### UNIVERSITATEA DIN BUCUREȘTI

**FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ**

**SPECIALIZAREA CALCULATOARE ȘI TEHNOLOGIA INFORMAȚIEI**

**Lucrare de licență**

# Aplicație desktop ce urmărește modernizarea sistemului de management al unei biblioteci

### Absolvent

### Buturuga George-Alexandru

**Coordonator științific**

**Marius Mihăilescu**

**București, iunie 2024**

#### Rezumat

În era digitală, bibliotecile se confruntă cu provocări din ce în ce mai complexe în gestionarea resurselor, oferirea unor servicii eficiente și adaptarea la nevoile utilizatorilor. O aplicație modernă, bazată pe o bază de date ușor de accesat prin cadrul serviciilor bazate pe REST Api, un sistem bine proiectat prin care stocurile pot fi ușor de gestionat, o interfață intuitivă și un algoritm precis de recomandări poate reprezenta un pas crucial în evoluția gestionării bibliotecilor. Această propunere explorează o abordare inovatoare bazată pe tehnologie pentru optimizarea operațiunilor și îmbunătățirea experienței utilizatorilor.

#### Abstract

In the digital era, libraries face increasingly complex challenges in managing resources, providing efficient services, and adapting to user needs. A modern application, based on an easily accessible database through a framework of services based on REST API, a well-designed system through which stocks can be easily managed, an intuitive interface, and a precise recommendation algorithm can represent a crucial step in the evolution of library management. This proposal explores an innovative technology-based approach tooptimizing operations and enhancing the user experience.

Cuprins

[UNIVERSITATEA DIN BUCUREȘTI 1](#_Toc168326542)

[Aplicație desktop ce urmărește modernizarea sistemului de management al unei biblioteci 1](#_Toc168326543)

[Absolvent 1](#_Toc168326544)

[Buturuga George-Alexandru 1](#_Toc168326545)

[1. Introducere 5](#_Toc168326546)

[1.1 Istoria aplicaților desktop 5](#_Toc168326547)

[1.2 Software-ul pentru desktop de-a lungul anilor 5](#_Toc168326548)

[1.3 Viitorul aplicaților desktop 6](#_Toc168326549)

[1.4 Provocări 7](#_Toc168326550)

[1.5 De ce am ales această temă? 7](#_Toc168326551)

[1.6 Tehnologii utilizate 7](#_Toc168326552)

[1.7 Flux de lucru 8](#_Toc168326553)

[2. Servicii 9](#_Toc168326554)

[2.1. Serviciul identity 9](#_Toc168326555)

[2.2. Serviciul biblioteca-backend 10](#_Toc168326556)

[2.3. Arhitectura Serviciului de Frontend 10](#_Toc168326557)

[3. Design 11](#_Toc168326558)

[3.1. Design-ul formularelor 11](#_Toc168326559)

[3.2. Design-ul elementelor 11](#_Toc168326560)

[4. Funcționalități 15](#_Toc168326561)

[4.1 Pagina de logare 15](#_Toc168326562)

[4.2 Pagina principala Utilizator 15](#_Toc168326563)

[4.3 Pagina principala Admin 17](#_Toc168326564)

[4.4 Funționalitatea căutare 18](#_Toc168326565)

[4.5 Pagina Profil 19](#_Toc168326566)

[4.5.1 Funționalități 19](#_Toc168326567)

[4.5.2 Pop-up Schimbă Numele 20](#_Toc168326568)

[4.6 Pagina cereri utilizatori 21](#_Toc168326569)

[4.7 Pagina toate recenziile 22](#_Toc168326570)

[4.8 Pagina adaugă carte 23](#_Toc168326571)

[4.8.1 Modul adăugare 23](#_Toc168326572)

[4.8.2 Modul editare 25](#_Toc168326573)

[4.8.3 Adaugă Autor 26](#_Toc168326574)

[4.8.4 Adaugă Autor 27](#_Toc168326575)

[4.9 Vizualizare carte 28](#_Toc168326576)

[4.9.1 Vizualizare 28](#_Toc168326577)

[4.9.2 Afișare recenzii 29](#_Toc168326578)

[4.9.3 Împrumută carte 30](#_Toc168326579)

[4.9.4 Lasă Recenzie 31](#_Toc168326580)

[4.10 Prezentare Generală 32](#_Toc168326581)

[5. Concluzie 33](#_Toc168326582)

[6. Bibliografie 34](#_Toc168326583)

# Introducere

## Istoria aplicaților desktop

În zilele noastre, înțelegem prin dezvoltare software dezvoltarea unui program, ce are ca rol îndeplinirea unei anumite funcții.

O aplicație desktop în mod normal trebuie instalată pentru a rula, acest lucru îi va permite ulterior utilizatorului să interacționeze cu aplicația fără a fi conectat la site-ul furnizorului.

## Software-ul pentru desktop de-a lungul anilor

**Anii 1960 - Primele aplicații software:** Discută despre primele programe și aplicații software dezvoltate în anii 1960, cum ar fi limbajele de programare de bază și primele sisteme de operare.

**Anii 1970 - Microcalculatoare și primele aplicații comerciale:** Detaliază cum apariția microcalculatoarelor a dus la dezvoltarea primelor aplicații comerciale și a software-ului vândut separat de hardware.

**Anii 1980 - Era PC-ului și apariția GUI:** Explorează impactul introducerii PC-urilor și a interfețelor grafice cu utilizatorul (GUI) asupra popularității și utilizabilității software-ului pentru desktop.

**Anii 1990 - Revoluția Internetului și distribuția software-ului:** Analizează cum distribuția software-ului a fost revoluționată de apariția CD-ROM-urilor și a internetului, facilitând descărcarea și actualizarea software-ului.

**Anii 2000 și ulterior - Aplicații integrate și mobilitatea software-ului:** Discută despre tranziția către aplicații integrate și mobile, și despre cum software-ul pentru desktop continuă să evolueze pentru a rămâne relevant.

În anii 1980, introducerea PC-urilor și a interfețelor grafice cu utilizatorul (GUI) a transformat dramatic peisajul software-ului pentru desktop. IBM a lansat primul său computer personal în 1981, care includea sistemul de operare MS-DOS de la Microsoft. În 1985, Microsoft a lansat prima versiune a Windows, aducând o interfață grafică ușor de utilizat pentru computerele personale. Aceasta a permis utilizatorilor să interacționeze cu software-ul într-un mod mai intuitiv și a deschis calea pentru dezvoltarea unui ecosistem vast de aplicații software.[[1](#bibliografie_1)]

## Viitorul aplicaților desktop

Viitorul aplicațiilor desktop este strâns legat de evoluția tehnologiilor software si hardware. O aplicație desktop consumă mai puțină memorie decât o aplicație web, însă datorită evoluției acelerată a harware-ului și modului în care tehnologiile funcționează e posibil să se facă abstracție de aceste lucruri. De asemenea apar aplicații ce permit o integrare ușoară a tehnologiilor web precum Angular într-o aplicație desktop. Inteligența artificială va juca un rol foart imortant în dezvoltarea de noi funcționalități, cum ar fi asistenții virtuali și analiză predictivă, transformând aplicațiile desktop în instrumente și mai puternice și mai intuitive.

## Provocări

În dezvoltarea proiectului, am întâmpinat probleme la interconectarea serviciilor între ele, de asemenea am mai întâmpinat probleme la portabilitatea între dispozitive și mutarea bazei de date, însă într-un final am reușit să integrez aceste funcționalități.

## De ce am ales această temă?

Am ales această temă deoarece oferă o mare flexibilitate asupra structurii pe care o gândeam inițial în realizarea proiectului. De asemenea am vrut să abordez o nouă metodă prin care bibliotecile ar putea îmbunătății și reduce timpul cititorilor în bibliotecă.

## Tehnologii utilizate

Tehnologiile utilizate în proiect au fost selectate în funcție de sectoarele pe care am dorit să-mi aprofundez cunoștințele. Acestea sunt reprezentate de:

* WindowsFormApplication: Utilizând C#
* Entity Framework Core: Utilizat pentru maparea entităților, lucru ce mi-a permis să lucrez cu baza de date.
* ASP.CORE: Core open-source și cross-platform pentru dezvoltarea aplicațiilor web moderne
* Identity FrameWork: ASP.CORE Identity a fost utilizate pentru a creea serviciul de identity unde generez token-ul de autentificare și unde se efectuează logarea și înregistrarea utilizatorilor.
* Sql Server – Utilizat pentru gestionarea bazei de date a proiectului prin interfața din Visual Studio.
* WindowsForm – Bibliotecă GUI din .NET Framework utilizată pentru a crea aplicația desktop.

Tehnologii suport:

* NuGet: Manager de pachete pentru platforma .NET
* Git: Sistem de control al versiunii
* Visual Studio: Mediu de dezvoltare de la Microsoft pentru dezvoltarea aplicaților

## Flux de lucru

* Autentificare: Utilizatorii se pot înregistra și autentifica prin API-ul de autentificare
* Gesionarea Cărților: Administatorii pot adăuga, vizualiza și șterge cărți prin interfață.
* Împrumuturi și recenzii: Utilizatorii pot împrumuta și adăuga recenzii.

# Servicii

## 2.1. Serviciul identity

Serviciul de identity din cadrul proiectului meu, are ca scop gestionarea autentificarii și autorizării utilizatorilor. Acesta a fost creat cu utilizând framework-ul ASP.NET Core Identity. [[2](#bibliografie_2)].

Framework-ul ASP.NET Core Identity este un framework ușor de gestionat în ceea ce privește facilitarea utilizatorilor acces la funcționalitățile aplicației, principalele avantaje fiind:

Securitate: prin gestionarea complexă a parolelor și a token-urilor.

Personalizare: framework-ul permite personalizarea modelelor.

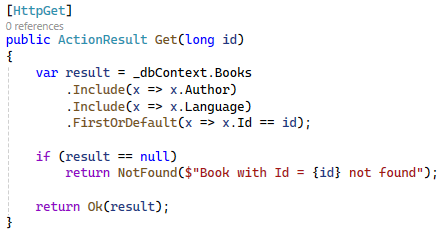
Integrare ușoară: framework-ul este bine documentat și ușor de gestionat în cadrul aplicației.

## Serviciul biblioteca-backend

Serviciul biblioteca-backend este responsabil de gestionarea funcționalităților esențiale ale aplicației, asigură o bună funcționare, validează datele și securizează conexiunile.

Arhitectura Serviciului:

* Controlere: Gestionează cererile HTTP primite de la frontend și trimit răspunsurile corespunzătoare. Mai jos avem exemplificat unul dintre end-pointuri.



* Servicii: Serviciile sunt utilizate de controlere pentru a efectua operațiuni complexe.
* DTOs (Data Transfer Objects): Obiecte folosite pentru a transfera datele
* Modele: Reprezinta structura datelor din baza de date

## Arhitectura Serviciului de Frontend

Arhitectura serviciului de frontend se bazeaza pe Windows Forms Application, o interfață de programare a aplicațiilor grafice(API) inclusă ca parte a .NET Framework de la Microsoft, oferind acces la elemente native la interfeței Microsoft Windows prin intermediul interfeței. [[3](#bibliografie_3)]

Proiectul dispune de opt formulare, cu interfață intuitivă și ușor de utilizat. Procesele sunt realizate asincron astfel incat aplicatia sa fie cât mai fluidă și ușor de utilizat.

# Design

### 3.1. Design-ul formularelor

Pentru designul formularelor am utilizat numai componente puse la dispoziție de Vistual Studio, modificând proprietățiile componentelor. Logo-ul și restul elementelor vizuale ce nu sunt incluse în Visual Studio au fost modelate în photoshop(imagini). De-a lungul proiectului am utilizat documentația oficială pentru a înțelege fiecare componentă în parte. [[4](#bibliografie_4)].  
 Font-ul utilizate în proiect este PoorRichard. [[5](#bibliografie_5)].

### Design-ul elementelor

* + 1. **Buton**

Am ales un design simplu pentru butoane astfel încât să reflecte atmosfera calmă a unei biblioteci, îmbinându-se perfect cu restul culorilor.   
 Proprietăți:

* BackColor: Transparent
* FontStyle: PoorRichard
* ForeColor: White
* FlatStyle: Flat (Proprietate ce permite personalizarea borderului)
* FlatAppearance
  + BorderColor: White
  + BorderSize: 1
  + MouseDownBackColor: Silver
  + MouseOverBackColor: Gray

  
 Fig. 3.1 Buton

* + 1. **Panel**

Panelul are o interfață ușor de vizionat păstrând aceeași textura ca form-ul.   
  
 Proprietăți:

* BackColor: 54,75,109
* Pentru a realiza marginile, am utilizat 4 paneluri cu dimensiune de 1px si cu dock in cele 4 colturi ale panelului parinte.

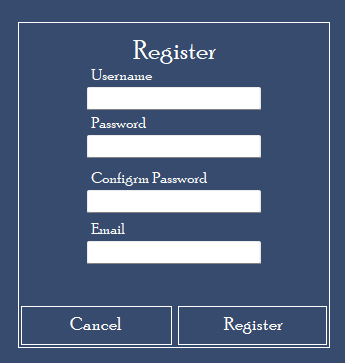


Fig. 3.2 Panel

* + 1. **DataGridView**

DataGridView-ul pastrează același model, preia culoarea formularului, la fel și pentru rândurile inserate.  
  
 Proprietăți:

* BackColor: 54,75,109
* AlternatingRowsDefaultCell
  + BackColor: 54,75,109
  + ForeColor: White
* DefaultCellStyle
  + BackColor:54,75,109
  + ForeColor: White
* Dock: Fill (în panelul parinte)

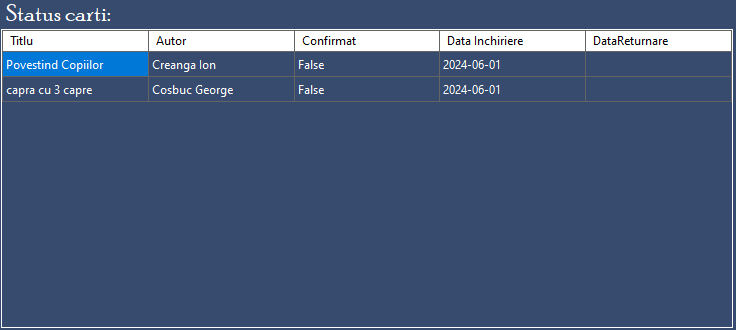


Fig. 3.3 DataGridView

**4. Vizualizare Carți Noi** Această secțiune este compusă dintr-un panel părinte, ce conține 4 pictureBox-uri, un label și două butoane, ce permit navigarea printre cărți.   


Fig. 3.4. Vizualizare Cărți Noi

**5. Căutare cărți** Componenta de căutare cărți este compusă dintr-un textBox și un ListBox, cărora le-au fost eliminate marginile. Cele două comunică și sunt acționate concomitent.

**** Fig. 3.5. Căutare cărți

# Funcționalități

### 4.1 Pagina de logare

Pagina de logare are două scopuri, înregistrare și autentificare. Logarea utilizatorilor se face în mod implicit la pornirea aplicației, aceștia având posibilitatea să înregistreze un nou cont în cazul în care nu au unul deja creat. La prima logare utilizatorul va fi inițializat în baza de baze, primind și un mesaj de confirmare. Daca utilizatorul este admin, acesta va primi acces la funcții speciale în aplicație.

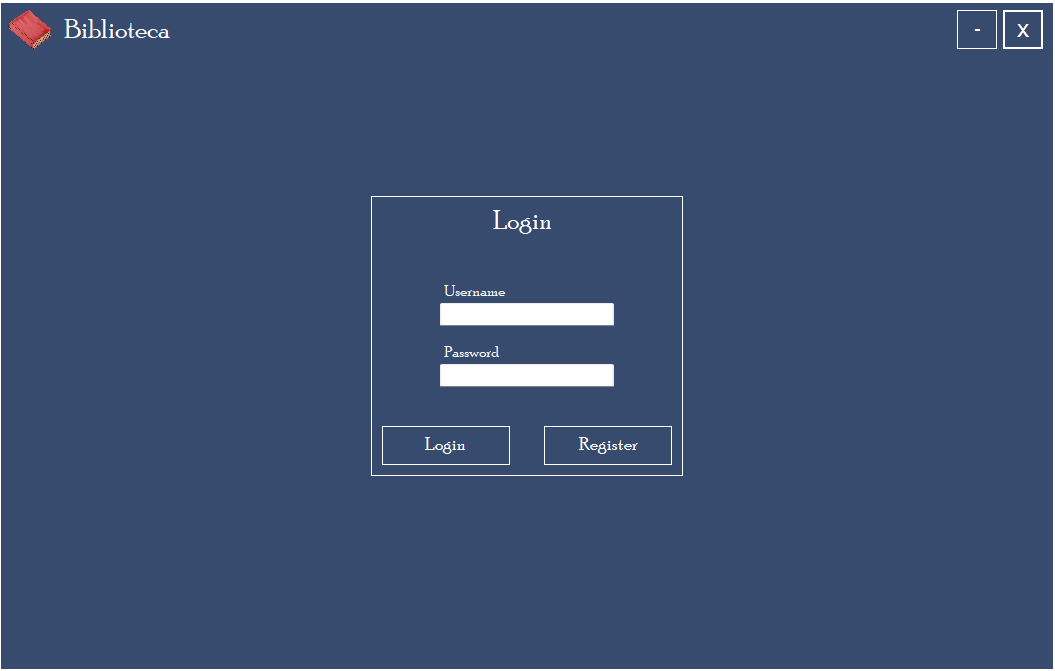
  
Fig. 4.1.1 Logare



Fig. 4.1.1 Logare

## 4.2 Pagina principala Utilizator

Dupa autentificare, utilizatorul este redirecționat către pagina principală, de unde poate naviga mai departe în aplicație. În secțiunea cărți noi, apar toate cărțile din baza de date filtrate în mod implicit după data adăugării, utilizatorul poate derula lista acționând unul dintre cele două butoane situate în partea de sus a secțiunii “Cărți noi”. Label-ul afișează numărul ultimei curent al ultimei cărți din lista vizibila și numărul total de cărți ce au fost aduse din baza de date.

Secțiunea “Cărți populare”, afișează în ordine descrescătoare cărțile cu cele mai multe recenzii din aplicație.

Meniul utilizatorului conține 4 opțiuni:

* Căutare: Îi permite utilizatorului să caute cărți în funcție de nume. Apelul de căutare folosește cărțile prezente deja în lista “Cărți noi”, acest lucru sporind performanța aplicației nefiind nevoie ca datele să fie filtrate pe serviciul de backend.
* Profil: Utilizatorul va fi redirecționat către pagina de profil.
* Contact: Se va afișa un MessageBox cu datele de contact.
* Exit: Îi permite utilizatorului să închidă aplicația.

Alte caracteristici:

* Drag&drop: oferind posibilitatea utilizatorului să o poziționeze pe ecran dupa bunul plac.
* Actualizare în timp real: O dată la câteva secunde, aplicația trimite requesturi catre pe backend pentru a putea vedea toate modificările aduse cărților.
* Buton “minimizare”: fereasta va fi minimizată și poate fi ușor accesibilă din desktop bar.
* Buton “exit”: Închide aplicația, similar cu butonul de Exit, butonul în plus oferă o viziune mai clară asupra acestui scop.

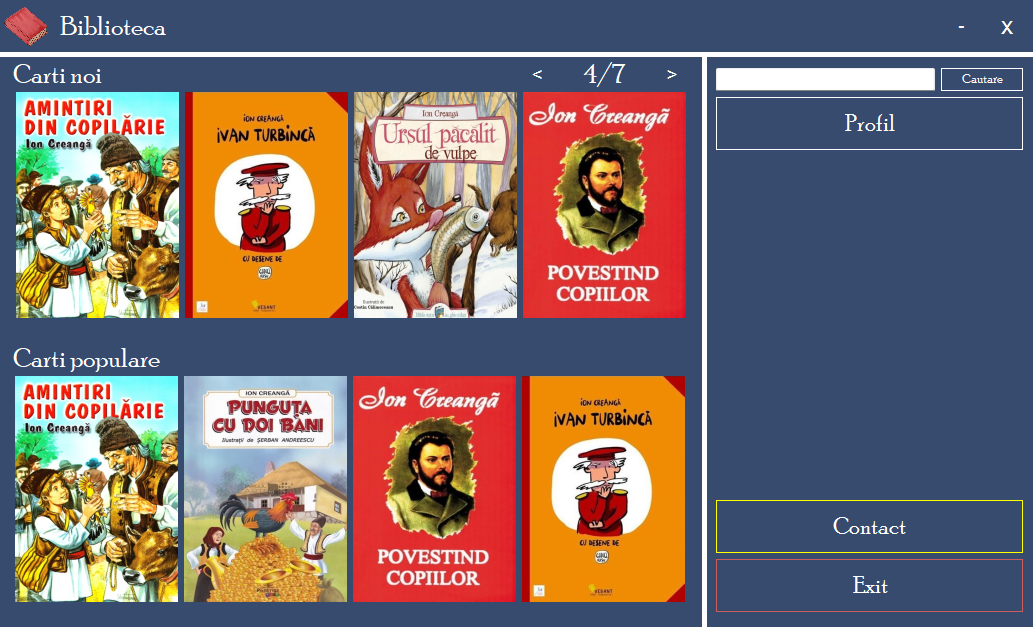


Fig. 4.1.2.1 Pagina Principala Utilizator

În figura 4.1.2.2 avem un exemplu de apelare al serviciului ce aduce cele mai populare cărți din baza de date. Variabila popularBooksCurrentNumber precum în cazul cărților noi, are ca scop menținerea numărului ultimei cărți afișate în listă. Implementarea actuală permite extinderea funcționalității actuale.

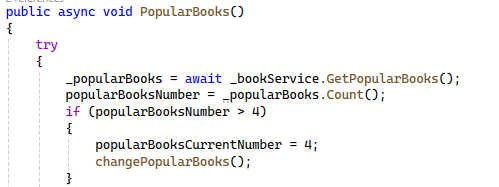


Fig. 4.1.2.2 Pagina Principala Utilizator

## 

## 4.3 Pagina principala Admin

Dupa autentificare, aplicația verifică în mod automat dacă utilizatorul logat are rol de admin, în caz afirmativ, va apărea un MessageBox ce va anunța calitatea de admin a utilizatorului și în interfață va avea acces funcțiile de admin, ce vor apărea în partea din dreapta cu chenar verde, acesta va avea acces de asemenea și la funcțile de bază ale utilizatorilor.

Funcțiile de admin sunt reprezentate de:

* Cereri utilizator – Va deschide formularul de cereri în care adminul poate vedea toate cererile de închiriere ale utilizatorilor.
* Adauga carte – Va deschide formularul de adaugare/editare carte
* Toate Recenziile – Va afișa toate recenziile, oferindui adminului posibilitatea să șteargă recenziile vulgare/nedorite

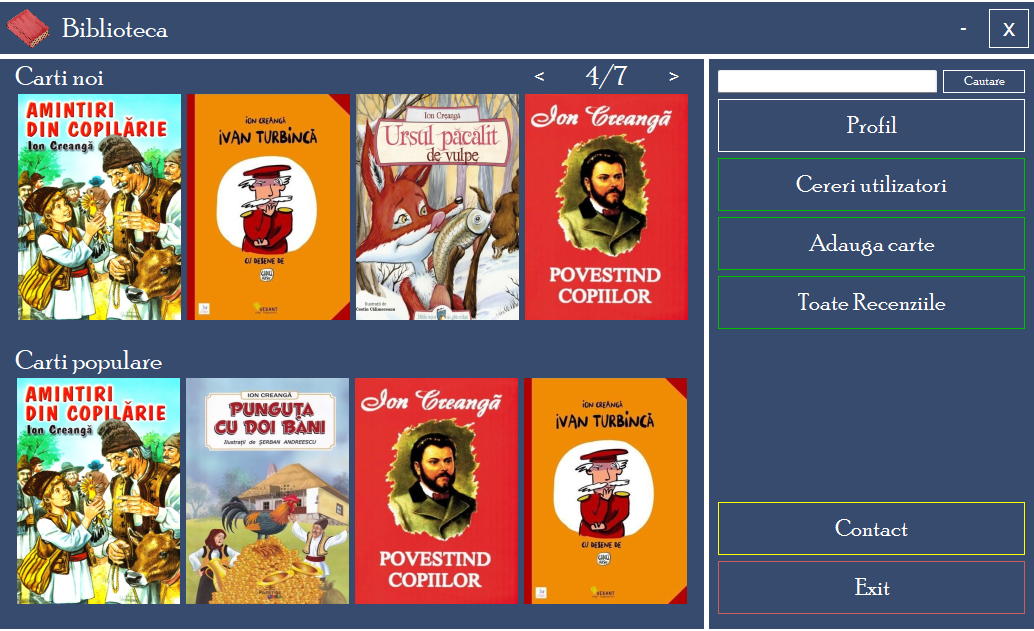


Fig. 4.1.3 Pagina Principala Admin

## 4.4 Funționalitatea căutare

Funția de căutare îi oferă utilizatorului posibilitatea să caute cărțile dupa titlu. Pentru realizarea acestei funcționalități am utilizat lista deja stocată în memorie la deschiderea aplicației, lucru ce simplifică foarte mult procesul și performanța, deoarece că nu este apelatată partea de backend pentru aducerea datelor. Lista în care sunt afișate sugestii legate de posibilele căutări este realizată pe același principiu, aceasta dispune de următoarele proprietăți:

* Se deschide dupa ce scriem ceva în componenta textBox.
* Daca dăm click în afară, aceasta va dispărea.
* Dacă apăsăm pe text-ul din listă, textBox-ul se va completa automat cu textul respectiv.
* Lista se actualizează la fiecare modificare din TextBox, rezultatele sunt bazate pe funcția „contain”

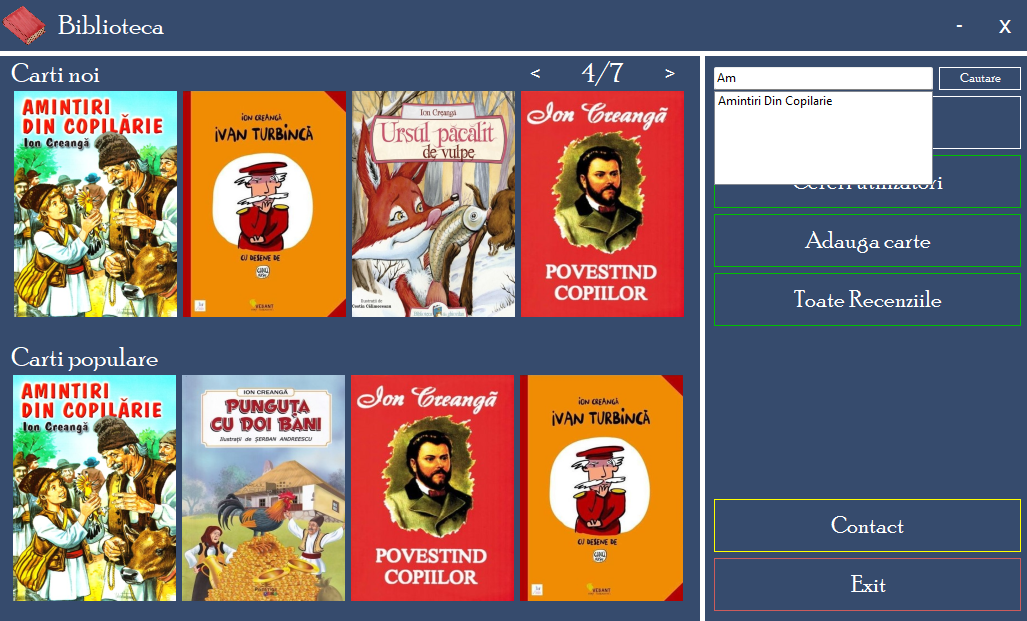


Fig. 4.1.4.1 Cautare În Pagina principală

În figura 4.1.4.2, este expus codul din spatele funcției de căutare, am utilizator metoda .Contains, pentru a găsi cărțile ce se potrivesc cu textul căutat.

După filtrarea rezultatelor, acestea sunt preluate de ListBox, de unde utilizatorul le poate selecta.

În figura 4.1.4.2, avem codul ce îi permite programului să transmită textul selectat din listBox către textBoxul în care este inițializată funcționalitatea de căutare.

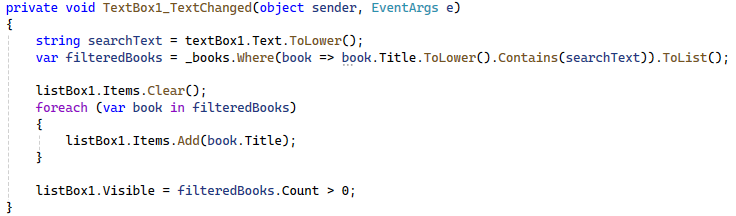


Fig. 4.1.4.2 Cautare În Pagina principală cod filtrare



Fig. 4.1.4.3 Cautare În Pagina principală cod alegere rezultat

## 4.5 Pagina Profil

## 4.5.1 Funționalități

În pagina de Profil, utilizatorul poate vizualiza cererile de închiriere și statusul lor într-un dataGridView. Principalele caracteristici ale paginii sunt reprezentate de:

* Drag&Drop: Utilizatorul poate poziționa pagina în ce mod dorește
* Minimizare: Prin intermediul butonului “-“
* Exit: Prin intermediul butonului “X”
* Schimbarea numelui: Prin intermediul butonului “Schimba numele”
* În formular apar numărul de cărți ce au fost citite de utilizator și numele acestuia



Fig. 4.1.5 Pagina Profil

## 4.5.2 Pop-up Schimbă Numele

În pagina de Profil, la apăsarea butonului “Schimbă Numele”, se va deschide un pop-up creat pe baza unui panel căruia îi este schimbată valoarea de visible.

La apăsarea butonului “Confirm”, va fi creat un apel ce va schimba numele utilizatorului, actualizânduse în timp real în aplicație.

La apăsarea butonului “X” panelul va devein invizibil pentru utilizator și va dispărea posibilitatea de a interacționa cu acesta.

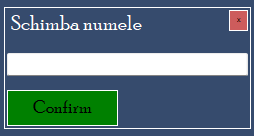


Fig. 4.1.6 Pop-up Schimba Numele

## 4.6 Pagina cereri utilizatori

Pagina cereri, are ca scop afișarea cererilor de închiriere a cărților de către utilizatori. Adminul are posibilitatea de a aproba sau refuza cererile create de un utilizator din aplicație.

În cazul în care cererile sunt aprobate, utilizatorul va putea vedea acest lucru în pagina de profil.

În cazul în care acestea sunt refuzate, cererea va fi ștearsă utilizând end-point-ul corespunzător iar utilizatorul nu va vedea cererea.

Daca în aplicație nu există cereri curente, va fi afișat un mesaj corespunzător.

Alte funții prezente:

* Drag&Drop: Utilizatorul poate poziționa pagina în ce mod dorește
* Minimizare: Prin intermediul butonului “-“
* Exit: Prin intermediul butonului “X”

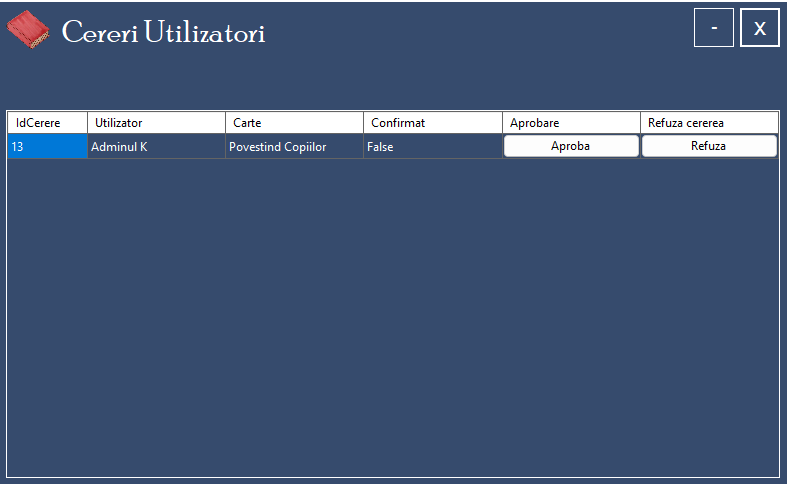
 

Fig. 4.1.6 Pop-up Schimba Numele

## 4.7 Pagina toate recenziile

Această pagină îi oferă adminului o viziune largă asupra recenziilor, acesta având posibilitatea de a le controla pe cele neadecvate sau pe cele considerate spam.

Pagina prezintă mai multe funcționalități:

* Vizualizare recenzie: prin intermediul unui richTextBox ce conține recenzia propriu zisă.
* Vizualizare nume
* Vizualizare notă: numărul de steluțe este egal cu nota lăsată de către utilizator în aplicații, unde nota maximă este 10 și cea minima este 1
* Butoane ce permit navigarea printre recenzii: Dupa apelarea serviciului de back-end, datele sunt stocate într-o listă de unde sunt ușor accesibile utilizând cele două butoane. Limitele listei sunt marcate de un messageBox ce anunță sfârșitul listei.

Alte funcții:

* Drag&Drop: Utilizatorul poate poziționa pagina în ce mod dorește
* Minimizare: Prin intermediul butonului “-“
* Exit: Prin intermediul butonului “X”



Fig. 4.1.7 Pagina toate recenziile

## 4.8 Pagina adaugă carte

## 4.8.1 Modul adăugare

Această pagină îi oferă adminului posibilitatea de a adăuga o nouă carte în baza de date. Formularul beneficiază de următoarele funții:

* PictureBox căruia îi este atașat un openFileDialog. În momentul în care se dă click pe pictureBox, utilizatorul va putea să aleagă o poză din memoria calculatorului, la momentul confirmării, aceasta va fi trecută prin filtrul de extensie și programul va valida imaginea încărcată, dacă aceasta trece de filtre cu succes, va fi afișată în pictureBox.
* TextBox pentru titlu cu validator pentru lungime.
* TextBox pentru numărul de pagini, cu validator pentru a primi numai cifre.
* DataPicker ce nu poate primi date calendaristice din viitor.
* TextBox pentru editură cu validator pentru lungime.
* ComboBox pentru Autor, ce va fi populat cu date la încărcarea formularului, acesta este actualizat în timp real o dată cu funcția de adăugare autor.
* ComboBox pentru limbă, va încărca datele la deschiderea formularului și va fii actualizat în timp real.
* RichTextBox pentru descriere cu validatoare pentru lungime.



Fig. 4.1.8 Adaugă Carte



Fig. 4.1.9 Adaugă Carte Completat

* La apăsarea butonului ștergere, datele completate vor fi șterse și imaginea din pictureBox va fi setată la valoarea inițială.
* Butonul “Adaugă” va face apelul către baza de date unde se face încă o validare a datelor.
* Butonul “Adaugă autor” va deschide formularul de adăugare autor din componența formularului curent, acesta este făcut pe baza unui panel implicit având visible = false.
* Butonul “Adaugă limbă” va deschide formularul de adaugă limbă din componența formularului curent, acesta este făcut pe baza unui panel implicit având visible = false.

Alte funcții:

* Drag&Drop: Utilizatorul poate poziționa pagina în ce mod dorește
* Minimizare: Prin intermediul butonului “-“
* Exit: Prin intermediul butonului “X”

## 4.8.2 Modul editare

Deasupra obiectului de tip pictureBox este un comboBox în care sunt încărcate toate cărțile din baza de date în urma unui apel la deschiderea formularului.

La selectarea unei cărți din comboBox, programul va procesa cererea și va încărca datele cărții selectate în obiectele formularului, adițional față de formularul adăugare această funție prezinta:

* Buton “Înapoi” ce îi permite adminului să se întoarcă la modul de adăugare carte. Obiectele vor fi golite de date și intrarea din baza de date va rămâne nemodificată.
* Buton “Editează”: în momentul apăsării va aplica aceleași validări ca butonul adăugare pentru fiecare câmp în parte, acesta va apela un apel de tip “PUT” ce va actualiza intrarea din baza de date dacă nu apar erori legate de validare.
* Label ce va conține Id-ul cărții.
* Buton de ștergere, ce îi va permite adminului să elimine cartea din baza de date.



Fig. 4.1.9 Adaugă Carte Completat

## 4.8.3 Adaugă Autor

Formularul “Adaugă Autor” se află în componena “Adaugă carte”. Prin intermediul acestuia, adminul poate adăuga un nou Autor în baza de date.

* TextBox pentru nume cu validările relevante
* TextBox pentru prenume cu validările relevante
* DataPicker pentru a selecta data nașterii
* Buton “Ștergere” ce va goli câmpurile formularului.
* Buton “Adaugă” ce va trimite requestul pe backend după ce sunt realizate validările relevante.

Alte funcții:

* Drag&Drop: Utilizatorul poate poziționa pagina în ce mod dorește
* Exit: Prin intermediul butonului “X”

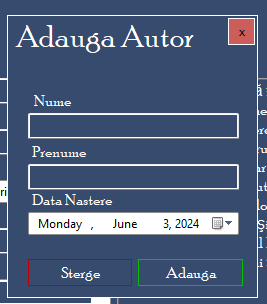


Fig. 4.1.10 Adaugă Autor

## 4.8.4 Adaugă Autor

Formularul “Adaugă Limbă” se află în componena “Adaugă carte”. Prin intermediul acestuia, adminul poate adăuga o nouă Limbă în baza de date.

Formarul dispune de următoarele funcții:

* TextBox pentru denumire cu validări relevante
* Buton “Șterge” ce va goli formularul
* Buton “Adăugare” ce va aplica validările relevante și va trimite requestul de adăugare.

Alte funcții:

* Drag&Drop: Utilizatorul poate poziționa pagina în ce mod dorește
* Exit: Prin intermediul butonului “X”

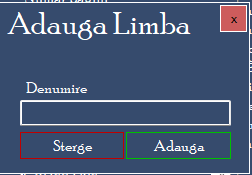


Fig. 4.1.11 Adaugă Limba

## 4.8.4 Funcție Adaugă Autor

În Figura 4.1.9.2. , putem observa cum imaginea adăugată a fost salvată în proiect utilizând comanda File.Copy din System.IO.  
 Dupa ce am copiat imaginea cu succes, adăugăm noua carte în baza de date cu ajutorul serviciului

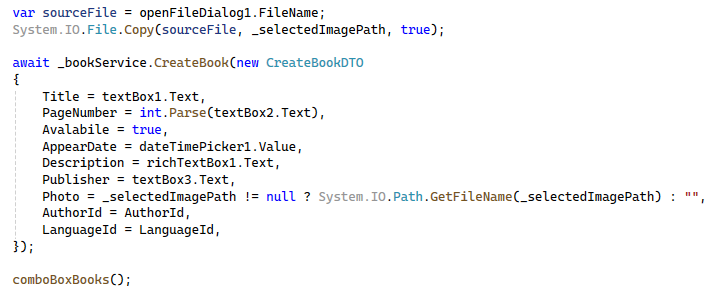


Fig. 4.1.9.2 Funcție adaugă carte

## 4.9 Vizualizare carte

## 4.9.1 Vizualizare

Formularul vizualizare carte îi va permite utilizatorului să vizualizeze cartea dorită. În această pagină avem mai multe obiecte care sunt populate în urma unui apel de GET dupa id-ul transmis din formularul precedent. Acesta dispune de:

Afișare carți

* PictureBox pentru imagine
* Pentru afișarea obiectelor au fost afișate label-uri simple cu cărora le-a fost setată culoarea textului la alb.
* RichTextBox pentru afișarea descrierii.
* Buton “Împrumută” ce îi permite utilizatorului să creeze o cerere de împrumut a cărții.
* Buton “Lasă Recenzie” ce va deschide formularul răspunzător de recenzii.



Fig. 4.1.12 Vizualizare Carte

## 4.9.2 Afișare recenzii

În partea de jos a formularului, utilizatorul poate vizualiza recenziile cărților. Acestea sunt aduse la deschiderea formularului de vizualizare.

* Funcționalitate ce specifică moementul în care cartea nu are nici o recenzie
* Panel ce va refleta nota recenziei prin intermediul steluțelor
* Butoane pentru a naviga printre recenzii
* RichTextBox pentru vizualizarea recenziei



Fig. 4.1.13 Vizualizare Carte - Recenzii

Alte funcții:

* Drag&Drop: Utilizatorul poate poziționa pagina în ce mod dorește
* Minimizare: Prin intermediul butonului “-“
* Exit: Prin intermediul butonului “X”

## 4.9.3 Împrumută carte

La apăsarea butonului împrumută, aplicația va crea un nou request în urma căruia va fi creată o nouă intrare în baza de date. Requesturile utilizatorilor pot fi văzute si acceptate de către admin. În momentul în care o cerere este creată cu succes, utilizatorul va primi un mesaj de confirmare.

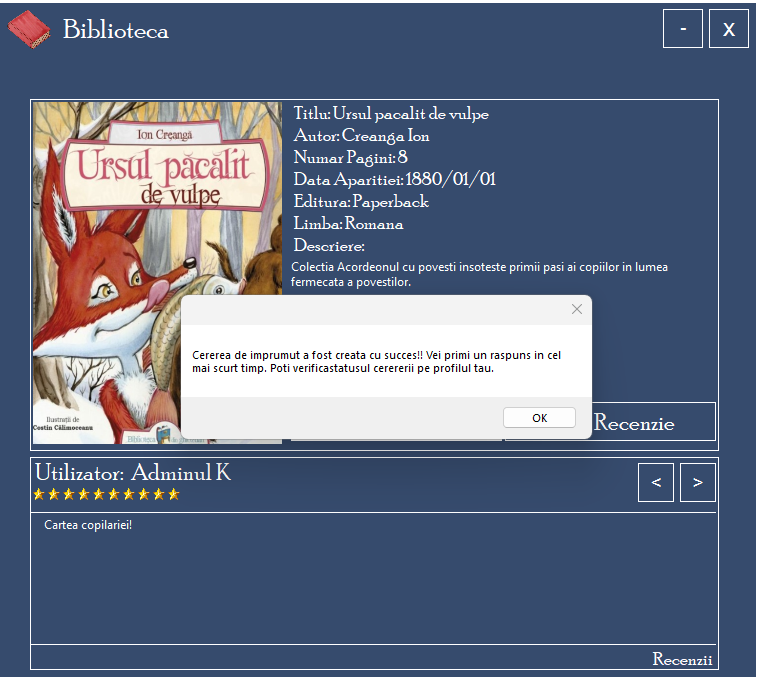


Fig. 4.1.14 Vizualizare Carte – Închiriere

## 4.9.4 Lasă Recenzie

Formularul “Lasă recenzie” îi permite utilizatorului să ofere un feedback privind cartea vizualizată. Formularul conține următoarele funcționalități:

* NumericBox pentru a selecta o notă de la 1 la 10, limitat de aceste valori(maxim si minim)
* RichTextBox pentru a scrie recenzie
* Buton Ștergere pentru a putea curăța câmpurile la nevoie
* Buton Lasă Recenzie pentru a trimite apelul către backend, unde se va face salvarea recenziei în baza de date

Alte funcții:

* Drag&Drop: Utilizatorul poate poziționa pagina în ce mod dorește
* Minimizare: Prin intermediul butonului “-“
* Exit: Prin intermediul butonului “X”

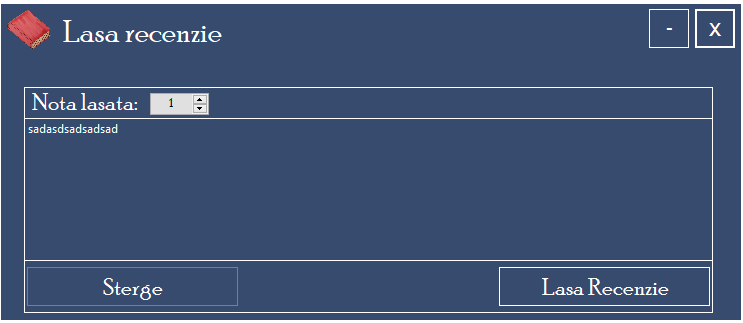
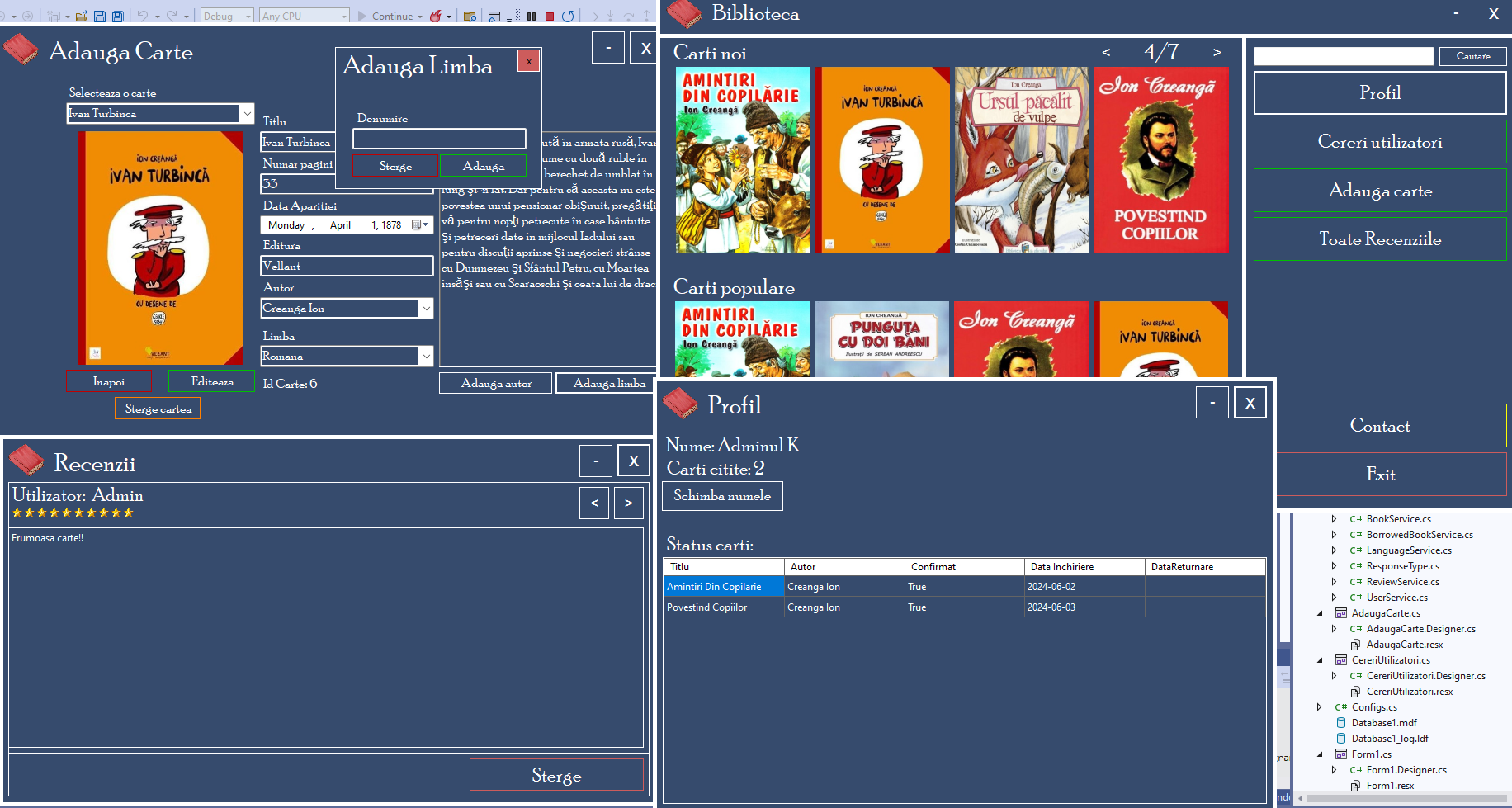


Fig. 4.1.14 Vizualizare Carte – Închiriere

## 4.10 Prezentare Generală

Aplicația folosește puține resurse, ceea ce-i permite adminului să opereze aplicația chiar și pe un sistem cu performanțe reduse. Interfața intuitivă permite managerierea fluidă a mai multor pagini simultan, cu actualizare în timp real între formulare. Programarea asincronă combinată cu gestionarea erorilor utilizează resursele procesorului într-un mod eficient evitând astfel blocajele de lung timp sau situațile în care aplicația s-ar putea închide automat într-un mod neașteptat.

 Fig. 4.1.15 Prezentare Generală

## 4.11 Schemă bază de date

Schema inițială a bazei de date a acoperit majoritatea funcționalităților, aceasta suferind foarte puține modificări ulterioare.

## 

Fig. 4.1.15 Prezentare Generală

# 5. Concluzie

Proiectul ilustrează un mod interactiv în care bibliotecile ar putea integra o modernizare legată de procesul de închiriere și management al cărților. O integrare inteligentă ar putea reduce timpul pe care utilizatorii îl petrec într-o bibliotecă de a găsi cărțile dorite. Din punct de vedere personal, proiectul de licență a reprezentat un mod de a recapitula și consolida toate cunoștințele pe care le-am cumulat în facultate, aducând și lucruri în plus.

# 6. Controlul versiunii

Pentru a avea acces la resursele aplicației am utilizat aplicația GitHub Desktop. [10]. Am ales această variantă deoarece, aplicația oferă o interfață grafică ușor de utilizate și la îndemâna oricui. În imaginea alăturată se pot vedea cele trei repository-uri, câte unul, pentru fiecare serviciu.

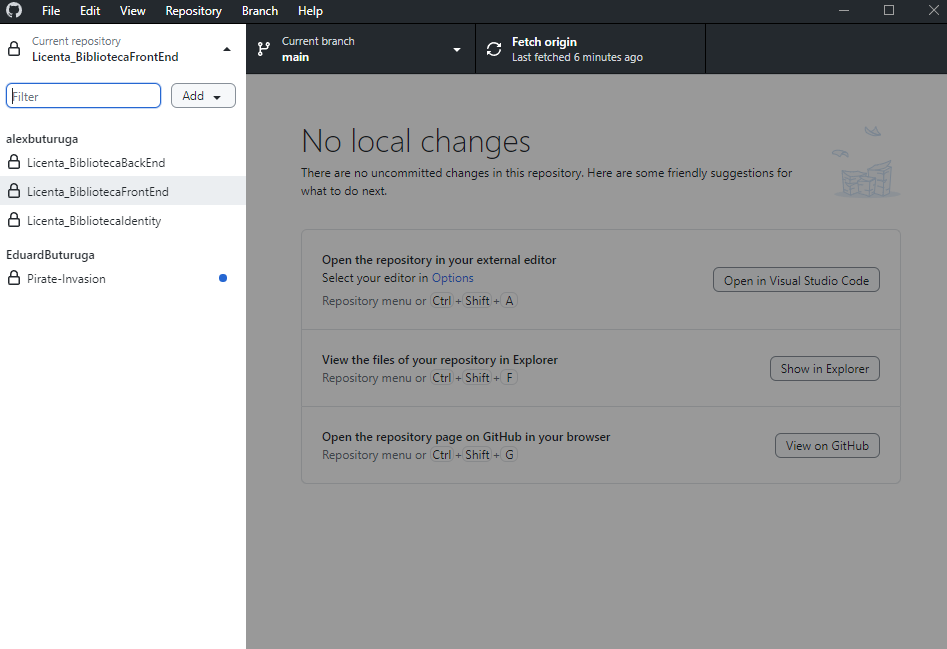


Fig. 4.1.16 Controlul versiunii

# 6. Bibliografie

[1] Bruce Policky, „The Evolution of Desktop Software”, URL: <https://www.tabs3.com/the-evolution-of-desktop-software.html>.

[2] ASP.NET Core documentation, ASP.NET Identity, URL: <https://learn.microsoft.com/en-us/aspnet/identity/>

[3] Windows Forms, URL: <https://microsoft.fandom.com/wiki/Windows_Forms>

https://www.tabs3.com/the-evolution-of-desktop-software.html

[4] Windows Desktop 8 Documentation, URL: <https://learn.microsoft.com/en-us/dotnet/api/system.windows.forms.button?view=windowsdesktop-8.0>

[5] ASP.NET Core documentațion: <https://dotnet.microsoft.com/en-us/apps/aspnet>

[6] Entity Framework Core documentațion: <https://fontmeme.com/fonts/poor-richard-font/>

[7] Identity FrameWork:

[https://learn.microsoft.com/en-us/aspnet/core/security/authentication/identity?view=aspnetcore-8.0&tabs=visual-studio](https://learn.microsoft.com/en-us/aspnet/core/security/authentication/identity?view=aspnetcore-8.0&tabs=visual-studio%20)

[8] Sql Server: <https://www.microsoft.com/en-us/sql-server/sql-server-downloads>

[9] NuGet: <https://learn.microsoft.com/en-us/nuget/>

[10] Git: <https://github.com/alexbuturuga>

[11] Visual Studio: <https://visualstudio.microsoft.com/vs/>

[12] Font utilizat, URL: <https://fontmeme.com/fonts/poor-richard-font/>