Documentatia proiectului - TopMusic

Straton Elena-Ștefania, grupa A5

6 Ianuarie 2020

1 Introducere

Scopul lucrarii este de a descrie proiectul pe care l-am realizat in limbajul C++, mai exact TopMusic, de nivel B. Acest proiect este o aplicatie client-server ce reda o buna gestionare a unui top muzical in functie de anumite criterii. Aplicatia se adreseaza tuturor utilizatorilor de linux, acestia se pot conecta la aplicatie printr-un nume de utilizator si o parola, iar in cazul in care nu au cont aplicatia le pune la dispozitie o functie prin care se vor inregistra in baza de date. Utilizatori pot fi de 2 feluri: de tip clasic si de tip administrator. Dupa conectare, utilizatorilor li se pune la dispozitie un meniu, dintre care pot alege diferite comenzi. Utilizatori de tip administrator au un meniu mai variat, de unde pot alege functionalitati pe care un utilizator normal nu le poate accesa (de exemplu stergerea unei melodii din top sau restrictionarea votului unui utilizator). In baza de date o melodie va avea asociat un nume, o descriere, un gen muzical, un link catre videoclip (pe YouTube sau alte site-uri) si ultima coloana consta in numarul de voturi pe care il are acea melodie.

2 Tehnologiile utilizate

Am specificat in introducere ca pentru realizarea proiectului am folosit limbajul de programare C++, sub Linux. Un limbaj de programare este un limbaj formal ce contine expresii si reguli valide prestabilite, iar limbajul pe care l-am folosit permite scrierea de programe eficiente, aflandu-se la baza mai multor limbaje dezvoltate ulterior, iar Linux-ul este un sistem de operare orientat pe comenzi, multi-tasking şi multi-user.

O tehnologie utilizata in realizarea acestui proiect este protocolul TCP(Transmission Control Protocol) concurent. Am ales TCP-ul deoarece este un protocol orientat spre conexiune, adica clientul nu se poate conecta daca serverul nu este deschis(deci mereu trebuie sa deschidem serverul prima data), un alt motiv pentru aceasta alegere este faptul ca legatura este realizata si mentinuta pana cand serverul cat si clientul termina de trimis mesajele.

In cazul proiectului TopMusic avem nevoie ca pachetele de date sa ajunga in ordinea specificata initial si din cauza asta am ales TCP(oferă siguranță în transmiterea datelor, nu sunt pierderi de date), iar daca am fi ales UDP acest

lucru nu s-ar fi intamplat deoarece toate pachetele sunt independente unul de celalalt.

Am specificat in al doilea paragral al acestui capitol ca am folosit protocolul TCP concurent. Am asigurat concurenta prin fork pentru a eficientiza execuția programelor. Serverul TCP va crea cate un proces copil (cu ajutorul primitivei fork())pentru fiecare client care doreste sa intre pe el, astfel asigurand posibilitatea conectarii mai multor clienti in acelasi timp.

O alta tehnologie utilizata este SQLite C/C++ care este o interfata prin care server-ul acceseaza o baza de date. Pe aceasta o folosim cu ajutorul librariei $< sqlite3.h >$ ce va fi folosita pentru stocarea datelor necesare intr-un tabel. In cazul aplicatiei TopMusic avem 3 tabele: Users, Melodii si Comentarii. Tabelul Users are 4 coloane: username, password, Admin, vote(ce reprezinta dreptul de a vota). Tabelul Melodii are 5 coloane: nume, descriere, gen (muzical), link (catre videoclip) , numar voturi. Tabelul Comentarii are 3 coloane: username, nume (melodie) si comentariu. In functie de necesitate se vor putea adauga si alte tabele ulterior. Am ales acest tip de stocare de date (si nu XML sau TXT) pentru performanta, deoarece operatiile de citire si scriere sunt rapide.

3 Arhitectura aplicației

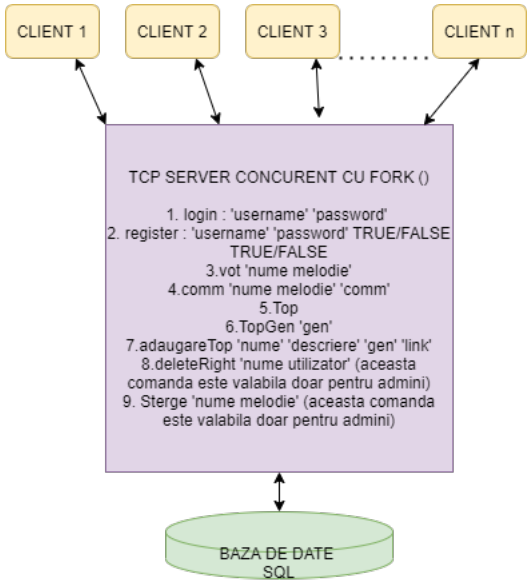

Figure 1: Arhitectura aplicatiei

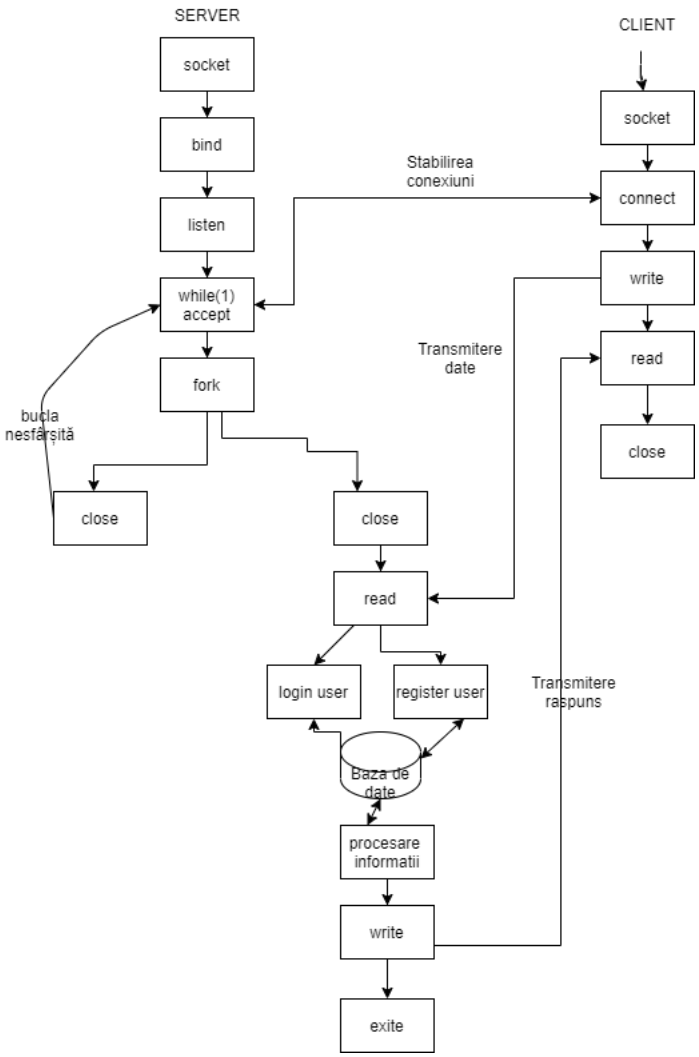

Figure 2: Model Server-Client

In Figura 1 avem reprezentata schema arhitecturi aplicatiei TopMusic. La aceasta aplicatie se pot conecta un numar oarecare de clienti dupa deschiderea serverului. Utilizatori trebuie sa se conecteze printr-un nume si o parola iar daca nu au contul necesar conectari, atunci aplicatia le pune la indemana comanda register prin care isi pot crea contul. Functionalitatile oferite de server sunt (vezi Figure 1):

1.login : 'username' 'password'

2.register: 'username' 'password' 'TRUE/FALSE'(Daca este sau nu admin) 'TRUE/FALSE'(dreptul la vot)

3.vot 'nume melodie'
4.adaugareTop 'numele melodiei' 'descriere' 'gen' 'link'
5.Top /*(in functie de nr de voturi)*/
6.TopGen 'gen muzical'
7.comm 'nume de utilizator' 'comentariu'
8.Stergere 'nume' (aceasta comanda este valabila doar pentru admin)
9.deleteRight 'nume' (aceasta comanda este valabila doar pentru admin)

Se poate observa in figura numarul 2 ca este reprezentat Modelul Server-Client al aplicatiei TopMusic. Serverul TCP al aplicatiei fiind unul concurent, acesta creaza pentru fiecare utilizator un proces copil in asa fel incat va exista posibilitatea servirii mai multor utilizatori in mod simultan si eficient. Acesta va avea rolul de a prelua datele de la client, de ale procesa si ulterior sa transmita rezultatul inapoi la utilizator . Clientul va trebui sa stabileasca un port pentru conexiune (in cazul nostru o a ne conectam cu local host-ul, adica 127.0.0.1 si cu portul 5459). Acesta va trebui sa preia datele introduse de utilizator si sa le transmita la server, iar dupa ce le primeste inapoi, le va afisa pe ecran.

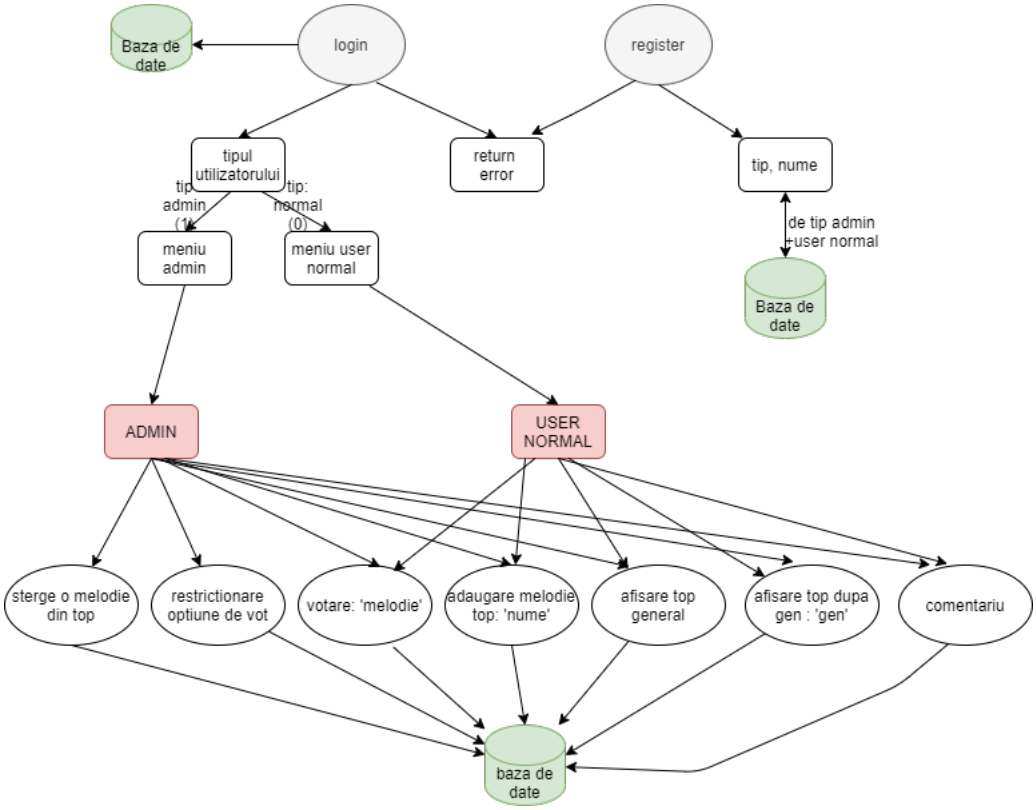

Figure 3: Functionalitatile programului

4 Detalii de implementare

Pentru o utilizare mai buna a aplicatiei, functiile implementate vor trebui sa respecte diagramele de mai sus. Functia "login" are rolul de a introduce clientul in aplicatie, acesta se poate conecta printr-un nume de utilizator si o parola care este deja in baza de date existenta (mai exact tabelul Users). In cazul in care utilizatorul tasteaza gresit numele sau parola, aplicatia va da un mesaj de eroare.

Functia "register" are rolul de a inregistra un utilizator (de orice fel) dupa urmatoarele 4 date: username, password, si 2 valori bool (true/false) care redau dreptul de administrator (TRUE-administrator, FALSE- user normal) si dreptul de vot (TRUE-are drept de vot, FALSE- nu are drept de vot). Daca tipul userului sau parolei este gresit sau numele exista deja in tabela SQL, se va afisa un mesaj de eroare. In cazul in care utilizatorul are deja un cont, el va folosi functia "login". Odata logat cu succes, aplicatia ii va pune userului la dispozitie un meniu divers si se vor putea realiza diverse comenzi.

Cum am specificat si mai sus avem 2 tipuri de useri: normali si administratori. Administratorul fata de un user normal are doua functii in plus: dreptul de a sterge o melodie din top in functie de numele ei si de a restrictiona capacitatea de votare a unui user. Daca administratorul scrie gresit numele melodiei atunci cand sterge melodia din top programul va intoarce un mesaj de eroare. Intr-o coloana din tabela Users numita "Vote" din baza de date unde este reprezentat dreptul user-ului, prin valoarea 1 se va intelege ca userul are voie sa voteze si prin valoarea 0, ca nu are dreptul de a vota nicio melodie.

Meniul care este comun pentru ambele tipuri de utilizatorul constra in 6 functii. Functia uzuala pe care o au majoritatea aplicatiilor este comanda "exit", aceasta functie va putea fi accesata oricand si va deloga orice client care a cerut acest lucru si are sesiune activa. Functia "vot 'nume melodie' " o poate folosi orice tip de users. Daca un client doreste sa voteze o melodie, atunci baza noastra de date, mai exact tabelul Melodii, coloana "nr voturi" din linia cu numele acelei melodi va fi incrementata. In cazul in care numele melodiei este scris gresit programul va da un mesaj de eroare. Functia de "Top" va afisa topul melodiilor si va lua in considerare numarul de voturi al tuturor melodiilor din baza de date, indiferent de genul muzical si va afisat Topul. Comanda "TopGen 'gen muzical' " va cauta in baza de date toate melodiile cu acel gen muzical (pe care il va da utilizatorul) si va compara numarul de voturi al fiecarei melodii si va afisa Topul lor. In cazul in care utilizatorul scrie gresit genul muzical, atunci aplicatia va da un mesaj de eroare referitor la asta. Legatura intre tabele se va realiza prin Join-uri. Functia "comm 'nume melodie' 'comm' " asigura un feedback pozitiv sau negativ al oricarui utilizator al aplicatiei care doreste sa scrie un comentariu legat de o anumita melodie. Acest comentariu poate fi alcatuit din mai multe fraze. De retinut ca dupa fiecare comanda data trebuie sa dam un space si dupa enter.

Mai jos voi pune secvente de cod reprezentative pentru proiect:

```
void login(sqlite3 *db,char user[100],char pass[100])
{
    char *sql=(char*)malloc(1000);
    char *zErrMsg=0;
    //formam interogarea prin care cautam in baza de date existenta username-ului cu parola c
    strcpy(sql,"SELECT * FROM Users WHERE username=");
    strcat(sql,user);
    strcat(sql," AND password=");
    strcat(sql,pass);
    strcat(sql,";");
    //afisam introgarea
    printf("INTEROGAREA : %s",sql);

    int rc = sqlite3_exec (db, sql, CheckIfExistsUser, 0, &zErrMsg);

    if (rc != SQLITE_OK)
    {
     fprintf (stderr,"SQL error: %s\n", zErrMsg);
     sqlite3_free (zErrMsg);
    }
   sqlite3_close(db);
}
```

Figure 4: Autentificare clienti

```
void comm(sqlite3 *db, char melodie[100],char comm[1000],char User[100],char mesajInapoi[1000])
{
   char *sql=(char *) malloc(1000);
   char *mesajEroare;
   //aceasta operatii reprezinta o inserare in baza de date
   //exemplu: INSERT INTO Comentarii VALUES('Bogdan','Arunca-ma','Delia esti geniala!');
   strcpy(sql,"INSERT INTO Comentarii VALUES(");
   strcat(sql,User);
   strcat(sql,",");
   strcat(sql,melodie);
   strcat(sql,",");
   strcat(sql,comm);
   strcat(sql,");");
   cout << "adaugare: " << sql << endl;
   int rc ;
   rc = sqlite3_open("StefaniaDataBase.db",&db);//pentru a realiza o operatie cu baza de date, mai intai trebuie sa o de
   rc = sqlite3_exec(db,sql,NULL,0,&mesajEroare);
   if(rc !=SQLITE_OK)
   {
     strcpy(mesajInapoi,"\n\n Ne pare rau , a aparut o problema ! Incercati sa scrieti toate datele corect! \n\n");
     fprintf(stderr,"SQL error: %s\n",mesajEroare);
     sqlite3_free(mesajEroare);
  }else {
        strcpy(mesajInapoi, "\n\nComentariul a fost adaugat! \n\n");
       }
  sqlite3_close(db);// inchidem baza de date
}
```

Figure 5: Functia prin care un user adauga un comentariu la o anumita melodie

In figura 4 avem un screenshot cu functia de login, aceasta este pentru autentificarea oricarui tip de utilizator. In aceasta functie am format interogarea care preia username-ul si parola pe care o da clientul din consola si sa le caute in baza de date deja existenta, ca in SQL ("SELECT * FROM Users WHERE

username=user AND password=pass;"). Initial iau un pointer pe care il intitulez "sql" (de tip char) si ii aloc memorie pentru a reusi sa pun in aceasta toate caracterele pentru interogare. Linia in care regasim funcția sqlite3 exec() evaluează instrucțiunea SQL, mentionez ca am folosit si o functie de callback numita "CheckIfExistUser"(care extrage anumite informatii din baza de date, mai exact daca exista sau nu acel user). Pentru a ne da seama cine este un user normal sau admin ne ajutam de o functie intermediana numita ("IsAdminThisUser").

In fingura 5 avem functia prin care orice tip de user poate adauga un comnetariu doar daca este logat la sistem. Aceasta operatie reprezinta o inserare in baza de date in tabla Comentarii (exemplu: INSERT INTO Comentarii VALUES('Bogdan','Arunca-ma','Delia esti geniala!');). Initial luam un pointer pe care il intitulez "sql" (de tip char) si ii aloc memorie pentru a reusi sa pun in aceasta toate caracterele pentru interogare. In aceasta functie formam interogarea prin care reusit sa inseram valoriile date de user. In pointerul deja format punem prima parte a inserari, iar dupa aceea o sa fie puse in aceasta variabila cele 3 valori necesare pentru inserare, mai exact numele userului, numele melodiei si comentariul, precum in screenshot. De data aceasta nu a mai fost nevoie de o functie de callback deoarece nu am extras informatii din tabele.

```
//functia prin care un membru al aplicatiei poate vota
void votareMelodie(sqlite3 *db, char melodie[100],char answerBack[1000])
{
   //aceasta operatie reprezinta un update in baza de date, mai exact trebuie
   //sa incrementam din tabela Melodii ,coloana cu numar_voturi, linia in care sa afla melodia respectiva si sa o incrementam
   //exemplu: UPDATE Melodii SET numar_voturi=numar_voturi+1 WHERE trim(nume)='Arunca-ma';
  char *sql=(char*)malloc(1000);
  char *mesajEroare;
  strcpy(sql,"UPDATE Melodii SET numar_voturi=numar_voturi+1 WHERE trim(nume)=");
  strcat(sql,melodie);
  strcat(sql,";");
  cout << sql << endl;

  int rc ;
  rc = sqlite3_open("StefaniaDataBase.db",&db);
  rc = sqlite3_exec(db,sql,NULL,0,&mesajEroare);
  if(rc !=SQLITE_OK)
  {
    strcpy(answerBack,"\n\n Ne pare rau , a aparut o problema ! \n\n");
    fprintf(stderr,"SQL error: %s\n",mesajEroare);
    sqlite3_free(mesajEroare);
  }else {
         strcpy(answerBack, "Votarea melodiei s-a realizat cu succes! \n");
        }
  sqlite3_close(db);
}
```

Figure 6: Functia prin care un user voteaza

In figura 6 este redata functia prin care un membru poate vota. Aceasta operatie reprezinta un update in baza de date, mai exact trebuie sa incrementam din tabela Melodii ,coloana cu numar voturi, linia in care sa afla melodia respectiva (cea data de utilizator) si sa ii adaugam valoarea 1. In aceasta functie folosim aceasi metoda cu pointerul si formarea interogatiei.

In figura 7 este reprezentata functia care afiseaza topul general in functie de numarul de voturi. Aceasta functie reda o fraza select din tabela Melodii ce

```
//functia care afiseaza topul general in functie de numarul de voturi
//aceasta functie reda o fraza select din tabela Melodii ce aseaza in ordine descrescatoare dupa nr de voturi toate melodiile
void TopMusic(sqlite3 *db,char mesajInapoi[1000])
{
    bzero(TopMelodii,2000);
    char *sql=(char*)malloc(1000);
    char *zErrMsg=0;
    strcpy(sql,"SELECT nume,numar_voturi FROM Melodii ORDER BY numar_voturi DESC;");

   int rc ;
   rc = sqlite3_open("StefaniaDataBase.db",&db);
   rc = sqlite3_exec(db,sql,CheckIfTop,0,&zErrMsg);
   if(rc !=SQLITE_OK)
     {
         strcpy(mesajInapoi,"\n\n Ne pare rau , a aparut o problema ! \n\n");
         fprintf(stderr,"SQL error: %s\n",zErrMsg);
         sqlite3_free(zErrMsg);
     }else{
            strcpy(mesajInapoi,TopMelodii);
           }
  sqlite3_close(db);

}
```

Figure 7: Functia prin care se afiseaza Topul general

```
}
static int CheckIfTop (void *str, int argc, char **argv, char **azColName)//functia de callback
{

    strcat(TopMelodii,argv[0]);
    strcat(TopMelodii," ");
    strcat(TopMelodii,argv[1]);
    strcat(TopMelodii," ");
    strcat(TopMelodii,"\n");
   // printf("%s",argv[0]);
    return 0;
}
```

Figure 8: Functia de callback pentru Top

extrage doar 2 campuri(nume, numar de voturi) si aseaza in ordine descrescatoare dupa numarul de voturi toate melodiile. Pentru aceasta functionalitate am folosit o functie de callback pentru a extrage informatiile din cele 2 campuri. "argv[0]" reprezinta coloana cu numele melodiilor, iar "argv[1]" reprezinta coloana cu numarul de voturi ale foecarei melodii.

Pentru a reda modul in care un utilizator gestionateaza programul se foloseste o diagrama Use Case (Figura 5). Aceasta are rolul de a prezenta functionalitatiile pe care le ofera aplicatia utilizatorului si raspunsurile acestora. Pentru a reda o buna functionalitate a acestei aplicatii vom lua 2 scenarii unul de succes si unul de esec.

Fiecare scenariu de succes incepe cu logare sau inregistrare, deci primul scenariu de succes o sa presupunem ca un utilizator normal se va conecta la aplicatie si dupa va accesa diferite comenzi de care ii sunt puse in meniu.

Scenariul de esec este reprezentat prin faptul ca utilisatorul nu se logeaza initial si el incearca din rasputeri sa acceseze comenzi de tipul : votare melodie, afisare top, etc ceea ce nu este posibil si se vor afisa doar mesaje de eroare.

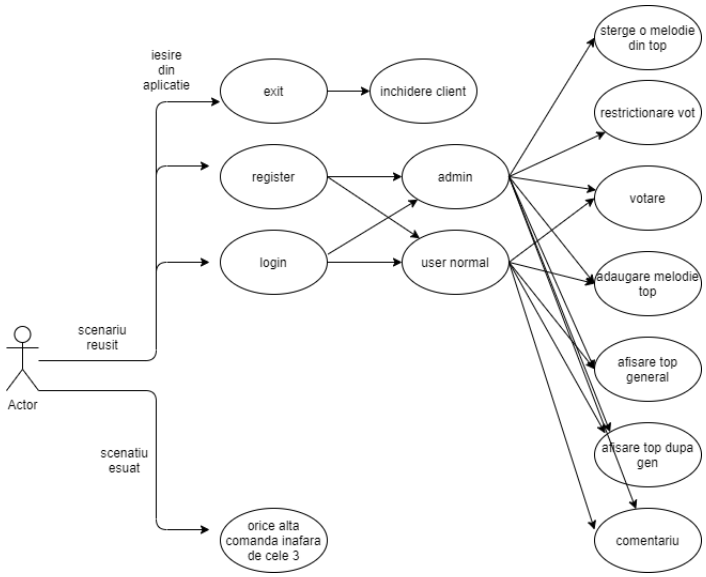

Figure 9: Autentificare clienti

5 Concluzii

Aplicaţia pe care am terminato constituie pentru mine un mod bun prin care am învaţat si aprofunda cum funcţionează comunicarea dintre Server si Client. Proiectul poate fi îmbunătaţit mai ales pe partea de client care ar putea oferi o interfaţă grafica si atractivă, o alta imbunatatire ar putea fi efectuata la comanda de votare, ar trebui modificata in asa fel incat un user sa voteze o singura data o anumita melodie. Totodata o alta imbunatatire ar fi in momentul cand avem mai multe melodii cu acelasi numar de voturi, acestea sa fie afisate in top dupa un anumit criteriu.

6 Bibliografie

References

[1] Curs 5 Rețele de Calculatoare
https://profs.info.uaic.ro/∼computernetworks/files/5rc$_{P}rogramareaInReteaI_{r}o.pdf$

[2] Curs 7 Rețele de Calculatoare
https://profs.info.uaic.ro/∼computernetworks/files/7rc$_{P}rogramareaInReteaIII_{E}n.pdf$

[3] Baze de Date
http://zetcode.com/db/sqlitec/

[4] Model diagramă
https://drive.google.com/file/d/1Zso1PTE6RBATLZ6MzaXfUwFcpEncYL8R/view

[5] SQLite
https://www.sqlite.org/index.html

[6] TCP/UDP
https://www.linkedin.com/pulse/concurrent-udp-servers-jorge-frisancho/
http://intronetworks.cs.luc.edu/1/html/udp.html

[7] Limbaj de programare
https://ro.wikipedia.org/wiki/Limbaj-de-programare