagrama detaliata a aplicatiei, ce contine *functiile* importante de la baza programului si nu in ultimul rand, *fisierele* la care aceste functii fac referire in cadrul unui apel. Asadar, pentru fiecare optiune introdusa de client, exista un set de functii care vor fi utilizate in vederea reusitei unei comenzi. Toate acestea se afla in *server*.



Clientul nu va face altceva decat sa afiseze o interfata interactiva pe ecran, care il va ghida pe client pas cu pas. Posiblitatea de a efectua oricate comenzi doreste se datoreaza tot unei bucle ***while*** care il va lasa sa tasteze pana la intalnirea cuvantului ***exit***. Clientul nu va utiliza niciodata, in mod direct, fisierele prezentate in diagrama. In acest fel, am facut o separare foarte radicala a atributiilor pe care le are clientul fata de server si vice-versa. Atasez mai jos o imagine ce suprinde functionalitatea practica.




Client
Username:tudor.manoleasa
Password:tudorgw2
Type:a
-----Adauga piesa-----
NumePiesa: Spaceman
Gen: House
Link: youtube.com/hard
Descriere: really cool track!
-----Voteaza piesa-----
Nume: Acele
Votes: 1
-----Vezi top general-----
Piesa: Spaceman Votes: 2
Pisa: Acele Votes: 1
-----Vezi topul unui anumit gen-----
Gen @House
Piesa: Spaceman Votes:1
-----Posteaza comentariu-----
Piesa: Acele
Comentariu: Not that good!
-----Vezi comentariu-----
Piesa: Spaceman
Comentariu1: Amazig track..
Comentariu2: So cool
-----Sterge piesa-----
Piesa: Spaceman
-----Restrictioneaza user-----
User: tudor.manoleasa
Exit
Closed client

Chapter 4

Detalii de implementare

4.1 Structuri de date & Functii predefinite

Principalele structuri de date folosite in acest proiect sunt:
• *Vectorii* - folositi, de exemplu, pentru a tine evidenta numarului de voturi ale unei piese.
• *Matricile* - folosite pentru a retine, de exemplu, piesele din cadrul clasamentului

Pe langa acestea si alte variabile simple, nu am folosit altceva intrucat complexitatea aplicatiei depinde in principal de viteza de procesare a fiecarei operatii introduse de client. Folosirea altor structuri nu ar fi redus cu nimic spatiul alocat de cele descrise mai sus.
Totodata, functiile predefinite din cadrul librariilor **C** au contribuit substantial la reducerea dimensiunii codului. Printre acestea se pot numara ***bzero(void *s, size_t n)***, ce initializeaza n biti ai lui s cu 0, ***htonl/htons***, care au convertit adrese in biti, ***strcat/strcpy/strlen***, ce manipuleaza cu succes sirurile de caractere si bineinteles functiile de ***read/write/connect/bind etc.***, vitale in comunicarea dintre procese.

4.2 Cod relevant

Voi incepe prin a ilustra printr-un sablon modul in care am realizat serverul.

```
// Server
int main()
{
    struct sockaddr_in server;
    server.sin_family = AF_INET;
    server.sin_addr.s_addr = htonl(INADDR_ANY);
    server.sin_port = htons(PORT);
    int sd = socket(AF_INET, SOCK_STREAM, 0);
    bind(sd, (struct sockaddr*)&server, sizeof(struct sockaddr));
    listen(sd,5);
    while(1)
    {
        struct sockaddr_in from;
        int client,length=sizeof(struct sockaddr);
        client = accept(sd, (struct sockaddr*)&from, &length);
        if(fork() == 0)
        {
            close(sd);
            while{ /*-----TRATARE CAZURI-----*/ }
            close(client);
            exit(0);
        }
    }
}
```

In continuare puteti observa un sablon pentru client, adaptat la proiect.

```
//Client
int main ()
{
    struct sockaddr_in server;
    port = atoi (argv[2]);
    server.sin_family = AF_INET;
    server.sin_addr.s_addr = inet_addr(argv[1]);
    server.sin_port = htons (port);

    int sd;
    sd = socket (AF_INET, SOCK_STREAM, 0);

    connect (sd, (struct sockaddr *) &server,sizeof (struct sockaddr));

    read(0,optiune,sizeof(optiune));
    while(1)
    {
        if(strcmp(optiune,"1")==0)
        {
            ............
        }
        if(strcmp(optiune,"2")==0)
        {
            ............
        }
        if(strcmp(optiune,"3")==0)
        {
            ............
        }
        if(strcmp(optiune,"4")==0)
            .
            .
            .
            .
        if(strcmp(optiune,"exit")==0)
                break;
    }
}
```

O alta portiune semnificativa de cod ar putea fi reprezentata de una din functiile principale. Spre exemplu, functia de ***adaugare utilizator***.

```
char* creeazaCont(char user[],char password[],char type[])
{
    char date[256];
    bzero(&date,sizeof(date));
    FILE *f=fopen("users.txt","a+");
    strcpy(date,user);
    strcat(date," ");
    strcat(date,password);
    strcat(date," ");
    strcat(date,type);
    fprintf(f,"%s\n",date);
    fclose(f);
    return " We have created your account";
}
```

Se poate observa, evident, legatura cu fisierul ***users.txt***.

Mai departe, voi atasa alte portiuni de cod din cadrul proiectului.

```
    char* adaugaMelodie(char melodie[], char gen[], char votes[], char link[], char descriere[])
{
    FILE *f=fopen("songs.txt","a+");
    char date[256];
    bzero(&date,sizeof(date));
    strcpy(date,melodie);
    strcat(date,"|");
    strcat(date,gen);
    strcat(date,"|");
    strcat(date,votes);
    strcat(date,"|");
    strcat(date,link);
    strcat(date,"|");
    strcat(date,descriere);
    fprintf(f,"%s\n",date);
    fclose(f);
    return "The song has been added";
}
```

Aici este functia ce face topul general:

```
    void clasamentGeneral()
{
    char sir[256],*p,aux[256];
    int i,j,aux2;
    FILE *f=fopen("songs.txt","rw");
    while(fgets(sir,256,f))
        if(strstr(sir,"|")!=0)
        {
            p=strtok(sir,"|");
            if(!isInMatrix(p))
            {
                n++;
                strcpy(a[n],p);
                d[n]=1;
            }
        }
    fclose(f);
    for(i=1; i<=n; i++)
    {
        f=fopen("songs.txt","rw");
        while(fgets(sir,256,f))
            if(strstr(sir,a[i])!=0 && strstr(sir,"|")==0)
                d[i]++;
        fclose(f);
    }
    for(i=1;i<n;i++)
        for(j=i+1;j<=n;j++)
            if(d[i]<d[j])
            {
                strcpy(aux,a[i]);
                strcpy(a[i],a[j]);
                strcpy(a[j],aux);
                aux2=d[i];
                d[i]=d[j];
                d[j]=aux2;
            }}
```

Posibilitatea de adauga comentarii este exploatata in aceasta functie:

```
    char* adaugaComentariu(char user[],char melodie[],char comentariu[])
{
    FILE *f=fopen("comentarii.txt","a+");
    char sir[256],line[256];
    bzero(&sir,sizeof(sir));
    strcpy(sir,user);
    strcat(sir,": ");
    strcat(sir,melodie);
    strcat(sir,": ");
    strcat(sir,comentariu);

    FILE *f2=fopen("songs.txt","r");
    int ok=0;
    while(fgets(line,256,f2))
        if(strstr(line,melodie))
        {
            ok=1;
            break;
        }
    fclose(f2);

    if(ok==1)
    {
        fprintf(f,"%s\n",sir);
        fclose(f);
        return "Your comment has been added\n";
    }
    else
    {
        fclose(f);
        return "The song is not in our file\n";
    }
}
```

De asemenea, functia de restrictionare a unui user este:

```
    void restriction(char user[])
{
    FILE *f=fopen("restrictii.txt","a+");
    fprintf(f,"%s\n",user);
    fclose(f);
}
```

Chapter 5

Concluzii

In concluzie, solutia propusa in acest raport este una generala, dar care nu sare din vedere nicio cerinta specificata de catre enunt. Elementele de retialistica folosite sunt fundamentale si triviale deoarece am urmarit o implementare cat mai coerenta care sa nu duca la discutii contradictorii cu privire la posibilele modalitati de eficientizare. Exista alte metode de implementare? Da! As fi putut sa folosesc in locul fisierelor o baza de date care sa retina userii si melodiile adaugate pe parcursul operatiilor. De asemenea, o interfata mai atractiva realizata prin intermediul bibleotecii **qt** este un alt *addon* potrivit proiectului. Pe langa acestea, conectarea la un *website* ar fi facut posibila si o portarea in mediul *online*.

Chapter 6

Bibliografie

[1] https://profs.info.uaic.ro/ computernetworks/files/homework2_ro.txt

[2] https://profs.info.uaic.ro/ computernetworks/cursullaboratorul.php

[3] https://www.youtube.com/watch?v=9seb8hddeK4t=1411s

[4] https://www.youtube.com/watch?v=gFAQd0ueIMc

[5] https://stackoverflow.com/questions/20716785/how-do-i-delete-a-specific-line-from-text-file

[6] https://sites.google.com/view/fii-rc/laboratoare

[7] https://meta.stackexchange.com/questions/76902/how-can-i-write-math-formula-in-a-post

[8] https://en.wikipedia.org/wiki/Client–server_model

[9] http://www.networking-forum.com

[10] https://www.lucidchart.com/documents

[11] https://stackoverflow.com/questions/32810981/fork-function-in-c

[12] https://golatex.de

[13] https://intronetworks.cs.luc.edu

[14] https://www.geeksforgeeks.org/socket-programming-cc/

[15] https://www.dipmat.univpm.it/ demeio/public/the_c_programming_language_2.pdf

[16] https://codereview.stackexchange.com/questions/13461/two-way-communication-in-tcp-server