Tema 2 - Raport TopMusic

Octavian Iacob - Anul 2, Grupa A3

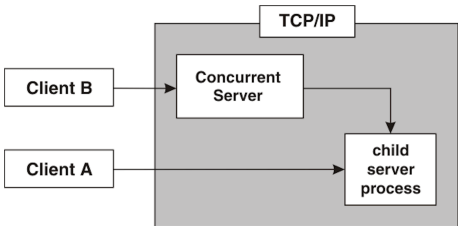
1 Introducere

Acest raport vizează proiectul TopMusic, acesta având gradul B de dificultate. Proiectul constă într-o aplicație de tip client-server, scrisă în limbajul de programare C. Aplicația are rolul de a crea și de a manageriza un top muzical, în care sunt prezente mai multe funcționalități: un sistem de înregistrare/login, atât pentru utilizatorii normali precum și pentru adminii aplicației.

Fiecare utilizator va avea funcții și acces la unele componente ale aplicației în funcție de gradul său de acces la aplicație. Ca o masură de securitate, funcționalitățile aplicației nu pot fi accesate dacă un utilizator nu este logat. Administratorul poate șterge melodii din top, poate șterge comentarii și poate revoca dreptul de vot al unui utilizator. Un utilizator normal poate adăuga melodii în top, poate vota (dacă acest drept nu i-a fost revocat), poate să posteze comentarii la fiecare melodie și poate vedea topul muzical sortat după diverse criterii.

2 Tehnologii utilizate

Aplicația folosește un model client/server orientat conexiune bazat pe protocolul TCP. Am ales protocolul TCP datorită calității maxime a serviciilor, integrarea mecanismelor de stabilire - eliberare a conexiunii și a controlului fluxului de date. Implementarea serverului este concurentă, astfel serverul poate procesa simultan mai multe cereri de la clienți diferiți. După stabilirea conexiunii între client și server, serverul preia comenzile de la client, le proceseaza și întoarce înapoi clientului rezultatele. Concurența se realizează prin alocarea fiecărui client a unui proces copil al serverului.



Stocarea datelor aplicației se face printr-o bază de date de tip SQLITE. Am ales acest sistem relațional de gestionare a bazelor de date datorită vitezei, configurării ușoare, compatibilității bune cu C/C++ și a documentației detaliate. În aceasta se regăsesc următoarele tabele: admin, user, songs, genres și comments.

În tabela admin (și similar pentru tabela user) se stochează datele de logare a fiecărui administrator(user). În tabela songs se memorează pentru fiecare melodie titlul, artistul, o scurtă descriere opțională, unul sau mai multe genuri muzicale, numărul de voturi și un link către YouTube/Spotify sau altă platformă similară. În tabela genres se regăsesc genurile muzicale în care pot fi categorizate melodiile.

3 Arhitectura aplicației

Structura bazei de date

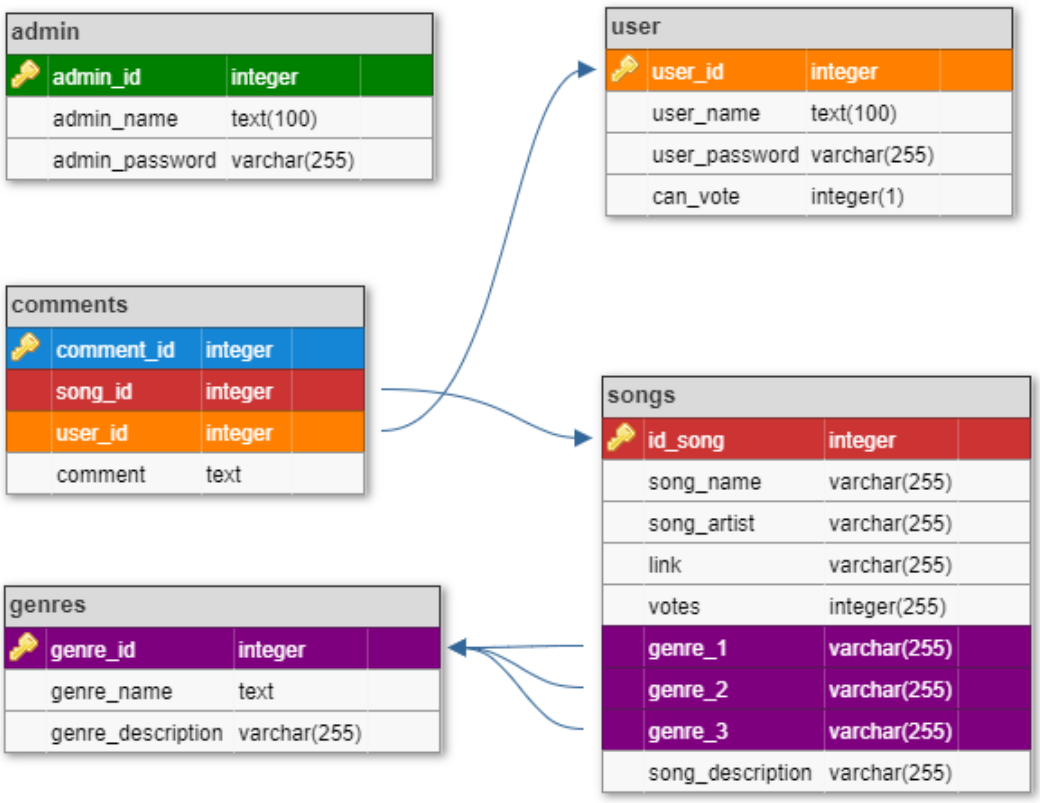


Diagrama aplicației

4 Detalii de implementare

Odată cu conectarea clientului la server, utilizatorul este întâmpinat cu un meniu ce îi prezintă 3 opțiuni: 1) Înregistrare ca utilizator nou, 2) Logare ca utilizator sau ca admin, 3) deconectarea de la server.

Înregistrarea

Dacă utilizatorul decide sa se inregistreze, acestuia i se vor cere un nume și o parola. După ce aceste date sunt primite, sunt trimise la server pentru a fi validate. Se verifica numele dat în tabela user din baza de date. În caz în care deja exista un utilizator cu acest nume, serverul trimite la client un avertisment, și se intrerupe procesul de inregistrare. În caz contrar, inregistrarea are loc cu succes. Se trimite un mesaj ce confirma validitatea datelor introduse inapoi la client, și se introduc efectiv informatiile în baza de date. Prin procesul de inregistrare, se pot crea doar utilizatori normali.

Logarea

Dacă utilizatorul introduce comanda "login", acestuia i se cer numele și parola. După ce sunt introduse de la tastatură, odată ce sunt primite pe server sunt validate. Mai întâi se verifică datele în tabela de admini. Dacă se găsesc datele în tabela, atunci utilizatorul va fi logat ca admin. Dacă nu se găsesc în tabela admin, vor fi verificate și în tabela user. Dacă se găsesc acolo, atunci utilizatorul va fi logat ca un utilizator normal. Dacă nu se găsesc în nici o tabela, este returnat un mesaj de eroare.

După logarea cu succes, se prezintă comenzi noi disponibile pentru utilizator. Dacă un admin s-a logat, acesta va avea în plus alte funcții disponibile față de un utilizator normal. Odată logat, nu se mai pot accesa funcțiile de register și de login.

Adaugarea unei melodii

Aceasta funcție este disponibilă atât utilizatorilor normali, cât și administratorilor. La apelarea comenzii "addSong" utilizatorului i se vor cere introducerea mai multor date despre melodia pe care vrea să o introduca: denumirea, artistul, un URL și unul sau mai multe genuri muzicale din care aparține melodia respectivă. Utilizatorul va trebui să aleaga genul muzical dintr-o listă prestabilită de administratori.

```
void addSong(char song_name[], char song_artist[], char song_link[],
    char genre1[], char genre2[], char genre3[])
{
   rc = sqlite3_open("topmusic.db", &database);
   if (rc)
```

```
      printf("Error opening database.\n");
   else
      printf("Database opened successfully.\n");
   asprintf(&query, "insert into songs (song_name, song_artist, link,
       genre1, genre2, genre3, votes, song_description) values (\"%s\",
       \"%s\", \"%s\", \"%s\", \"%s\", \"%s\", 0, \"\");", song_name,
       song_artist, song_link, genre1, genre2, genre3);
   printf("SQL: '%s'\n", query);
   sqlite3_prepare_v2(database, query, strlen(query), &statement, NULL);
   rc = sqlite3_step(statement);
   if (rc != SQLITE_DONE)
      printf("ERROR inserting data: %s\n", sqlite3_errmsg(database));
   else
      printf("Song inserted successfully.\n");
   sqlite3_finalize(statement);
   free(query);
   sqlite3_close(database);
}
// --- FUNCTIE ADD SONG -----

else if (strcmp(input, "addSong") == 0)
{
    char song_name[BUF], song_artist[BUF], song_link[BUF], genre1[BUF],
        genre2[BUF], genre3[BUF];

   read(client, song_name, BUF); // citeste song_name de la client (1);
   printf("Song name is: %s \n", song_name);
   read(client, song_artist, BUF); // citeste song_artist de la client
       (2);
   printf("Song artist is: %s \n", song_artist);
   read(client, song_link, BUF); // citeste link de la client (3);
   printf("link is: %s \n", song_link);

   char genres[BUF];
   showGenres(genres);
   write(client, genres, BUF); // trimite la client genres (4)

   read(client, genre1, BUF); // citeste genre1 de la client (5)
   printf("Song genre1 is: %s \n", genre1);
   if (isValidGenre(genre1) == 1)
   {
      bzero(output, BUF);
      strcat(output, "valid");
      write(client, output, BUF); // scrie valid la client pt genre1 (6)
      bzero(output, BUF);

      read(client, genre2, BUF); // citeste genre2 de la client (7)
      printf("Song genre2 is: %s \n", genre2);

      if (isValidGenre(genre2) == 1 || strcmp(genre2, "") == 0)
```

```
        {
            bzero(output, BUF);
            strcat(output, "valid");
            write(client, output, BUF); // scrie valid la client pt genre2
                (8)
            bzero(output, BUF);

            read(client, genre3, BUF); // citeste genre3 de la client (9)
             printf("Song genre3 is: %s \n", genre3);

            if (isValidGenre(genre3) == 1 || strcmp(genre3, "") == 0)
            {
                bzero(output, BUF);
                strcat(output, "valid");
                write(client, output, BUF); // scrie valid la client pt
                    genre2 (10)
                bzero(output, BUF);
                printf("The genres are: %s , %s, %s \n", genre1, genre2,
                    genre3);
                addSong(song_name, song_artist, song_link, genre1, genre2,
                    genre3);
                printf("Song inserted successfully.\n");
            }
            else // eroare la genre3
            {
                bzero(output, BUF);
                strcat(output, "Invalid at genre 3");
                write(client, output, BUF); // scrie invalid la client pt
                    genre3 (10)
                bzero(output, BUF);
            }
        }
        else // eroare la genre2
        {
            bzero(output, BUF);
            strcat(output, "Invalid at genre 2");
            write(client, output, BUF); // scrie invalid la client pt
                genre2 (8)
            bzero(output, BUF);
        }
    }
    else // eroare la genre1
    {
        bzero(output, BUF);
        strcat(output, "Invalid at genre 1");
        write(client, output, BUF); // scrie invalid la client pt genre1
            (6)
        bzero(output, BUF);
    }
}
```

Afișarea tuturor melodiilor

Această funcție este disponibilă tuturor utilizatorilor. Se realizează o interogare pe baza de date, iar output-ul ei este salvat într-un șir de caractere, într-o formă de tip tabel. Acest output este trimis la client și apoi afișat.

Stergerea unei melodii

Această funcție este disponibilă doar administratorilor. Mai întâi i se prezintă administratorului o listă cu toate melodiile (metodă descrisă în secțiunea anterioară). Acesta introduce ID-ul melodiei pe care dorește să o șteargă. Acest ID este trimis la server, iar serverul execută o comandă SQL de tip DELETE pe baza de date, luând ca parametru ID-ul dat.

Adăugarea unui gen muzical

Această funcție este disponibilă doar administratorilor. La apelarea comenzii "addGenre" administratorului i se cere introducerea numelui genului și o descriere opțională despre acesta. Datele sunt trimise la server, apoi serverul le introduce în baza de date în tabela genres.

```
void addGenre(char genre_name[], char genre_description[])
{
   rc = sqlite3_open("topmusic.db", &database);
   if (rc)
      printf("Error opening database.\n");
   else
      printf("Database opened successfully.\n");
   asprintf(&query, "insert into genres (genre_name, genre_description)
       values (\"%s\", \"%s\");", genre_name, genre_description);
   printf("SQL: '%s'\n", query);
   sqlite3_prepare_v2(database, query, strlen(query), &statement, NULL);
   rc = sqlite3_step(statement);
   if (rc != SQLITE_DONE)
      printf("ERROR inserting data: %s\n", sqlite3_errmsg(database));
   else
      printf("Genre inserted successfully.\n");
   sqlite3_finalize(statement);
   free(query);
   sqlite3_close(database);
}

// --- FUNCTIE ADD GENRE -----
else if (strcmp(input, "addGenre") == 0)
```

```
{
   char genre_name[BUF], genre_description[BUF];
   read(client, genre_name, BUF); // citeste genre name de la client (1);
   printf("Song name is: %s \n", genre_name);
   read(client, genre_description, BUF); // citeste genre description de
       la client (2);
    printf("Song artist is: %s \n", genre_description);
   if (strcmp(genre_description, "") == 0)
      strcat(genre_description, " ");
   addGenre(genre_name, genre_description);
}
```

Afișarea genurilor muzicale

Această funcție este accesibilă atât utilizatorilor precum și administratorilor. La apelarea ei se interoghează baza de date în tabela genres. Rezultatul este salvat într-un string, acesta fiind după aceea trimis înapoi la client pentru a fi afișat.

Ștergerea unui gen muzical

Această funcție este disponibilă doar administratorilor. Mai întâi i se prezintă administratorului o listă cu toate genurile muzicale (metodă descrisă în secțiunea anterioară). Acesta introduce ID-ul genului pe care dorește să îl șteargă. Acest ID este trimis la server, iar serverul execută o comandă SQL de tip DELETE pe baza de date, luând ca parametru ID-ul dat.

Adăugarea / ștergerea unei descrieri pentru o melodie

Această funcție este disponibilă doar administratorilor. La apelarea comenzii "addDescription" administratorului i se prezintă lista tuturor melodiilor și se cere introducerea unui ID, apoi a unei descrieri. Aceste 2 date sunt trimise la server și sunt date ca parametri în comanda UPDATE de tip SQL. Pentru ștergere se apelează comanda "deleteDescription". Funcționează într-un mod similar cu adăugarea descrierii, doar că nu se mai cere utilizatorului o descriere, ci este setată automat cu un șir de caractere vid.

Afișarea unei descrieri pentru o melodie

Această funcție este disponibilă tuturor utilizatorilor. Se prezintă pe partea de client o lista cu toate melodiile și se cere un ID de la utilizator. Odată furnizat ID-ul, acesta este trimis la server pentru a fi dat ca parametru într-o interogare SQL. Rezultatul interogării este salvat sub tipul string și este trimis înapoi la client pentru a fi afișat. Dacă lungimea descrierii este mai mică decât 1, se consideră că melodia respectivă nu are o descriere, și se afișează un mesaj corespunzător.

Deschiderea unui link al unei melodii

Funcția este disponibilă tuturor utilizatorilor. La apelarea comenzii "openLink" se prezintă lista cu toate melodiile. Utilizatorul introduce ID-ul melodiei pentru care dorește să acceseze link-ul. Serverul interoghează baza de date și trimite înapoi la client link-ul melodiei. Link-ul trebuie să conțină "http://" pentru a fi considerat valid. Se efectuează un apel de sistem pe partea clientului ce deschide URL-ul primit în browser.

Adăugarea unui comentariu

Funcția este disponibilă tuturor utilizatorilor. La apelarea comenzii "addComment" se prezintă lista cu toate melodiile. Utilizatorul introduce ID-ul melodiei pentru care dorește să scrie un comentariu, urmat de comentariul său. Aceste date sunt trimise la server și se creează o nouă linie în baza de date cu ID-ul melodiei, ID-ul utilizatorului precum și comentariul respectiv care primește un ID.

Afișarea comentariilor pentru o melodie

Această funcție este disponibilă tuturor utilizatorilor. Se prezintă pe partea de client o lista cu toate melodiile și se cere un ID de la utilizator. Odată furnizat ID-ul, acesta este trimis la server pentru a fi dat ca parametru într-o interogare SQL. Rezultatul interogării este salvat sub tipul string și este trimis înapoi la client pentru a fi afișat.

Ștergerea unui comentariu

Această funcție este disponibilă doar administratorilor. Mai întâi i se prezintă administratorului o listă cu toate comentariile de la toate melodiile. Acesta introduce ID-ul comentariului pe care dorește să îl șteargă. Acest ID este trimis la server, iar serverul execută o comandă SQL de tip DELETE pe baza de date, luând ca parametru ID-ul dat.

Afișarea topului în funcție de un gen muzical

Această funcție este disponibilă tuturor utilizatorilor. La apelarea comenzii "topByGenre", serverul trimite la client lista de genuri muzicale disponibile. Utilizatorul introduce numele genului pe care îl dorește. Serverul executa o interogare SQL și utilizează genul primit de la client ca parametru. Se trimite la client pentru a fi afișată o listă de melodii care au unul din genurile ei genul ales de utilizator.

Afișarea topului în funcție de numarul de voturi

Această funcție este disponibilă tuturor utilizatorilor. Funcția apelata este "topByVotes" și este foarte similară cu "showSongs", singura diferență fiind adăugarea unei condiții WHERE în interogarea SQL. Se realizează o interogare pe baza de date, iar output-ul ei este salvat într-un șir de caractere, într-o formă de tip tabel. Acest output este trimis la client și apoi afișat.

Afișarea tuturor utilizatorilor

Această funcție este disponibilă doar administratorilor. La apelarea comenzii "showUsers", serverul trimite la client lista tuturor utilizatorilor din tabela user.

Modificarea drepturilor de vot ale unui utilizator

Această funcție este disponibilă doar administratorilor. La apelarea comenzii "setVoteRight", administratorului I se prezintă lista tuturor utilizatorilor împreună cu ID-urile lor. Acesta alege ID-ul utilizatorului căruia dorește să îi schimbe dreptul de vot. Apoi poate scrie 1 pentru a oferi drept de vot, sau 0 pentru a interzice votul pentru acel utilizator.

Votarea

Această funcție este disponibilă tuturor utilizatorilor. La apelarea comenzii "vote", serverul verifică dacă utilizatorul curent are drept de vot. Dacă are, atunci programul trimite la client o lista cu toate melodiile. Utilizatorul scrie ID-ul melodiei pe care dorește să o voteze. ID-ul este trimis la server, iar prin SQL se incrementează numărul de voturi pentru melodia cu ID-ul respectiv.

Promovarea din utilizator în administrator

Această funcție este disponibilă doar administratorilor. La apelarea comenzii "makeAdmin", se afișează la client lista tuturor utilizatorilor. Administratorul alege un ID al user-ului pe care dorește sa îl promoveze. Acest user este trimis la server. Serverul face o interogare în baza de date pe baza ID-ului primit și salvează temporar username-ul și parola corespunzătoare ID-ului, după care șterge din tabela user utilizatorul respectiv. După aceea creează o linie noua în tabela admin cu datele salvate temporar.

5 Concluzii

Aplicația poate fi îmbunătățită prin adăugarea unei interfețe grafice, pentru o experienta mai placută. De asemenea, mai poate fi îmbunătățită calitatea codului prin înlocuirea unor linii de cod repetitive, mai ales în interacțiunile cu baza de date. Printre alte îmbunătățiri se pot număra un sistem de recuperare de parolă, criptarea parolelor sau un istoric al tuturor comenzilor făcute pe server și de către cine au fost făcute.

6 Bibliografie

https://sites.google.com/view/fii-rc
https://profs.info.uaic.ro/ computernetworks/
https://www.sqlite.org/cintro.html
http://www.wassen.net/sqlite-c.html
http://zetcode.com/db/sqlitec/
https://www.geeksforgeeks.org/sql-using-c-c-and-sqlite/
https://www.linuxjournal.com/content/accessing-sqlite-c
https://www.geeksforgeeks.org/socket-programming-cc
https://wiki.archlinux.org/index.php/Bash/Prompt$_customization$
$https://stackoverflow.com$