ReadsProfiler

Iulian Oleniuc (B3)

1 Introducere

ReadsProfiler este o aplicație minimalistă, de tip client-server, care oferă acces la o librărie online. Librăria conține cărți din diverse genuri și subgenuri, având facilități ce permit explorarea acestora într-un mod simplu și intuitiv.

Cărțile pot fi căutate după criterii precum ISBN, titlu, autor, gen, an și rating. De asemenea, clientul poate explora ierarhia de genuri și subgenuri, precum și lista de genuri abordate de un anumit autor. Nu în ultimul rând, sistemul îi oferă utilizatorului sugestii de noi cărți pe baza căutărilor și descărcărilor sale.

2 Tehnologii utilizate

2.1 TCP/IP

Aplicația utilizează tehnologia TCP/IP (Transfer Control Protocol/Internet Protocol) pentru a crea o conexiune full-duplex fiabilă între client (utilizator) și server (librărie). Aceasta se bazează pe un schimb de mesaje de tipul cerere-răspuns. Protocolul TCP/IP asigură faptul că datele vor ajunge la destinație intacte și în ordinea în care au fost trimise.

2.2 Socket-uri

Comunicarea dintre client și server se realizează prin intermediul socket-urilor. Un socket este un punct terminal de comunicare între două noduri ale rețelei. Fiecare astfel de punct este determinat de o adresă IP și un port TCP. Acest proiect folosește socket-uri BSD, cele oferite de limbajul C.

2.3 SQLite și Qt

Pentru a facilita stocarea și accesarea datelor, aplicația folosește o bază de date relațională, a cărei structură este detaliată în secțiunea 3.2. Aceasta cuprinde date despre utilizatori, cărți, autori și genuri. Ca sistem de gestiune a bazelor de date am ales SQLite, deoarece este rapid, cross-platform și necesită zero configurare. De asemenea, aplicația folosește biblioteca Qt, deoarece aceasta oferă un API C++ modern pentru interacțiunea cu SQLite.

3 Arhitectura aplicației

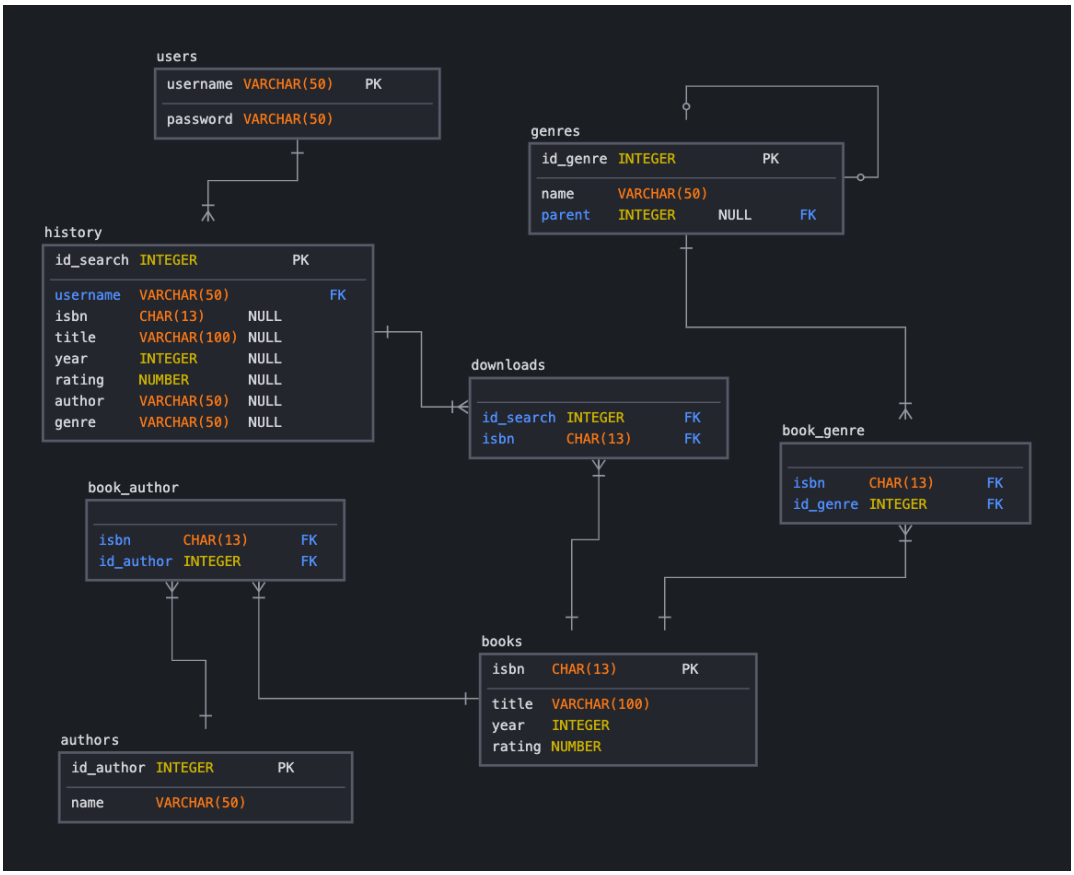
3.1 Interfața grafică (CLI)

Aplicația ReadsProfiler rulează în terminal, însă dispune de o interfață pseudo-grafică atractivă. În acest sens, navigarea prin meniuri/cărți se realizează folosind săgețile, precum și tastele `ENTER` și `BACKSPACE`. Mai jos puteți observa câteva capturi de ecran cu meniul principal, pagina de recomandări și ecranul *Explore Authors*.

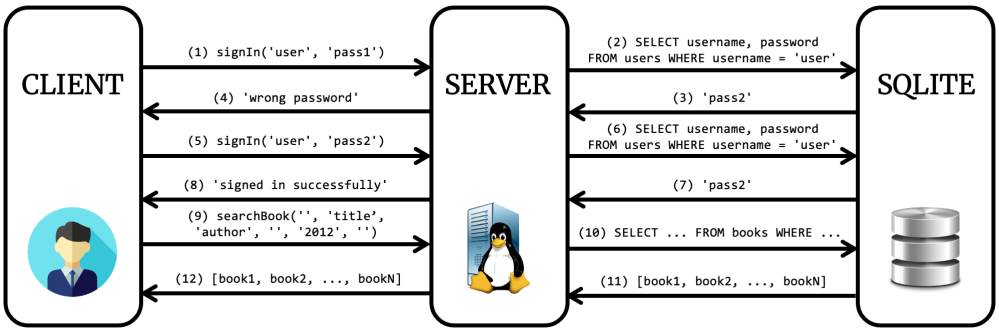
3.2 Stocarea datelor

Datele despre utilizatori, autori și cărți sunt stocate într-o bază de date relațională, conform schemei de mai jos. Tabelele `books`, `authors`, `genres` rețin date despre cărți, autori și respectiv genuri. Ierarhia de genuri și subgenuri este modelată de atributul `parent` al tabelei `genres`, care reține id-ul genului părinte, dacă acesta există.

Tabela `book_author` modelează relația dintre o carte și autorii săi, iar `book_genre` relația dintre o carte și genurile în care se încadrează. Tabela `history` conține istoricul căutărilor utilizatorilor, împreună cu parametrii acestora. De asemenea, tabela `downloads` conține, pentru fiecare căutare din `history`, lista cărților descărcate de către utilizator în urma căutării respective.



3.3 Scenariu de utilizare



4 Detalii de implementare

4.1 Realizarea concurenței

Servirea concurentă a clienților este realizată prin thread-uri:

```
vector<pthread_t> threads;
while (true) {
    cout << "[server] listening at port " << PORT << '\n';
    sockaddr_in client;
    bzero(&client, sizeof(client));
    socklen_t size = sizeof(client);
    Thread *thread = new Thread;
    thread->thread = threads.size();
    thread->client = accept(sd, (sockaddr*)&client, &size);
    threads.emplace_back();
    pthread_create(&threads.back(), nullptr, &treat, thread);
}
```

4.2 Protocolul de comunicare

Pachetele trimise de la client la server (și reciproc) au următoarea structură:

1. `length`: lungimea payload-ului
2. `type`: tipul pachetului
3. `payload`: șirul de caractere util

Valorile pe care le poate lua `type` sunt:

type	semnificație
si	sign-in
sb	search book
db	download book
ud	update downloads
ml	you may like
eg	explore genres
ea	explore authors
so	sign-out

Formatul `payload`-ului diferă în funcție de valoarea lui `type`. În cazul pachetelor trimise de către client, aceste formate sunt:

type	format
si	username$password
sb	username$isbn$title$author$genre$year$rating
db	title
ud	idSearch$isbn
ml	username
eg	-
ea	author

Iar în cazul pachetelor trimise de către server clientului, formatele sunt:

type	format
si	(un|pw|ok)
sb	idSearch$K($isbn$title$N$author1$...$authorN$M$genre1$...$genreM)^K
db	-
ud	-
ml	K($isbn$title$N$author1$...$authorNMgenre1$...$genreM)^K
eg	N$genre1$...$genreN$M$u1$v1$...$uM$vM
ea	N$genre1$...$genreN

Simbolul $ reprezintă caracterul *newline*.

4.3 Sistemul de recomandări

Aplicația îi oferă utilizatorului, la cerere, o listă cu cinci recomandări. Algoritmul pentru alegerea acestor cărți este următorul. Ne uităm în tabela `history` la căutările efectuate de către utilizatorul curent. Între timp, construim pentru fiecare atribut din mulțimea $A = \{\texttt{author}, \texttt{genre}\}$ câte o listă cu toate valorile pe care le ia acesta în cadrul căutărilor respective. Valorile care se repetă au șanse mai mari să se regăsească mai târziu în recomandări.

Parcurgem listele și, pentru fiecare pereche (a, v), adăugăm la mulțimea de candidați cărțile ce au atributul a egal cu valoarea v, cu condiția ca acestea să nu fi fost descărcate deja de către utilizator. În cazul în care cartea curentă se află deja în mulțime, îi incrementăm frecvența.

La final, sortăm candidații descrescător după frecvență și îi alegem pe primii cinci. Dacă nu am obținut suficiente cărți, atunci încercăm să completăm recomandările cu cărți returnate de același algoritm, dar rulat de data aceasta pe mulțimea $A = \{\texttt{isbn}, \texttt{title}, \texttt{year}, \texttt{rating}\}$. Cu alte cuvinte, luăm în considerare și atributele mai puțin relevante. Dacă tot nu am obținut cinci cărți, completăm cu primele linii din tabela `books`.

4.4 Afișarea ierarhiei de genuri

Din perspectiva teoriei grafurilor, ierarhia de genuri și subgenuri este o pădure. Astfel, fiecare relație de forma (u, v), cu semnificația *genul v este subgen al genului u*, reprezintă o muchie în această pădure. Pentru a putea afișa ierarhia într-un mod intuitiv, trebuie să determinăm rădăcinile arborilor din pădure, adică genurile fără părinte, precum și muchiile acestor arbori.

Muchiile pot fi obținute efectuând un inner-join pe tabela `genres`, după condiția `g2.parent = g1.id_genre`. Inițial, toate genurile din tabela `genres` vor fi rădăcini. Însă, pentru fiecare muchie (u, v) găsită, vom elimina nodul v din set, deoarece el are un părinte, și anume u.

```
pair<set<string>, map<string, vector<string>>> getGenreHierarchy() {
    QSqlQuery query;
    query.exec("SELECT name FROM genres");
    set<string> roots;
    while (query.next())
        roots.insert(query.value(0).toString().toStdString());
    query.exec(
        "SELECT g1.name, g2.name FROM "
        "genres g1 JOIN genres g2 ON g2.parent = g1.id_genre"
    );
    map<string, vector<string>> edges;
    while (query.next()) {
        const string fath = query.value(0).toString().toStdString();
```

```
        const string node = query.value(1).toString().toStdString();
        edges[fath].push_back(node);
        roots.erase(node);
    }
    return make_pair(roots, edges);
}
```

5 Concluzii

Aplicația ReadsProfiler oferă funcționalitatea de bază a unei librării online, folosind tehnologii precum TCP/IP și SQLite. Acest proiect combină:

- tehnici de programare în rețea;
- lucrul cu baze de date relaționale;
- algoritmică și structuri de date.

Două posibile îmbunătățiri ale aplicației sunt:

- implementarea unei interfețe grafice, pentru a facilita utilizarea de către persoanele mai puțin experimentate cu terminalul;
- îmbunătățirea sistemului de recomandări.

6 Bibliografie

- qt.io
- sqldbm.com
- wikipedia.org
- profs.info.uaic.ro/ bd/
- profs.info.uaic.ro/ computernetworks/