

# ReadsProfiler

Buzenchi Paula-Roberta, grupa 2B4

Universitatea Alexandru-Ioan Cuza, Iași, Facultatea de Informatică  
buzenchiroberta@gmail.com

## 1 Introducere

Scopul proiectului **ReadsProfiler** este cel de a realiza o aplicație client-server prin care să se ofere acces la o librărie online. Aceasta poate gestiona căutări după criterii multiple și să ofere rezultate relevante, cât și posibilitatea de a descărca cărțiile. De asemenea, se introduce și un sistem de recomandări care se bazează pe preferințele utilizatorilor, cum ar fi genurile literare preferate, genurile abordate de un anumit autor, rating-urile cărților, căutările sau descărcările cititorilor. Astfel, obiectivul principal al proiectului este acela de a dezvolta o aplicație stabilă, ușor de extins, care să ofere funcționalități clasice utilizatorilor, cât și mecanisme de recomandări personalizate, așa încât experiența acestora pe platformă să fie cât mai plăcută.

## 2 Tehnologii Aplicate

Implementarea proiectului este realizată în limbajul **C/C++**, folosind funcții POSIX pentru programarea în rețea. Comunicarea dintre client și server se realizează prin intermediul protocolului **TCP**, datorită orientării sale asupra conexiunii, proprietăților prin care se asigură transmiterea datelor fără pierderi, într-o ordine corectă și fără duplicări. Aceste caracteristici sunt importante pentru operațiile de căutare după criterii multiple, accesare de ierarhii de genuri și subgenuri, descărcare de cărți și oferire de recomandări.

Facilitățile POSIX se reflectă prin apeluri precum **socket**, **bind**, **listen**, **accept**, **send**, și **recv**, și permit gestionarea conexiunilor simultane și definirea protocolului la nivel de aplicație. Pe partea serverului, datele sunt gestionate prin intermediul unei baze de date **SQLite**, ce permite stocarea informațiilor despre cărți, autori, genuri și istoricul utilizatorilor, ușurând generarea de recomandări.

Clientul permite utilizatorului să trimită comenzi, să se logheze, să caute, să descarce cărți și să ceară recomandări și să interacționeze cu funcționalitățile serverului într-un mod cât mai facil.

## 3 Structura aplicației

Proiectul **ReadsProfiler** este realizat pe modelul **server-client** cu protocol **TCP**. Scopul serverului este cel de a gestiona căutările, baza de date, sistemul

de recomandări, iar al clientului cel de a oferi o interfață grafică interactivă pentru utilizatori. Astfel, aplicația este organizată pe trei nivele: **nivelul de prezentare, nivelul de logică și nivelul de persistență**.

### 3.1 Concepte de Modelare

Modelarea aplicației are la baza mai multe entități ce descriu structura librăriei, interacțiunile dintre utilizatori și funcționalitățile ce generează recomandări. Fiecare entitate are rolul de a facilita funcționalitățile de căutare și descărcare.

#### – Books

Este obiectul central al sistemului, pe care toți utilizatorii o caută, vizualizează și descarcă

Atributele esențiale:

- id
- titlu
- autor
- genuri subgenuri
- an apariție
- ISBN
- rating
- calea pentru descărcare

#### – Book-genres

Atributele esențiale:

- id-ul cărților
- id-ul genurilor

#### – Authors

Scriitorii cărților.

Atributele esențiale:

- nume
- genurile literare abordate

#### – Author-genres

Atributele esențiale:

- autorii
- genurile abordate pentru fiecare

#### – Downloads

Atributele esențiale:

- id
- id-ul utilizatorului
- id-ul cărții
- când a fost descărcat

#### – Genres

Permite clasificarea cărților și reprezintă un criteriu pentru recomandări. Are structura unui arbore cu niveluri multiple, iar o carte poate apartine simultan mai multor noduri din arbore.

#### – Search-results-accesed

- id
- id-ul utilizatorului
- interogarea făcută
- când a fost făcută
- **Sqlite Sequence**
  - numele câmpului
  - câte date pentru fiecare nume sunt integrate
- **Users**
  - id-ul utilizatorului
  - numele de utilizator

În continuare vom vorbi despre componentizarea aplicației. Vom delimita responsabilitățile serverului și ale clientului.

Serverul reprezintă creierul aplicației. Toate operațiile de căutare, descărcare și recomandare sunt executate de către acesta. Vom detalia componentele principale ale serverului.

#### a. Connection Manager

- se ocupă cu comunicarea din TCP
- deschide socket-ul serverului
- inițializează socket-ul serverului(**socket, bind, listen**)
- acceptă clienti noi cu **accept**
- gestionează comunicarea cu fiecare client
- închide conexiunile

#### b. Request Parser

- rolul său este de a interpreta corect mesajele de la client
- validează structura cererii, identifică operația pe care trebuie să o facă (descărcare, căutare, recomandare), și extrage cuvintele esențiale

#### c. Search Engine

- motorul de căutare al aplicației
- pe baza parametrilor primiți(titlu, autor, gen, an, rating) construiește interogări **SQL** și le execută pe baza de date **SQLite**
- rezultatul este o listă de cărți care respectă preferințele utilizatorului

#### d. Recommendation Engine

- are rolul de a genera recomandări personalizate pentru fiecare utilizator pe baza comportamentului din trecut
- extrage istoricul utilizatorilor
- analizează genurile preferate
- identifică autori asemănători celor parcursi
- generează recomandări după genuri, autori și utilizatori cu preferințe similare

#### e. Download Manager

- se ocupă de descărcarea cărților și transmiterea fișierelor către client
- îi transmite informațiile prin intermediul TCP-ului folosind **send()**
- gestionează erorile

**f. SQLite Storage Layer**

- reprezintă stratul ce se ocupă cu gestionarea bazei de date
- se conectează baza de date SQLite
- definește și execută interogări SQL
- menține conexiunile cu tabelele de cărți, autori, relațiile dintre entități, ierarhia genurilor, utilizatori, istoric, ratinguri și descărcări
- închiderea conexiunilor și rezolvarea erorilor

Acum vom analiza componentizarea clientului. Acesta se ocupă cu interacțiunea cu utilizatorul.

**a. Command Dispatcher**

- trimitere cereri către server
- gestionează conexiunea **TCP** către server
- trimit datele prin **send()** și le recepționează prin **recv()**

**b. Response Handler**

- preia răspunsul serverului

### 3.2 Diagrama detaliată a aplicației

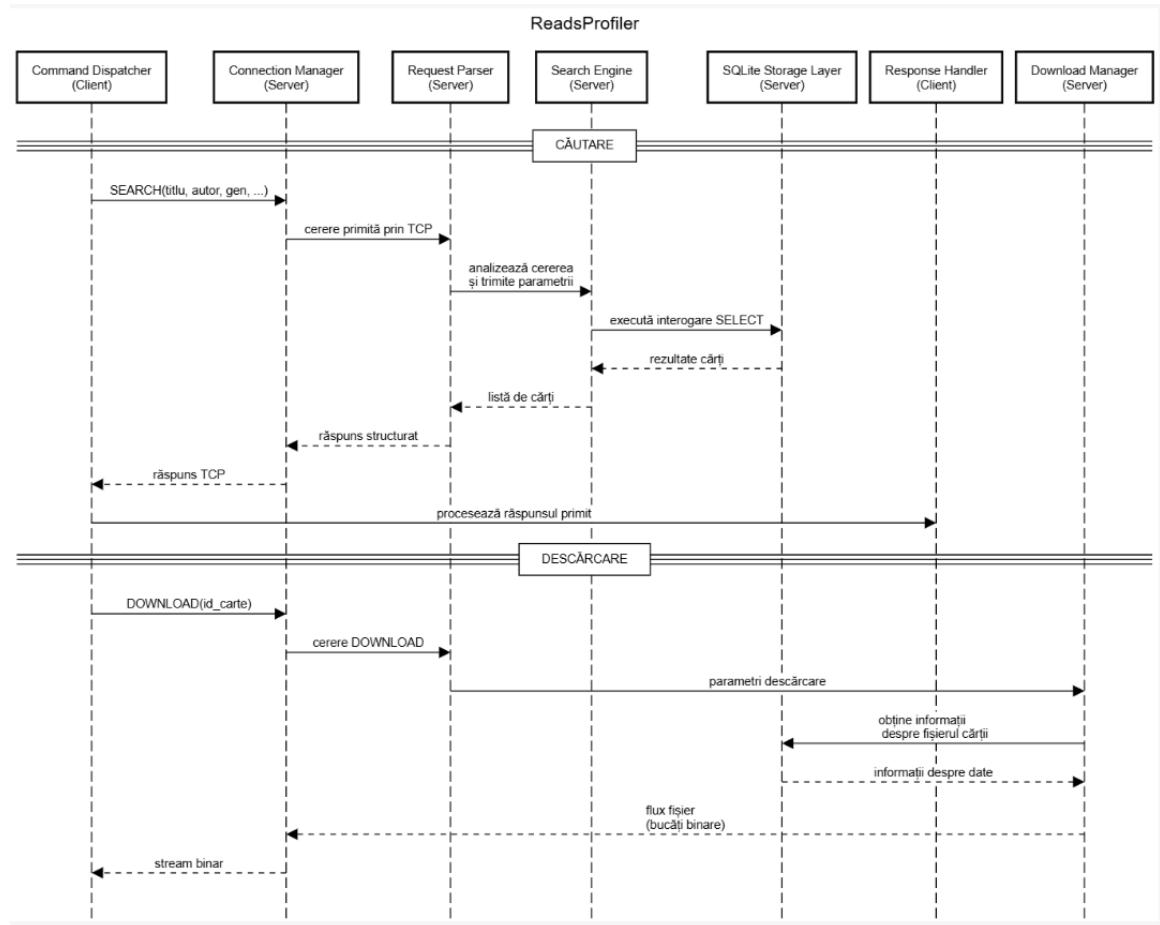


Fig. 1.

## 4 Protocolul de comunicare

Comunicare dintre cele două părți componente ale aplicației se realizează prin **TCP**. Se folosesc mesaje text care respectă formatul **COMANDA parametri**.

Protocolul include următoarele tipuri de mesaje trimise de client:

- **login <user>**: înainte de a efectua orice altă operație trebuie să fim logați

- **search <criterii>**: căutarea cărților se bazează pe un criteriu specific(titlu, autor, gen, subgen, an, rating)
- **download <criteriu>**: descarcă o carte pe baza titlului sau ISBN-ului
- **recommendations**: solicită recomandări personalizate pentru fiecare utilizator
- **quit**: închide aplicația

Comenzile vor fi scrise pe linii separate. În cazul în care o comandă nu este corectă sau validă, serverul va trimite un mesaj de eroare.

## 5 Scenarii de utilizare

### 5.1 Scenarii de succes

- \* **Căutarea după criterii multiple**
  - a. Utilizatorul accesează aplicația și introduce criterii pentru căutare(titlu, rating, autor, gen, rating)
  - b. Clientul trimite cererea de căutare către server, serverul interoghează baza de date SQLite, iar rezultatele căutării sunt retrimise clientului și afișate pe interfața grafică
  - c. Cititorul vede o listă de cărți ale căror caracteristici corespund criteriilor introduse de utilizator
- \* **Descărcarea unei cărți**
  - a. Utilizatorul selectează una sau mai multe cărți pentru descărcare
  - b. Clientul trimite cererea către server, iar acesta trimitе fiecare fișier ce se dorește a fi descărcat
  - c. În final, toate cărțile sunt descărcate corect și salvate local
- \* **Generare de recomandări**
  - a. Cititorul cere recomandări pe baza istoricului căutărilor și descărcărilor
  - b. Serverul procesează istoricurile, identifică genurile și autorii cei mai citiți de utilizator și generează recomandări
  - c. Utilizatorul primește o listă de recomandări

### 5.2 Scenarii de eșec

- \* **Căutare cu parametri incorecți**
  - a. Cititorul introduce criteriile de căutare într-o ordine greșită, incorecți sau incompleți
  - b. Clientul trimite cerere către server cu parametri invalizi
  - c. Serverul returnează un mesaj de eroare (error <motiv>) care explică ce s-a greșit, astfel încât ca utilizatorul să poată corecta
- \* **Descărcare eşuată din cauza unui fișier inexistent**
  - a. Utilizatorul încearcă să caute o carte care nu mai există sau introduce un ID greșit
  - b. Clientul trimite cererea de descărcare către server, însă acesta eșuează în a găsi cartea

- c. Serverul trimite înapoi un mesaj de eroare care explică de ce nu a putut găsi cartea
- \* **Recomandări eşuate din cauza lipsei istoricului**
  - a. Cititorul cere recomandare, însă nu a căutat sau descărcat nicio carte
  - b. Clientul trimite cererea către server, dar acesta nu poate procesa datele
  - c. Serverul returnează un mesaj de eroare care spune faptul că nu există informațiile necesare pentru a face recomandări

## 6 Concluzii

Aplicația **ReadsProfiler** se bazează pe sistemul server-client ce gestionează o librărie online, cu algoritmi de căutare, descărcare și de generare de recomandări. Sistemul de recomandări ar putea fi îmbunătățit prin utilizarea unor tehnici de învățare automată, care să ofere recomandări mai precise pe baza istoricelor de căutări și descărcări ale utilizatorului. În mod similar, interfața grafică ar putea fi extinsă cu opțiuni de filtrare și sortare mai avansate, iar notificările pentru recomandări ar îmbunătăți interacțiunea dintre utilizatori și aplicație. Aceste optimizări ar contribui la creșterea performanței și scalabilității proiectului.

## 7 Referințe bibliografice

1. *Computer Network – Course Materials*. Se găsesc la adresa: <https://edu.info.uaic.ro/computer-networks/>
2. *The Linux Programming Interface*. Disponibil la : The Linux Programming Interface - TCP/IP
3. *TCP Server – Client implementation in C*. Disponibil la: <https://www.geeksforgeeks.org/c/tcp-server-client-implementation-in-c/>
4. Springer. *Lecture Notes in Computer Science – Author Guidelines*. Disponibil la: <https://www.springer.com/gp/computer-science/lncs/conference-proceedings-guidelines>
5. SequenceDiagram.org. *Online Sequence Diagram Editor*. Disponibil la: <https://sequencediagram.org>
6. Cartea: *Computer Networking – A Top – Down Approach*, Kurose Ross