Universitatea Tehnică de Construcții din București

Facultatea de Hidrotehnică

Specializarea: Automatică și Informatică Aplicată

Proiect la Programarea calculatoarelor și limbaje de programare II

Profesori coordonatori:

Ing. Gabriela Olteanu

Ing. Cosmin Fudulu

Student:

Lavinia-Teodora Gheorghe

Universitatea Tehnică de Construcții din București

Facultatea de Hidrotehnică

Specializarea: Automatică și Informatică Aplicată

The Book Nook

Sistem de gestionare a bibliotecii personale

Profesori coordonatori:

Ing. Gabriela Olteanu

Ing. Cosmin Fudulu

Student:

Lavinia-Teodora Gheorghe

CUPRINS

I. MOTIVARE ALEGERE TEMĂ

Am ales să realizez un sistem de gestionare a bibliotecii personale, deoarece sunt pasionată de lectură. De-a lungul timpului, am adunat un număr mare de cărți acasă, în biblioteca fizică personală, având o varietate de categorii și genuri. Organizarea acestora a devenit o necesitate și am început sa testez diferite aplicații care sunt deja pe piață. Fiecare aplicație în parte are trăsături diferite, ideale pentru cititori, dar niciuna dintre ele nu le are pe toate în același timp. Din acest motiv, am ales să creez o aplicație care să îmbine măcar o parte din caracteristicile pe care aș dori să le văd într-un sistem pentru gestionarea bibliotecii, din punctul de vedere al unui iubitor de lectură.

II. OBIECTIVE PROPUSE

În cadrul acestui proiect, mi-am creat câteva obiective pentru a ajunge la rezultatul final dorit. Scopul principal este dezvoltarea unei aplicații dedicate gestionării cărților dintr-o bibliotecă personală, care să ofere utilizatorilor un mod eficient, simplu și prietenos pentru organizarea cărților. Primul obiectiv este crearea unei opțiuni prin care utilizatorul poate adăuga o carte nouă în sistem, oferind diverse detalii, de la titlu și autor, la data și locul de unde a fost achiziționată cartea. În acest mod, utilizatorul are un loc unde își poate găsi chiar și peste mai mulți ani, informațiile unei cărți.

Alt obiectiv personal este oferirea posibilității de a sorta cărțile după un criteriu ales. Câteva opțiuni de sortare sunt după titlu, după rating, după formatul cărții, după anul în care au fost citite, dar și altele. Astfel, utilizatorul poate vizualiza mai ușor și mai rapid cărțile pe care le deține, după filtrul dorit.

De asemenea, aplicația oferă posibilitatea de a căuta o carte, iar în urma acestei opțiuni, sunt afișate și detaliile cărții, într-un mod cât mai lizibil și prietenos. Altă trăsătură a sistemului este de a edita detaliile deja înregistrate în baza de date, sau ulterior, ștergea completă a unei cărți.

Un ultim obiectiv este crearea unei interfețe intuitive și ușor de folosit, care să ofere acces rapid la date și, totodată, o experiență plăcută utilizatorului.

III. TEHNOLOGII FOLOSITE

III.1. LIMBAJUL DE PROGRAMARE C++

Pentru a construi codul specific pentru aplicația dorită, am folosit limbajul de programare C++, care vine cu multiple biblioteci și funcții.

Limbajele de programare sunt limbaje asemănătoare cu limbajul uman. Conțin cuvinte (destul de puține), semne de punctuație, operații matematice și au reguli de scriere. Programele care rulează pe orice calculator au fost scrise într-un limbaj de programare.

Programul scris într-un limbaj de programare se numește program sursă și trebuie tradus într-un limbaj pe care îl înțelege procesorul, numit cod mașină, sau program executabil. Pentru anumite limbaje de programare operația de traducere se numește compilare (cazul lui C, C++, Pascal, etc). Traducerea este realizată de un program specializat numit compilator sau interpretor.

Limbajul de programare C++ are multe avantaje care îl fac atractiv pentru dezvoltatorii de software. Iată câteva dintre acestea:

1. Performanță înaltă: C++ este cunoscut pentru performanța sa înaltă și este utilizat în multe aplicații unde viteza și eficiența sunt critice.
2. Scalabilitate: Conform hashdork, C++ se mândrește cu o scalabilitate ridicată, ceea ce înseamnă că poate funcționa atât pe date la scară mică, cât și la scară mare.
3. Programare multi-paradigmă: C++ suportă atât programarea procedurală, cât și cea orientată pe obiecte, oferind flexibilitate dezvoltatorilor.
4. Utilizare largă în industrie: Multe companii mari, inclusiv Google și Facebook, folosesc C++ pentru dezvoltarea backend-ului lor.
5. Biblioteci puternice: C++ are o gamă largă de biblioteci standard și terțe părți care pot fi utilizate pentru a extinde funcționalitatea codului.

III.1.1. Fișiere

Un fişier este o colecţie de date indicat printrun nume şi o extensie. Numele este desparţit de extensie prin punct. Există doua tipuri de fisiere: fisiere text si fisiere binare. Un fişier text conţine text (cifre si caractere). Un fisier binar poate conţine si imagini, baze de date, etc.

În C++ există mai multe modalități de lucru cu fișiere text. Toate respectă următoarele etape:

1. deschiderea fișierului/asocierea fișierului cu un flux de date;
2. citirea din fișier/scrierea în fișier;
3. închiderea fișierului/fluxului de date

O modalitate uzuală de a deschide fișiere constă în declararea unor variabile de tip flux. Acestea sunt de tip:

1. ofstream pentru fluxurile de ieșire – asociate cu fișierele în care vom scrie;
2. ifstream pentru fluxurile de intrare – asociate cu fișierele din care vom citi;
3. fstream – are ambele caracteristici (ifstream si ofstream)

Am folosit un fișier “books.txt” pentru a salva datele introduse de utilizatori, dar și pentru a lucra cât mai simplu și structurat.

```
114    ifstream fin("C:\\Users\\lavin\\Documents\\UTCB 2024 - 2025\\library_management_system\\books.txt");
115    if(!fin.is_open()) {
116        cerr << "❌ Eroare la deschiderea fisierului!\n";
117        return;
118    }
```

```
89    ofstream fout("C:\\Users\\lavin\\Documents\\UTCB 2024 - 2025\\library_management_system\\books.txt");
90    if(!fout.is_open()) {
91        cerr << "❌ Eroare la deschiderea fisierului!\n";
92        return;
93    }
```

III.1.2. Funcții

Funcțiile permit structurarea programelor în secvențe mai mici de cod care îndeplinesc anumite sarcini. În C++, o funcție este un grup de instrucțiuni căruia i se dă un nume și care poate fi apelat dintr-un alt punct al programului. Cea mai utilizată sintaxă pentru definirea unei funcții este:

tip nume (parametru1, parametru2, ...) { instrucțiuni }
unde:

- *tip* este tipul valorii returnate de funcție.
- *nume* este identificatorul cu care funcția poate fi apelată.
- *parametri* (în număr necesar): fiecare parametru constă într-un tip urmat de un identificator și fiind separat de următorul prin virgulă. Fiecare parametru seamănă foarte mult cu o declarație obișnuită de variabilă și, de fapt, acționează în interiorul funcției ca o variabilă obișnuită care este locală (în funcție).
- *instrucțiuni* formează corpul funcției. Este un bloc de instrucțiuni cuprinse între acolade { } care precizează ceea ce să facă funcția.

Pentru a obține un cod structurat și mult mai simplu de înțeles, m-am folosit de funcții de tip void și de tip int / string. Am creat șapte funcții care operează opțiunile principale ale aplicației, dar și alte două funcții pentru validarea datelor calendaristice introduce și pentru convertirea în litere mici a variabilelor de tip string.

III.1.3. Structuri de date

O *structură de date* este o colecție de informații grupate sub un singur nume. Aceste informații, cunoscute drept *membri*, pot avea tipuri de date diferite și lungimi diferite. Structurile de date pot fi definite în C++ folosind următoarea sintaxă:

struct nume_tip {
tip_data_1 id_membru_1;
tip_data_2 id_membru_2;
tip_data_3 id_membru_3;
.
.
} denumiri_obiecte;

unde :

- *nume_tip* este o denumire pentru tipul structură.
- *denumiri_obiecte* poate fi un set de identificatori valizi de obiecte care sunt de acel tip structură.
- Între acolade {} avem o listă cu informațiile membre, fiecare dintre ele fiind precizată cu tip de dată și un identificator.

În cadrul codului pentru aplicația mea, am folosit două structuri de date, una pentru datele calendaristice (zi, lună, an) și una pentru detaliile care definesc o carte în sistemul creat.

```
13  struct date{
14      int day;
15      int month;
16      int year;
17  };
18
19  struct book{
20      string title;
21      string author;
22      string category;
23      string format;
24      string status;
25      int rating;
26      int nr_pages;
27      string bought_from;
28      date start_date;
29      date finish_date;
30      date buy_date;
31  };
32
33  book biblioteca[MAX_CARTI];
```

III.1.4. Instrucțiunea Switch case

De multe ori în codurile noastre o să avem nevoie de o modalitate de a executa anumite linii de cod doar dacă se îndeplinește o anumită condiție. Pentru a rezolva această necesitate, s-au creat structurile de decizie, deseori numite și structuri alternative.

În plus față de instrucțiunea de decizie foarte utilizată, *if else*, am ales să folosesc și instrucțiunea *switch case*. Aceasta este foarte utilă în cazul în care vrem să comparăm o valoare cu mai multe alte valori.

III.1.5. Biblioteci C++

Putem considera aceste biblioteci ca fiind niște „colecții", sau librării de instrucțiuni și funcții predefinite pe care le putem folosi în algoritmii noștri (funcții de prelucrare a șirurilor de caractere, funcții matematice etc.) Se includ în directiva de preprocesare adăugând secvența *#include* în fața denumirii acesteia (denumirea este încadrată între caracterele < și >).

1. iostream: folosim în principal pentru funcțiile de „citire de la tastatură" și „afișare pe ecran" (cin și cout)
2. fstream: pentru a declara instanțele citirii și afișării dintr-un fișier, avem nevoie de această bibliotecă.
3. string: este folosită pentru a lucra ușor și sigur cu șiruri de caractere(texte), oferind funcții pentru concatenare, căutare, comparare și manipularea textului.
4. limits: această bibliotecă oferă informații despre limitele minime și maxime ale variabilelor declarate și introduse în program.

III.2. VISUAL STUDIO CODE

Codul a trebuit construit și testat constant, pentru a evita erori majore, așa că am ales mediul de programare Visual Studio Code.

Visual Studio Code (VS Code) este un editor de cod sursă ușor, dar puternic, dezvoltat de Microsoft, care rulează pe desktop și este disponibil pentru Windows, macOS și Linux. Acesta vine cu asistență încorporată pentru JavaScript, TypeScript și Node.js și are un ecosistem bogat de extensii pentru alte limbi (cum ar fi C+, C#, Java, Python, PHP și Go) și runtimes (cum ar fi .NET și Unity). Pentru Python, VS Code poate fi configurat să ofere autocompletare, sugerează corecturi de sintaxă și poate rula și depana codul direct din editor.

Unul dintre cele mai atractive aspecte ale VS Code este interfața sa prietenoasă și intuitivă. VS Code oferă un sistem de taburi care permite deschiderea mai multor fișiere în paralel. Aceasta înseamnă că dezvoltatorii pot lucra simultan la mai multe părți ale unui proiect. De asemenea, oferă un „split view” pentru a vizualiza și edita două fișiere în același timp.

VS Code oferă un sistem de debugging extrem de puternic, care permite identificarea și corectarea erorilor în cod într-un mod vizual și intuitiv. Utilizatorii pot seta puncte de oprire, pot urmări variabilele în timp real și pot naviga prin stiva de apeluri pentru a înțelege mai bine fluxul de execuție al programului. Acest sistem de depanare este integrat cu majoritatea limbajelor de programare populare și poate fi extins pentru limbaje mai puțin utilizate.

Integrarea cu Git este un alt avantaj major al VS Code. Editorul vine cu un sistem de control al versiunilor încorporat care permite dezvoltatorilor să efectueze acțiuni Git (precum commit, push, pull) direct din editor. Interfața Git din VS Code oferă o modalitate vizuală de a compara fișierele, a adăuga sau elimina modificări și a crea ramuri de dezvoltare, toate aceste acțiuni fiind foarte ușor de accesat.

Visual Studio Code a devenit unul dintre cele mai utilizate și apreciate editoare de cod datorită combinației sale de performanță ridicată, extensibilitate și suport pentru multiple limbaje de programare. Interfața intuitivă, împreună cu caracteristici avansate precum depanare, integrare Git și o gamă largă de extensii, face din VS Code o alegere excelentă pentru orice dezvoltator, indiferent de nivelul de experiență. Prin faptul că este ușor de personalizat și optimizat pentru diverse fluxuri de lucru, VS Code se potrivește într-o varietate de scenarii de dezvoltare, de la aplicații desktop până la dezvoltare web și cloud.

IV. STUDIU DE CAZ – THE BOOK NOOK

IV.1. IMPLEMENTARE PAS CU PAS

În primul rând, codul începe cu apelarea bibliotecilor necesare, inițializarea variabilelor globale și definirea unei limite maxime pentru numărul de cărți care pot fi introduse în bibliotecă.

Codul se continuă cu cele două structuri de date (date și book) iar mai apoi este definită variabila bibliotecă de tip book, căreia i se atribuie limita maximă MAX_CARTI. Această variabilă este folosită pentru a stoca toate datele despre cărțile utilizatorului.

```
1  #include <iostream>
2  #include <string>
3  #include <fstream>
4  #include <limits>  // ofera informatii despre limitele max si min ale inputurilor
5  using namespace std;
6
7  #define MAX_CARTI 100000 //limita maxima de carti permise in biblioteca
8
9  int optiune1, filtru, nr_carti;
10 string titlu_find, titlu_edit, yes = "da", da_find = "da", da_edit = "da", da_delete = "da", optiune_meniu = "da" ;
11 bool go = true;
```

Am creat trei funcții ajutătoare: una pentru transformarea literelor variabilelor de tip string în litere mici, cu scopul de a compara mai eficient două variabile de tip string.

```
35 string tolower_string(string s) {
36     for(int i = 0; i < s.length(); i++) {
37         s[i] = tolower(s[i]);
38     }
39     return s;
40 }
41
```

A doua funcție este pentru validarea datelor calendaristice introduse de utilizator. Am ținut cont de anii bisecți și de numărul de zile ale lunii februarie, ce poate varia.

```
int valid_date(date d) {
  if(d.day< 1)
    return 0;
  if(d.month < 1 || d.month > 12)
    return 0;
  if(d.year < 999 || d.year > 10000)
    return 0;
  if(d.month == 1 || d.month == 3 || d.month == 5 || d.month == 7 || d.month == 8 || d.month == 10 || d.month == 12) {
    if(d.day > 31)
      return 0;
  }
  else if(d.month == 4 || d.month == 6 || d.month == 9 || d.month == 11) {
        if(d.day > 30)
          return 0;
  }
  else if(d.month == 2 && d.year % 4 == 0) {
        if(d.day > 29)
          return 0;
  }
  else if(d.month == 2 && d.year % 4 != 0) {
        if(d.day > 28)
          return 0;
  }
  return 1;

}
```

A treia funcție este extrem de importantă în codul programului meu, deoarece îmi identifică posibilele erori de introducere a valorilor bune de către utilizator și în același timp, rezolvă și problemele pe care le aveam de la citirea variabilelor de tip int / bool si apoi a unei variabile de tip string. După citirea unei variabile de tip int va rămâne în buffer caracterul '\n', care va fi mai apoi citit de getline(cin, string). Astfel, textul care se dorea introdus este ignorat și de aici încolo, apar diverse erori în rularea codului.

```
int get_valid_int(const string& message) {
  int val;
  while(true) {
    cout << message;
    if(cin >> val) {
      cin.ignore();
      return val;
    }
    else {
      cout << "🤔 Hmm...cred ca ai gresit ceva. Incearca din nou!\n";
      cin.clear(); //reseteaza starea lui cin, care se afla modul de eroare
      cin.ignore(numeric_limits<streamsize>::max(), '\n');   // ignora caracterele introduse gresit pana la '\n'
      //numeric_limits<streamsize>::max() este functie specifica <limits> care returneaza
      //valoarea maxima posibila pentru streamsize (tip intreg folosit de stream-uri (cin, cout))
    }

  }
}
```

În continuare, în cod se regăsesc cele două funcții care se ocupă cu prelucrarea datelor din fișierul “books.txt”, mai precis funcția save_books() salvează în fișier toate datele furnizate de utilizator în cadrul unei rulări de program, iar funcția load_books(), apelată la început de program, a fost creată cu scopul de a încărca din fișier informațiile salvate anterior.

```
112 void load_books() {
113     nr_carti = 0;
114     ifstream fin("C:\\Users\\lavin\\Documents\\UTCB 2024 - 2025\\library_management_system\\books.txt");
115     if(!fin.is_open()) {
116         cerr << "❌ Eroare la deschiderea fisierului!\n";
117         return;
118     }
119     book b;
120     while(getline(fin, b.title, ';')) {
121         getline(fin, b.author, ';');
122         getline(fin, b.category, ';');
123         getline(fin, b.format, ';');
124         getline(fin, b.status, ';');
125 
126         string ratingg;
127         getline(fin, ratingg, ';');
128         b.rating = stoi(ratingg);
129 
130         string nrpag;
131         getline(fin, nrpag, ';');
132         b.nr_pages = stoi(nrpag);
133 
134         getline(fin, b.bought_from, ';');
135 
136         fin >> b.buy_date.day >> b.buy_date.month >> b.buy_date.year;
137         fin.ignore(1);  //ignore ';'
138         fin >> b.start_date.day >> b.start_date.month >> b.start_date.year;
139         fin.ignore(1);
140         fin >> b.finish_date.day >> b.finish_date.month >> b.finish_date.year;
141         fin.ignore();  //ignore '\n'
142 
143         biblioteca[nr_carti++] = b;
144 
145     }
146 
147     fin.close();
148 }
```

```
88  void save_books() {
89      ofstream fout("C:\\Users\\lavin\\Documents\\UTCB 2024 - 2025\\library_management_system\\books.txt");
90      if(!fout.is_open()) {
91          cerr << "❌ Eroare la deschiderea fisierului!\n";
92          return;
93      }
94 
95      for(int i = 0; i < nr_carti; i++) {
96          fout << biblioteca[i].title << ";"
97               << biblioteca[i].author << ";"
98               << biblioteca[i].category << ";"
99               << biblioteca[i].format << ";"
100              << biblioteca[i].status << ";"
101              << biblioteca[i].rating << ";"
102              << biblioteca[i].nr_pages << ";"
103              << biblioteca[i].bought_from << ";"
104              << biblioteca[i].buy_date.day << " " << biblioteca[i].buy_date.month << " " << biblioteca[i].buy_date.year << ";"
105              << biblioteca[i].start_date.day << " " << biblioteca[i].start_date.month << " " << biblioteca[i].start_date.year << ";"
106              << biblioteca[i].finish_date.day << " " << biblioteca[i].finish_date.month << " " << biblioteca[i].finish_date.year << "\n";
107 
108     }
109     fout.close();
110 }
```

Toate opțiunile oferite la început de program utilizatorului sunt gestionate de cinci funcții, denumite sugestiv.

```
150 > void add_book() {···
268
269 > void show_details_book(string title) {···
301
302 > void show_sorted_books(int filter) {···
448
449 > void edit_book(string title) {···
573
574 > void delete_book() {···
592
```

În momentul în care este apelată funcția *add_book()*, sunt afișate și citite pe rând informațiile care trebuie introduse. Pe baza anumitor răspunsuri, sunt cerute și alte date. De exemplu, dacă utilizatorul spune că formatul cărții este fizic, acesta va fi întrebat dacă dorește să introducă locul și data de când a fost cumpărată cartea. Altă opțiune similară este cea când este identificat statusul cărții ca fiind "citită", iar utilizatorul are varianta de a spune și perioada în care acesta a citit-o.

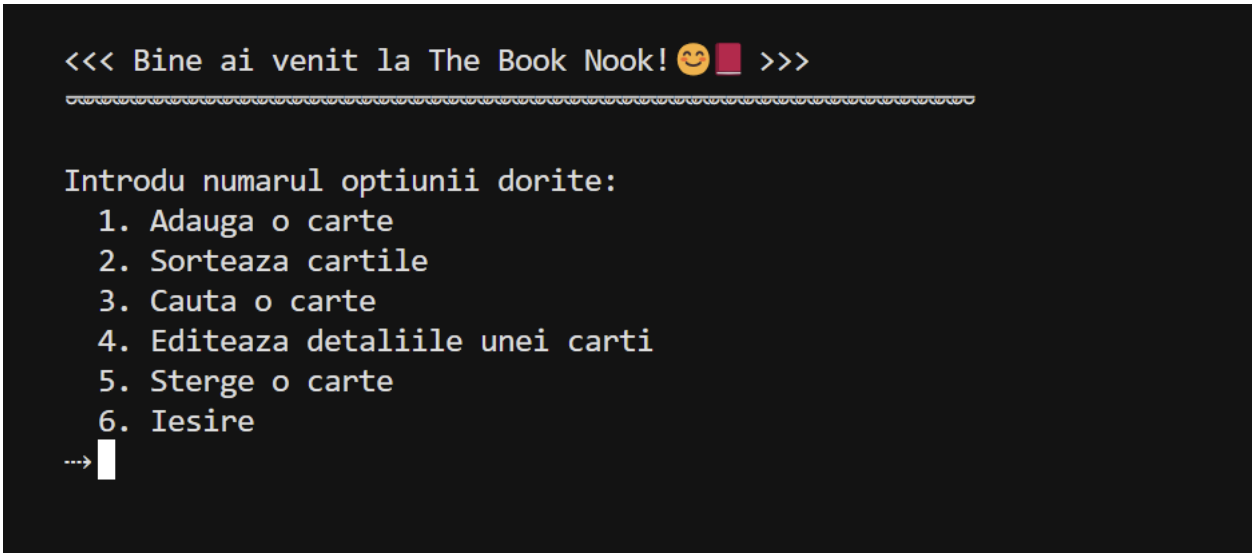
Funcția *show_details_book(string title)* are rolul de a afișa în terminal toate detaliile prezente în fișier, care aparțin de titlul cărții introdus de la tastatură. Este utilizată în cazul în care utilizatorul dorește să caute o carte.

Funcția *show_sorted_books(int filter)* folosește o instrucțiune *switch case* pentru a oferi mai eficient un răspuns utilizatorului. Există 11 variante de sortare în aplicație, după criterii precum titlu, autor, format, anul în care au fost citite cărțile, wishlist sau crescător după numărul de pagini sau a rating-ului oferit. De asemenea, utilizatorul poate alege și să afișeze cărțile nesortate, acestea fiind afișate după ordinea din fișier.

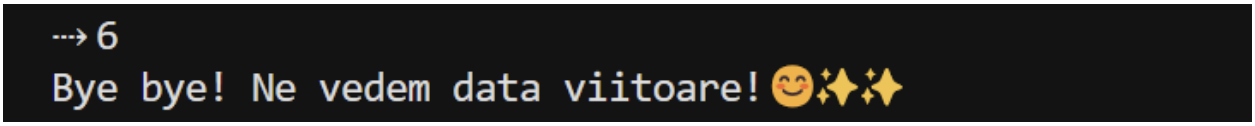
Funcția *edit_book(string title)* permite utilizatorului să modifice informații deja existente în fișier, întrebându-l mai întâi ce vrea să editeze.

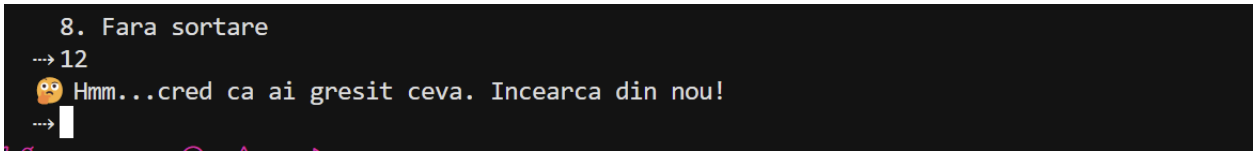
Funcția *delete_book()* este apelată în cazul în care utilizatorul dorește să șteargă de tot o carte din biblioteca sa digitală.

Nu în ultimul rând, funcția *main()* îmbină toate celelalte funcții și variabile create anterior. În int *main()* programul rulează meniul oferit utilizatorului, folosind o structură de decizie *switch case*.



Scopul meu a fost și să creez o interfață prietenoasă, de aceea am ales să fiu foarte atentă la mici detalii legate de afișare. Utilizatorul este întâmpinat de mesaje de eroare, mesaje de succes, dar și o interfață ordonată și accesibilă.





V. CONCLUZII

În primul rând, pe parcursul dezvoltării aplicației “The Book Nook” am reușit să îmi ating toate obiectivele inițial propuse. În final, am creat o aplicație care gestionează eficient și ușor toate cărțile din biblioteca unui utilizator, creând astfel un meniu interactiv cu multe opțiuni pentru acesta.

În al doilea rând, am reușit să creez această aplicație pe cont propriu, ajutându-mă de cunoștințele deja avute, dar și de materiale din cursuri, laboratoare sau de pe site-uri specializate pe limbaje de programare. După gândirea unui plan bine structurat, am parcurs pas cu pas etapele acestuia, iar organizarea bine pusă la punct se reflectă și în codul aplicației, acesta fiind structurat în funcții diverse, care mai apoi doar sunt apelate.

De asemenea, prin intermediul acestui proiect, am avut ocazia să îmi dezvolt abilitățile de programare și cunoștințele. Am învățat foarte mult să lucrez cu variabile de tip struct, care vin cu o bibliotecă cu funcții mai diferită față de variabilele de tip char, utilizate în majoritatea timpului. Am descoperit și o bibliotecă nouă, care oferă informații despre limitele variabilelor. În același timp, mi-am dezvoltat și gândirea logică și imaginația.

În concluzie, petrecând mult timp dezvoltând acest program interactiv, mi-am aprofundat cunoștințele în materie de programare, în special limbajul de programare C++, care vine cu o multitudine de biblioteci ajutătoare.

VI. BIBLIOGRAFIE

1. https://www.pbinfo.ro/articole/59/introducere-in-cpp
2. https://cplusplus.com/doc/tutorial-ro/functions/
3. https://cplusplus.com/doc/tutorial-ro/structures/
4. https://infoas.ro/lectie/3/instructiunea-de-decizie-in-c-if-else-switch-case
5. https://www.ezinfo.ro/documentatie/biblioteci-cpp.html
6. https://www.w3schools.com/cpp/cpp_strings.asp
7. https://www.tutorialspoint.com/cpp_standard_library/limits.html

VII. ANEXE

```
books.txt
None of This Is True;Lisa Jewell;Thriller;ebook;citita;9;370;;0 0 0;2 10 2024;18 10 2024
House of Hollow;Krystal Sutherland;young adult;fizic;citita;8;300;;0 0 0;0 0 0;0 0 0
Gods of Jade and Shadow;Silvia Moreno-Garcia;fantasy;fizic;citita;9;336;Okian;4 5 2022;0 0 0;0 0 0
Little Women;Louisa May Alcott;classics;fizic;citita;8;354;Carturesti;0 0 0;16 7 2022;30 7 2022
The Empire of Gold;S.A. Chakraborty;Fantasy;fizic;citita;10;790;Carturesti;0 0 0;2 9 2024;13 9 2024
```

Fig. 8.1 – Datele salvate în fișier

```
⇢1
〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰
                Introdu:

 🔸Titlu: The Naturals

 🔸Autor: Jennifer Lynn Barnes
 🔸Categorie: Mystery
 🔸Format (fizic / ebook / audiobook): fizic

Doresti sa introduci locul de unde ai cumparat cartea? (Da / Nu)
⇢da
 🔸Loc achizitie: Okian

Doresti sa introduci data cand ai cumparat cartea? (Da / Nu)
⇢nu
 🔸Numar de pagini: 308
 🔸Status (wishlist / citita): citita
 🔸Rating (0-10): 10

Doresti sa introduci perioada in care ai citit-o? (Da / Nu)
⇢da
〰〰〰〰〰〰〰〰〰〰 🔸Data inceperii citirii 〰〰〰〰〰〰〰〰〰〰
 🔸Zi (dd): 09
 🔸Luna (mm): 02
 🔸An (yyyy): 2025
〰〰〰〰〰〰〰〰〰〰 🔸Data finalizarii citirii 〰〰〰〰〰〰〰〰〰〰
 🔸Zi (dd): 14
 🔸Luna (mm): 02
 🔸An (yyyy): 2025

Cartea a fost adaugata cu succes!🎉
〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰〰

Doresti sa mai adaugi o carte? (Da / Nu)
⇢nu
```

Fig. 8.2 – Adăugarea unei cărți

Fig. 8.3 – Sortarea cărților după rating

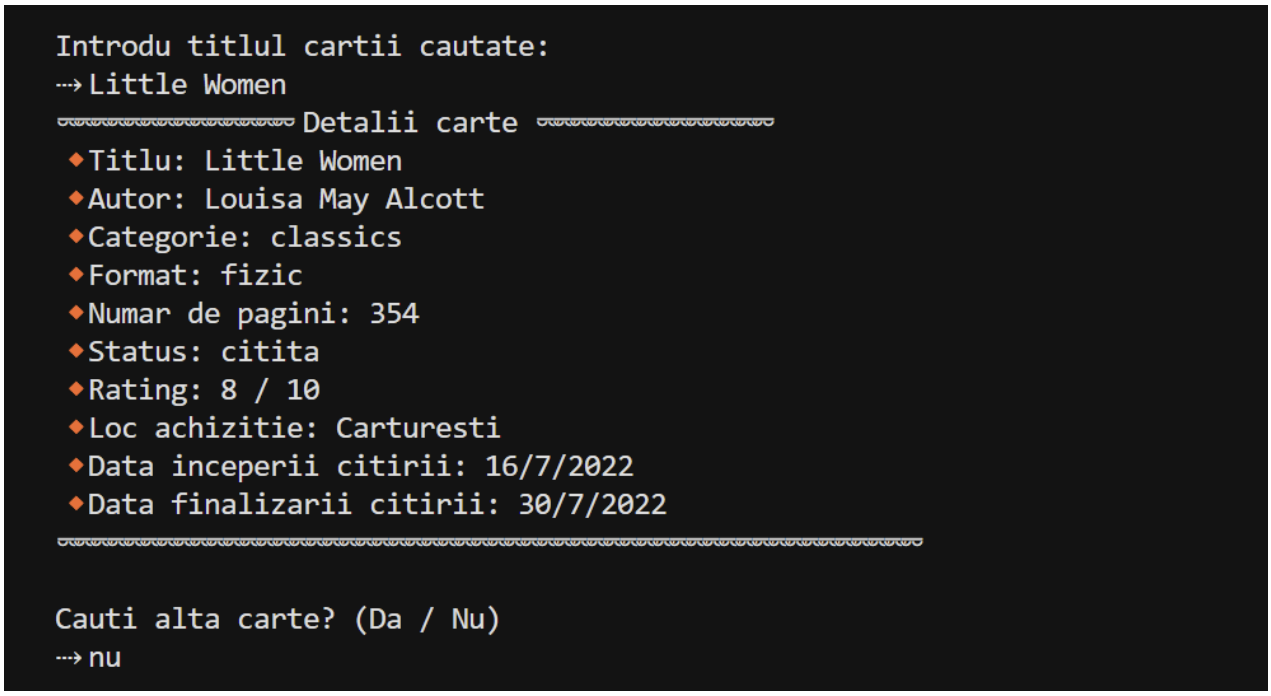


Fig. 8.4 – Căutarea unei cărți

```
Introdu titlul cartii pe care vrei sa o editezi:
house of hollow

Ce doresti sa modifici?
  1. Titlu
  2. Autor
  3. Categorie
  4. Format
  5. Status
  6. Numar pagini
  7. Rating
  8. Loc achizie
  9. Data achizitie
  10. Data inceput citire
  11. Data finalizare citire
⇢ 6
 ◆Numar nou de pagini: 300

Doresti sa editezi alta carte? (Da / Nu)
⇢ nu
```

Fig. 8.5 – Editarea unei cărți

```
⇢ 5
~~~~~~~~~~~~~~~~~ Cartile disponibile ~~~~~~~~~~~~~~~~~
  1. "None of This Is True"
  2. "House of Hollow"
  3. "Gods of Jade and Shadow"
  4. "Little Women"
  5. ""
  6. "The Empire of Gold"

Alege numarul cartii pe care doresti sa o stergi: 5
Cartea a fost stearsa cu succes!🎉

Doresti sa stergi alta carte? (Da / Nu)
⇢ nu
```

Fig. 8.6 – Ștergerea unei cărți