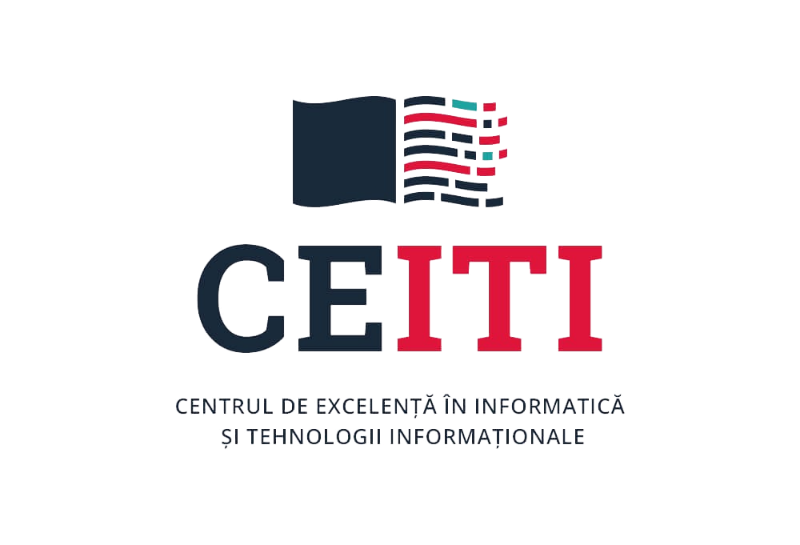
MINISTERUL EDUCAŢIEI ȘI CERCETAL REPUBLICII MOLDOVA

Centrul de Excelenţă în Informatică şi Tehnologii Informaţionale



**RAPORT**

**PENTRU PRACTICA TEHNOLOGICĂ**

**SPECIALITATEA** „Programarea și analiza produselor program”

A elaborat elevul: Darii Dumitru

Grupa: P-1932

Conducătorul practicii: Jumbei Olga

Chişinău

2022

# Cuprins

[**Cuprins**](#_qjr1qne5uqv) **2**

[**Introducere**](#_1t3h5sf) **4**

[Scopul practicii](#_vukul76epcdp) 4

[Obiective generale](#_c6t2q7vq5e54) 4

[**Modul de elaborare a aplicației**](#_2s8eyo1) **5**

[**Condiția sarcinii individuale**](#_hsjgiou41qxh) **6**

[Abilităților fiecărei interfețe de utilizator](#_1kl8vwdj5e1y) 7

[**Proiectarea bazei de date**](#_4i7ojhp) **8**

[Crearea și descrierea modelului relațional](#_xq7u0b2ffy4) 8

[Structura tabelului Edituri](#_af46jdu5bj63) 9

[Structura tabelului CategoriiGenuri](#_4p955r4sayp0) 9

[Structura tabelului Genuri](#_a4t5us95b4y) 9

[Structura tabelului TipuriCarti](#_5gpbbz8uwh4r) 10

[Structura tabelului StariStoc](#_xald80ddubdp) 10

[Structura tabelului Carti](#_nqtnghm01gze) 10

[Structura tabelului Autori](#_yyov3kxrxbxi) 11

[Structura tabelului CarteAutor](#_ktzarl40oldi) 11

[Structura tabelului Users](#_w4d8tnj6q40p) 12

[**Scenariul bazei de date**](#_hk3ofn4mx8am) **13**

[**Ghidul de utilizare al aplicației**](#_6cvpd3jplyhm) **21**

[**Descrierea aplicației**](#_vjzsl8berrn) **21**

[Autentificarea, înregistrarea și deconectarea](#_cq609y8h1pfm) 21

[Funcționalitățile aplicației](#_5un8dyoufefu) 26

[Pagina Catalog](#_39ds1jbzzu6u) 26

[Pagina Căutare](#_xrdrhvqg1w1t) 29

[Pagina Cărți Favorite](#_nv77qv4qfr4m) 31

[Pagina Administrare](#_njtpsk90nzyd) 33

[Pagina Adaugă Carte](#_xq2b4q6a4akg) 33

[Pagina Șterge Carte](#_uuff4ucypeji) 36

[**Concluzie**](#_1ci93xb) **39**

[**Bibliografie**](#_3whwml4) **40**

[**Anexe**](#_ebrzwumbe93v) **41**

[Formularul Main](#_7jv944in55v9) 41

[Butonul “Autentificare”:](#_jrku4zc6e6js) 41

[Butonul “Deconectare”](#_1bqacgrw2win) 41

[Formularul Autentificare](#_bm9em3febztr) 41

[Butonul “Log In”](#_70wyr8tfcntq) 41

[Butonul “Creare cont nou”](#_80dmx9b9vloi) 43

[Formularul Inregistrare](#_oc3nbhyirz14) 43

[Butonul “Sign Up:](#_liyob9b73gd0) 43

[UserControl-ul Catalog](#_kzdk1huac4ey) 45

[Evenimentul “dataGridViewCategoriiGenuri\_CellClick”](#_3gafoujfnsf0) 45

[Evenimentul “dataGridViewGenuri\_CellClick”](#_6542ziyzz3io) 45

[Evenimentul “dataGridViewCarti\_CellMouseDoubleClick”](#_2d6lvunt9m8x) 46

[Formularul BookInformation](#_x1os81ryt99j) 49

[Butonul Adaugă la Favorite](#_wnfzrz7j47gz) 49

[Butonul Șterge din Favorite](#_cuc419hctdj3) 49

[UserControl-ul Căutare](#_xh5ddprqb0ow) 50

[Bara de căutare](#_wfnbx6h245um) 50

[Butonul “Caută”](#_nryei5c547zb) 50

[Butonul “ClearSearch”](#_pp5qjdooc9ye) 50

[UserControl-ul Cărți Favorite](#_2go9v0fpy5m6) 50

[Evenimentul dataGridViewResult\_CellMouseDoubleClick](#_ku2576824mnm) 50

[Butonul “Exportă în Excel”](#_nf7aiwv1du9s) 50

[UserControl-ul “Administrare”](#_y1jgtea0k1z8) 55

[Butonul “Adaugă Carte”](#_y4bdnr34fm1) 55

[Butonul “Șterge Carte”](#_8b9pxrrvnjco) 55

[UserControl-ul “Adaugă carte”](#_8v7syxtbzffj) 55

[Butonul “Adaugă”](#_m3fayc22ufy8) 55

[UserControl-ul “Adaugă carte”](#_lil72cayedof) 58

[Butonul “Șterge”](#_bybpx9u25joo) 58

# Introducere

## Scopul practicii

Stagiul de practică reprezintă o modalitate de prosperare care dezvoltă abilitățile  
profesioniste. În cadrul acesteia se realizează un schimb de experiență care oferă noi viziuni legate de viitorul loc de muncă. Stagiul de practică servește la stabilirea dorințelor referitor la locul viitor de muncă. De asemenea, ne face cunoștință cu posibilele dificultăți care le vom putea întâlni pe parcursul activării în domeniul dat, precum și îmbunătățește capacitățile de colaborare în grup.

Stagiul de practică oferă șansa de a îmbina teoria cu practica. Este un avantaj foarte mare,  
întrucât pe parcursul anului de învățămînt nu se reușește să se ofere mai multă atenție practicii.

Îmbinarea dintre limbajul de programare C# și crearea bazelor de date este utilizată frecvent  
pe piață. De aceea, practica tehnologică are un impact favorabil asupra studenților datorită necesității dezvoltarea unor aplicații cerute pe piață.

## Obiective generale

* Consolidarea cunoștințelor acumulate legate de limbajul de programare C# și crearea bazelor de date prin SQL
* Dezvoltarea abilităților muncii independente
* Deprinderea cu regimul de lucru
* Axarea pe cerințele de programare și analiza produsului program
* Realizarea accesului la aplicația software în conformitate cu modulele și algoritmul aplicației
* Verificarea și securizarea aplicației, prin urmare asigurând confidențialitatea și integritatea datelor
* Întreținerea, manipularea și exploatarea corectă a aplicației elaborate
* Conștientizarea elementelor și funcțiilor specifice specialității
  + Proiectarea interfețelor aplicației
  + Respectarea procedurilor de către utilizator
  + Elaborarea algoritmilor
  + Utilizarea control-elor
  + Utilizarea meniurilor
  + Elaborarea rapoartelor

# Modul de elaborare a aplicației

Pentru realizarea acestei sarcini au fost utilizate următoarele instrumente:

* IDE Microsoft Visual Studio (reprezintă un mediu de dezvoltare integrat de la Microsoft, predestinat pentru dezvoltarea diferitor tipuri de aplicații)
* Platforma .NET (predestinată dezvoltatorilor, alcătuită din instrumente, limbaje de programare și biblioteci pentru dezvoltarea aplicațiilor )
* Limbajul de programare C# (predestinat dezvoltării de aplicații, pe platforma .NET)
* Framework-ul UI Windows Forms (reprezintă o bibliotecă de clase grafice pentru Windows)
* Microsoft SQLServer (server de baze de date relaționale dezvoltat de Microsoft)
* Microsoft SQL Server Management Studio (utilizat la configurarea, gestionarea și administrarea tuturor componentelor din Microsoft SQL Server)
* Limbajul T-SQL (extensie a limbajului SQL, utilizată pentru administrarea bazelor de date)

Instrumente opționale utilizate:

* Librăria .NET IronXL (predestinată pentru exportarea datelor în format .xlsx, adică Excel)
* Sistemul de control de versiuni GIT (predestinat facilitării lucrului asupra aplicației și administrarea versiunilor programului)
* GitHub.com - Hosting-ul de proiecte software, care utilizează GIT (predestinat pentru stocarea la distanță a fișierelor proiectului și pentru administrarea versiunilor în mod interactiv)

Părțile de bază ale interfeței grafice:

* Meniul este realizat printr-un sidebar, situat în stânga, care conține butoane cu linkuri către paginile aplicației.
* Zona de conținut este alăturată meniului, aceasta servind drept placeholder pentru paginile deschise în urma accesării link-urilor din meniu.
* Înregistrarea, autentificarea și detaliile despre cărțile se deschid la necesitate în ferestre separate.

# Condiția sarcinii individuale

Drept sarcină pentru acest stagiu de practică am ales să realizez o aplicație predestinată spre utilizare librăriei [Librarius](https://librarius.md/ro/), care reprezintă o librărie populară din Moldova. Aplicația dispune de două interfețe: una pentru Clienți și alta pentru Manageri.

Scopul principal al aplicației este de a permite Clienților și Administratorilor de a vizualiza cărțile din librăria Librarius și de a efectua unele operații asupra lor, care vor fi descrise mai jos. Aceste funcționalități vor putea fi accesate prin intermediul unui meniu.

Funcționalitățile aplicației:

* Înregistrarea și autentificarea utilizatorilor
* Vizualizarea catalogului de cărți al librăriei
* Obținerea detaliilor pentru fiecare carte din librărie
* Căutarea de cărți în baza a mai multor filtre
* Adăugarea cărților dorite la lista de cărți favorite
* Ștergerea că cărților dorite din lista de cărți favorite
* Oferirea a două interfețe: una pentru Clienți și alta pentru Administratori
* Adăugarea sau ștergerea cărților din baza de date, disponibilă pentru Administratori
* Exportarea datelor din lista de cărți favorite în Excel (fișier .xlsx)

## Abilităților fiecărei interfețe de utilizator

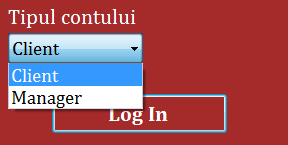
# 

Abilitățile clienților:

* Dispun de înregistrare
* Dispun de autentificare
* Pot vizualiza catalogul de cărți
* Pot obține detalii pentru fiecare carte
* Pot căuta cărți în baza la mai multe filtre
* Pot adăuga cărți la lista de cărți favorite
* Pot șterge cărți din lista de cărți favorite
* Pot exporta date din lista de cărți favorite în Excel (fișier .xlsx)

Abilitățile managerilor:

* Dispun de toate abilitățile clienților
* Au acces la modificarea bazei de date a librăriei
* Pot adăuga cărți noi în baza de date
* Pot șterge cărți din baza de date



1

Figura 0. Selectare tipului de cont la autentificare,

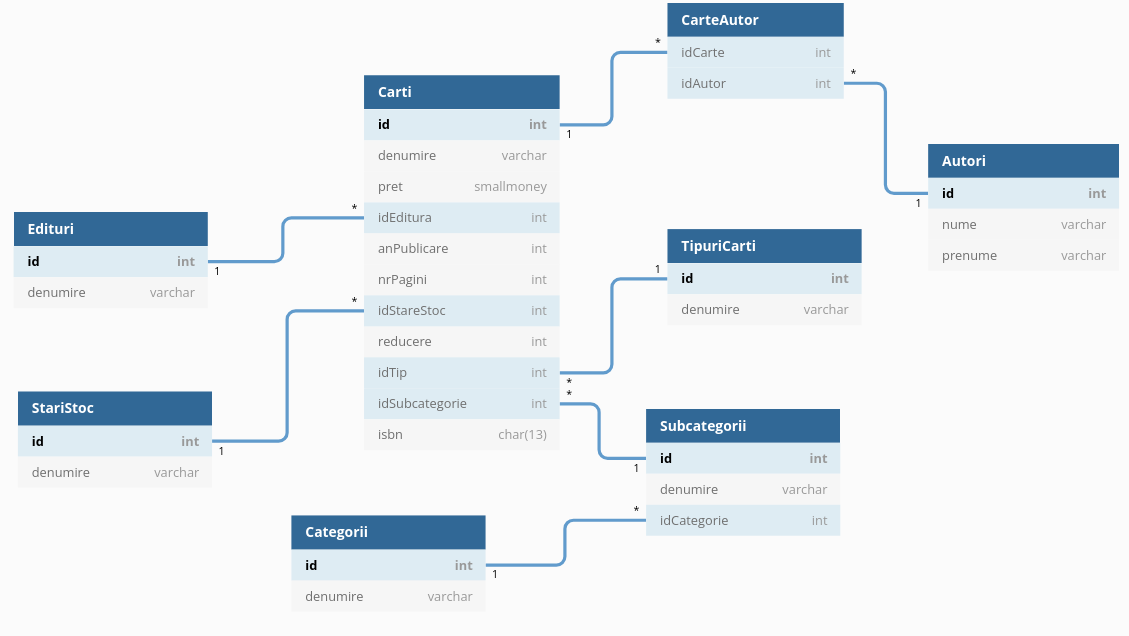
pentru accesarea uneia din cele două interfețe de utilizator

# Proiectarea bazei de date

Baza de date este numită „Librarius”. Aceasta conține 8 tabele predestinate stocării informației despre cărțile din librăriei și 1 tabel predestinat stocării datelor utilizatorilor înregistrați în aplicație.

Tabelele:

* Edituri, conține editurile care publică cărțile
* CategoriiGenuri, conține categoriile de genuri specifice cărților
* Genuri, conține genurile din care fac parte cărțile
* TipuriCarti, conține tipurile care pot fi atribuite cărților
* StariStoc, conține stările în care poate fi stocul unei anumite cărți
* Carti, conține cărțile și informația legată de ele
* Autori, conține autorii cărților
* CarteAutor, conține perechile de autori și cărți
* Users, conține datele utilizatorilor înregistrați în aplicație

Relațiile dintre tabele:

## Crearea și descrierea modelului relațional

Figura 1. Diagrama Entitate - Relație

### Structura tabelului Edituri

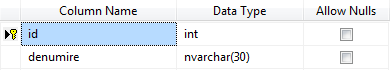


Figura 2. Structura tabelului Edituri

Descrierea câmpurilor:

id - cheia primară

denumire - denumirea editurii

### Structura tabelului CategoriiGenuri

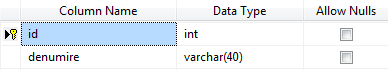


Figura 2. Structura tabelului CategoriiGenuri

# 

Descrierea câmpurilor:

id - cheia primară

denumire - denumirea categoriei de genuri de carte

### Structura tabelului Genuri

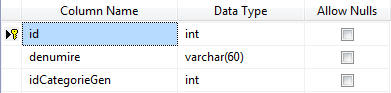


Figura 3. Structura tabelului Genuri

Descrierea câmpurilor:

id - cheia primară

denumire - denumirea genului de carte

idCategorieGen - cheia secundară care face referință la tabelul CategoriiGenuri

### Structura tabelului TipuriCarti

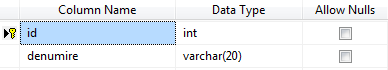


Figura 4. Structura tabelului TipuriCarti

Descrierea câmpurilor:

id - cheia primară

denumire - denumirea tipului de carte

### Structura tabelului StariStoc

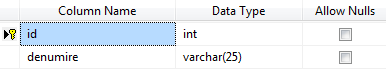


Figura 5. Structura tabelului StariStoc

Descrierea câmpurilor:

id - cheia primară

denumire - denumirea stării de stoc a cărții

### Structura tabelului Carti

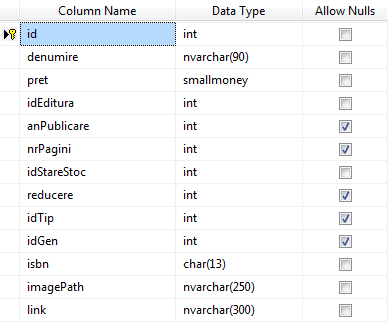


Figura 6. Structura tabelului Carti

Descrierea câmpurilor:

id - cheia primară

denumire - denumirea cărții

pret - prețul cărții

idEditura - cheie secundară care face referință la tabelul Edituri

anPublicare - anul publicării cărții de către editură

nrPagini - numărul de pagini

idStareStoc - cheie secundară care face referință la tabelul StariStoc

reducere - reducerea de care beneficiază cartea

idTip - cheie secundară care face referință la tabelul TipuriCarti

idGen - cheie secundară care face referință la tabelul Genuri

isbn - numărul unic ISBN din 13 cifre

imagePath - denumirea fișierului imaginii din folderul proiectului *images*

link - link-ul spre carte pe web site-ul librarius.md

### Structura tabelului Autori

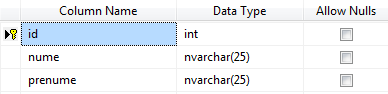


Figura 7. Structura tabelului Autori

Descrierea câmpurilor:

id - cheia primară

nume - numele autorului

prenume - prenumele autorului

### Structura tabelului CarteAutor

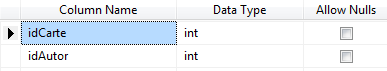


Figura 8. Structura tabelului CarteAutor

Descrierea câmpurilor:

idCarte - cheie secundară care face referință la tabelul Carti

idAutor - cheie secundară care face referință la tabelul Autori

### Structura tabelului Users

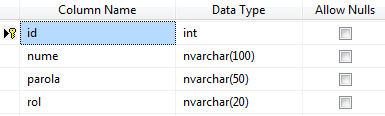


Figura 9. Structura tabelului Users

Descrierea câmpurilor:

id - cheia primară

nume - numele utilizatorului

parola - parola utilizatorului

rol - rolul utilizatorului (Client sau Manager)

# Scenariul bazei de date

-- Se conecteaza la baza de date master pentru a executa instructiunile de mai jos.

USE master

GO

-- use master; alter database Librarius set single\_user with rollback immediate; --drop database Librarius

-- Se creeaza o noua baza de date cu numele de 'Librarie\_Darii\_Dumitru'.

IF NOT EXISTS (

SELECT name

FROM sys.databases

WHERE name = 'Librarius'

)

CREATE DATABASE Librarius

GO

-- Se conecteaza la baza de date 'Librarie\_Darii\_Dumitru' pentru a o putea modifica.

USE Librarius

GO

-- Tema: Evidenta cartilor librariei Librarius

-- Crearea tabelelor bazei de date

/\* Tabelul Edituri contine editurile care editeaza cartile.

\* Descrierea colonelor:

\* id - id-ul editurii

\* denumire - denumirea editurii

\*/

CREATE TABLE Edituri

(

id INTEGER IDENTITY,

denumire NVARCHAR(30) UNIQUE NOT NULL,

PRIMARY KEY(id)

);

/\* Tabelul CategoriiGenuri contine categoriile de genuri specifice cartilor.

\* Descrierea colonelor:

\* id - id-ul categoriei

\* denumire - denumirea categoriei de gen de carte

\*/

CREATE TABLE CategoriiGenuri

(

id INTEGER,

denumire VARCHAR(40) UNIQUE NOT NULL,

PRIMARY KEY(id)

);

/\* Tabelul Genuri contine genurile din care fac parte cartile.

\* Descrierea colonelor:

\* id - id-ul genului

\* denumire - denumirea genului

\* idCategorieGen - id-ul categoriei din care face parte genul

\*/

CREATE TABLE Genuri

(

id INTEGER,

denumire VARCHAR(60) NOT NULL,

idCategorieGen INTEGER NOT NULL,

PRIMARY KEY(id),

);

/\* Tabelul TipuriCarti contine tipurile care pot fi atribuite cartilor.

\* Descrierea colonelor:

\* id - id-ul tipului de carte

\* denumire - denumirea tipului de carte

\*/

CREATE TABLE TipuriCarti

(

id INTEGER,

denumire VARCHAR(20) NOT NULL,

PRIMARY KEY(id)

);

/\* Tabelul StariStoc contine starile in care poate fi stocul unei anumite carti.

\* Descrierea colonelor:

\* id - id-ul starii de stoc a cartii

\* denumire - denumirea starii de stoc a cartii

\*/

CREATE TABLE StariStoc

(

id INTEGER,

denumire VARCHAR(25) UNIQUE NOT NULL,

PRIMARY KEY(id)

);

/\* Tabelul Carti contine cartile si informatia legata de ele.

\* Descrierea colonelor:

\* id - id-ul cartii

\* denumire - denumirea cartii

\* anPublicare - anul cand a fost publicata cartea

\* nrPagini - numarul de pagini ale cartii

\* pret - pretul cartii

\* idEditura - id-ul editurii cartii

\* idGen - id-ul genului din care face parte cartea

\* isbn - International Standard Book Number din 13 cifre, atribuit cartii

\*/

CREATE TABLE Carti

(

id INTEGER IDENTITY,

denumire NVARCHAR(90) NOT NULL,

pret SMALLMONEY NOT NULL,

idEditura INTEGER NOT NULL,

anPublicare INTEGER,

nrPagini INTEGER,

idStareStoc INTEGER NOT NULL,

reducere INTEGER CHECK((reducere > 0) AND (reducere < 100)),

idTip INTEGER,

idGen INTEGER,

isbn CHAR(13) NOT NULL,

imagePath NVARCHAR(250) NOT NULL,

link NVARCHAR(300) NOT NULL,

PRIMARY KEY(id)

);

/\* Tabelul Autori contine autorii cartilor.

\* Descrierea colonelor:

\* id - id-ul autorului

\* nume - numele autorului

\* prenume - prenumele autorului

\*/

CREATE TABLE Autori

(

id INTEGER IDENTITY,

nume NVARCHAR(25) NOT NULL,

prenume NVARCHAR(25) NOT NULL,

PRIMARY KEY(id)

);

/\* Tabelul CarteAutor contine perechile de autori si carti.

\* Descrierea colonelor:

\* idCarte - id-ul cartii din tabelul Carti

\* idAutor - id-ul autorului din tabelul Autori

\*/

CREATE TABLE CarteAutor

(

idCarte INTEGER NOT NULL,

idAutor INTEGER NOT NULL,

);

CREATE TABLE Users

(

id INT IDENTITY,

nume NVARCHAR(100) UNIQUE not null,

parola NVARCHAR(50) not null,

rol NVARCHAR(20) check (rol = 'Client' or rol = 'Manager') not null,

PRIMARY KEY(id)

);

GO

-- Crearea legaturilor dintre tabele

-- Genuri

ALTER TABLE Genuri

ADD CONSTRAINT FK\_Gen\_CategorieGen

FOREIGN KEY(idCategorieGen) REFERENCES CategoriiGenuri(id);

-- Carti

ALTER TABLE Carti

ADD CONSTRAINT FK\_Carte\_Editura

FOREIGN KEY(idEditura) REFERENCES Edituri(id);

ALTER TABLE Carti

ADD CONSTRAINT FK\_Carte\_Gen

FOREIGN KEY(idGen) REFERENCES Genuri(id);

ALTER TABLE Carti

ADD CONSTRAINT FK\_Carte\_TipCarte

FOREIGN KEY(idTip) REFERENCES TipuriCarti(id);

ALTER TABLE Carti

ADD CONSTRAINT FK\_Carte\_StareStoc

FOREIGN KEY(idStareStoc) REFERENCES StariStoc(id);

-- CarteAutor

ALTER TABLE CarteAutor

ADD CONSTRAINT FK\_CarteAutor\_Carte

FOREIGN KEY(idCarte) REFERENCES Carti(id);

ALTER TABLE CarteAutor

ADD CONSTRAINT FK\_CarteAutor\_Autor

FOREIGN KEY(idAutor) REFERENCES Autori(id);

GO

-- Introducerea datelor in tabele

INSERT INTO Edituri

( denumire)

VALUES

(N'Curtea Veche')

,(N'Europress')

,(N'Vremea')

,(N'Trei')

,(N'All')

,(N'Эксмо')

,(N'Flamingo')

,(N'Teora')

,(N'Niculescu')

,(N'Картография')

,(N'Steaua Nordului')

,(N'Poligraf-Design');

INSERT INTO StariStoc

(id, denumire)

VALUES

(1,'Ultima carte')

,(2,'Stoc suficient')

,(3,'Stoc limitat')

,(4,'Stoc epuizat');

INSERT INTO TipuriCarti

(id, denumire)

VALUES

(1, 'Brosata')

,(2, 'Cartonata');

INSERT INTO CategoriiGenuri

(id, denumire)

VALUES

(1,'Beletristica')

,(2,'Carte practica,specialitate')

,(3,'Carti pentru copii')

,(4,'Dictionare')

,(5,'Harti Atlase Calendare Agende')

,(6,'Limbi straine')

,(7,'Manuale si auxiliare scolare');

INSERT INTO Genuri

(id, denumire, idCategorieGen)

VALUES

(1,'Proza , Dramaturgie',1)

,(2,'Roman',1)

,(3,'Fictiune,mistica,fantazy',1)

,(4,'Economie',2)

,(5,'Management',2)

,(6,'Computer,internet',2)

,(7,'Literatura pentru copii 7+',3)

,(8,'Poezii,povesti,povestiri',3)

,(9,'Limba engleza',4)

,(10,'Specialitate',4)

,(11,'Harti',5)

,(12,'Limba germana',6)

,(13,'Gradinita',7);

INSERT INTO Carti

(denumire, pret, idEditura, anPublicare, nrPagini, idStareStoc, reducere, idTip, idGen, isbn, imagePath, link)

VALUES

(N'Kathie si hipopotamul',132.0000,1,2010,264,1,NULL,1,1,'9786065880702', 'images\kathie-si-hipopotamul.jpg', 'https://librarius.md/ro/book/000744-kathie-si-hipopotamul')

,(N'Drumul spre Biserica',60.0000,2,2013,312,1,NULL,1,2,'9789975444606', 'images\drumul-spre-biserica.jpg', 'https://librarius.md/ro/book/drumul-spre-biserica-348165')

,(N'Venus ia foc Marte e de gheata',235.0000,3,2013,232,2,NULL,1,3,'9789736455575', 'images\venus-ia-foc.jpg', 'https://librarius.md/ro/book/venus-ia-foc-marte-e-de-ghiata-326823')

,(N'Cu ochii mintii',170.0000,4,2016,216,1,30,1,3,'9786067198157', 'images\cu-ochii-mintii.jpg', 'https://librarius.md/ro/book/cu-ochii-intii-474355')

,(N'In apararea pietelor',185.0000,1,2012,416,3,NULL,1,4,'9786065883239', 'images\in-apararea-pietelor.jpg', 'https://librarius.md/ro/book/000260-in-apararea-pietelor')

,(N'Cele 12 elemente ale managementului performant',155.0000,5,2009,272,3,NULL,1,5,'9789737241887', 'images\12-elemente-management.jpg', 'https://librarius.md/ro/book/007339-cele-12-elemente-ale-managementului-performant')

,(N'Adobe InDesign CC. Официальный учебный курс (+ CD-ROM)',740.0000,6,2014,496,1,NULL,1,6,'9785699696598', 'images\adobe-indesign.jpg', 'https://librarius.md/ro/book/006182-adobe-indesign-cc-oficialinyy-uchebnyy-kurs-cd')

,(N'Щенок Элфи или Не хочу быть один!',65.0000,6,2014,144,3,NULL,2,7,'9785699711932', 'images\shenok-elfi.jpg', 'https://librarius.md/ro/book/018514-schenok-elfi-ili-ne-hochu-byti-odin')

,(N'Povesti in romana si germana',174.0000,7,2010,104,2,70,2,8,'9789738873308', 'images\povesti-ro-germ.jpg', 'https://librarius.md/ro/book/001372-povesti-in-romana-si-germana')

,(N'Dictionar roman-englez',450.0000,8,NULL,NULL,2,NULL,2,9,'9789736019999', 'images\dictionar-ro-en.jpg', 'https://librarius.md/ro/book/dictionar-roman-englez-496328')

,(N'Dictionar Oxford de istorie universala contemporana (2 vol.)',395.0000,5,2007,504,1,NULL,1,10,'9789735715519', 'images\dictionar-oxford.jpg', 'https://librarius.md/ro/book/007666-dictionar-oxford-de-istorie-universala-contemporana-2-vol')

,(N'Harta Bulgaria',106.0000,9,2006,2,2,NULL,1,11,'9789737480804', 'images\harta-bulgaria.jpg', 'https://librarius.md/ro/book/007787-harta-bulgaria')

,(N'Политическая карта мира 1:35000000',135.0000,10,2011,NULL,2,NULL,1,11,'9789669460608', 'images\harta-politica-lume.jpg', 'https://librarius.md/ro/book/politicheskaya-karta-mira-135000000-130527')

,(N'Ghid de conversatie roman–german',60.0000,11,NULL,NULL,2,NULL,1,12,'9786065114685', 'images\ghid-convers-ro-germ.jpg', 'https://librarius.md/ro/book/007688-ghid-de-conversatie-roman-german')

,(N'Abecedar 6-7 ani',45.0000,12,2014,NULL,2,NULL,1,13,'9789975140379', 'images\abecedar-6-7.jpg', 'https://librarius.md/ro/book/abecedar-6-7-ani-ed-2014-poligraf-design-372486');

INSERT INTO Autori

(nume, prenume)

VALUES

(N'Vargas Llosa',N'Mario')

,(N'Dabija',N'Nicolae')

,(N'Gray',N'John')

,(N'Dashner',N'James')

,(N'James',N'Lucian')

,(N'Wagner',N'R.')

,(N'Круз',N'Дж.')

,(N'Вебб',N'Х.')

,(N'Sojka',N'Anna')

,(N'Levitchi',N'Leon')

,(N'Palmowski',N'J.')

,(N'Huber',N'Niculescu')

,(N'Puscasiu',N'O.')

,(N'Jelescu',N'P.');

INSERT INTO CarteAutor

(idCarte,idAutor)

VALUES

(1,1)

,(2,2)

,(3,3)

,(4,4)

,(5,5)

,(6,6)

,(7,7)

,(8,8)

,(9,9)

,(10,10)

,(11,11)

,(12,12)

,(14,13)

,(15,14);

GO

-- Vedere care contine categoriile si genurile corespunzatoare.

CREATE VIEW vCategorieGen\_Gen

AS

SELECT c.denumire AS Categorie, g.denumire AS Gen

FROM Genuri g

JOIN CategoriiGenuri c ON (g.id = c.id);

GO

-- Vedere care contine informatia generala despre fiecare carte

CREATE VIEW vCartiInfo

AS

SELECT c.denumire AS "Cartea", c.pret AS "Pretul cartii", e.denumire AS "Editura",

c.anPublicare AS "Anul publicarii", nrPagini,

ss.denumire AS "Starea stocului", c.reducere, t.denumire AS "Tipul", Genuri.denumire AS "Gen", c.isbn,

CONCAT(a.nume,' ', a.prenume) as "Autor", c.imagePath as "ImagePath", c.link as "Link"

FROM Carti c

INNER JOIN Edituri e ON (e.id = c.idEditura)

INNER JOIN StariStoc ss ON (ss.id = c.idStareStoc)

INNER JOIN TipuriCarti t ON (t.id = c.idTip)

INNER JOIN Genuri ON (Genuri.id = c.idGen)

LEFT JOIN CarteAutor ca ON (ca.idCarte = c.id)

LEFT JOIN Autori a ON (a.id = ca.idAutor);

GO

-- Vedere care contine Genurile si Categoriile combinate impreuna.

CREATE VIEW vGenuriUnionCategoriiGenuri

AS

SELECT Genuri.denumire

FROM Genuri

UNION ALL

SELECT CategoriiGenuri.denumire

FROM CategoriiGenuri;

GO

GO

-- Proceduri

GO

create procedure spCartiAutoriInBazaEditurii

@editura nvarchar = N'vremea'

AS

SELECT c.denumire "carte", a.nume "nume autor", a.prenume "prenume autor"

FROM CarteAutor ca

INNER JOIN Carti c ON (c.id = ca.idCarte)

INNER JOIN Autori a ON (a.id = ca.idAutor)

INNER JOIN Edituri e ON (e.id = c.idEditura)

WHERE e.denumire like @editura

GO

execute spCartiAutoriInBazaEditurii

go

create procedure sp\_insertCategorieGen

@id int,

@denumire NVARCHAR(40)

as

INSERT INTO CategoriiGenuri

(id, denumire)

VALUES

(@id, @denumire)

GO

create procedure sp\_insertGen

@id int,

@denumire NVARCHAR(60),

@idCategorieGen int

as

INSERT INTO Genuri

(id, denumire, idCategorieGen)

VALUES

(@id, @denumire, @idCategorieGen)

go

create procedure sp\_deleteBookFromCarteAutor

@bookName NVARCHAR(100)

as

delete from CarteAutor where idCarte = (select id from Carti where denumire like @bookName);

GO

create procedure sp\_removeBook

@denumireCarte NVARCHAR(90)

as

begin

exec sp\_deleteBookFromCarteAutor @denumireCarte

delete from Carti where denumire like @denumireCarte;

end

GO

create procedure sp\_addAuthor

@lastName NVARCHAR(25),

@firstName NVARCHAR(25)

as

insert into Autori(nume, prenume) values (@lastName, @firstName);

GO

create procedure sp\_addPublisher

@name NVARCHAR(30)

as

insert into Edituri(denumire) values (@name);

GO

create procedure sp\_addBook

@id int,

@name nvarchar(90),

@price smallmoney,

@publisherId int,

@publishingYear int,

@pages int,

@stockStateId int,

@discount int,

@typeId int,

@genreId int,

@isbn char(13),

@authorId int

as

begin

declare @imagePath nvarchar(30) = 'images\default.png'

declare @link nvarchar(100) = 'https://librarius.md/ro/'

if @discount = 0

set @discount = null

insert into Carti(denumire, pret, idEditura, anPublicare, nrPagini, idStareStoc, reducere, idTip, idGen, isbn, imagePath, link)

values (@name, @price, @publisherId, @publishingYear, @pages, @stockStateId, @discount, @typeId, @genreId, @isbn, @imagePath, @link);

insert into CarteAutor(idCarte, idAutor) values (@id, @authorId);

return;

end

GO

# Ghidul de utilizare al aplicației

Cerințe minime hardware:

* Procesor 1.5 GHz
* RAM: 200 MB
* Spațiu pe disk: 500 MB

Cerințe minime software:

* Windows 7 și mai nou
* .NET Framework 4.5 și mai nou

# Descrierea aplicației

## Autentificarea, înregistrarea și deconectarea

**Primul formular** care se deschide la executarea aplicației se numește [**Main**](#gmcvuokz3r9r), acesta reprezentând **Pagina principală**. Aici utilizatorul întâlnește în partea dreaptă un meniu de tip sidebar cu butoane spre alte pagini. În zona cu conținut acesta are opțiunea de a se autentifica, apăsând pe butonul [Autentificare](#tsffpekv6hrk). Fără autentificare, utilizatorul NU POATE accesa paginile aplicației.

# 

# 

Figura 10. Formularul Main. Prima pagină a aplicației

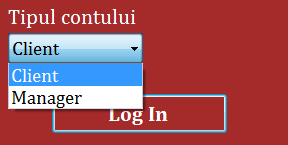
La apăsarea butonului Autentificare, se deschide formularul [**Autentificare**](#qoeewwaq412q). În acest formular utilizatorii deja înregistrați pot să se autentifice introducând numele de utilizator, parola și selectând tipul contului. Odată ce datele de autentificare au fost introduse, la accesarea butonului [Log In](#8tcfaxyrvi19) au potențial de dezvoltare mai multe situații: 

Figura 12. Rolurile disponibile la autentificare

Dacă datele introduse ale utilizatorului sunt invalide, goale, sau nu se găsesc în baza de date, va apărea un MessageBox cu mesaj de atenționare.

Dacă datele sunt corecte, utilizatorul va fi aruncat pe pagina principală, primind acces la utilizarea paginilor din meniu.

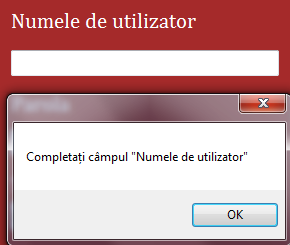
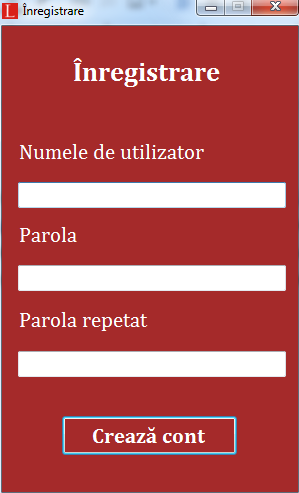


Figura 11. Formularul Autentificare

Figura 13. Exemplu de eroare la autentificare

Dacă utilizatorul nu are cont, înregistrat acesta poate face accesa butonul [Creare cont nou](#4t04tdxktj3e), astfel deschizând formularul [**Inregistrare**](#8wefg3ifzhsb). În acest formular utilizatorul trebuie să introducă un nume, o parolă și să repete parola încă o dată. Dacă numele este introdus, iar parolele coincid, la apăsarea butonului [Sign Up](#eb6h47nuezoj), acesta va fi redirecționat pe pagina principală, primind acces la utilizarea paginilor din meniu.

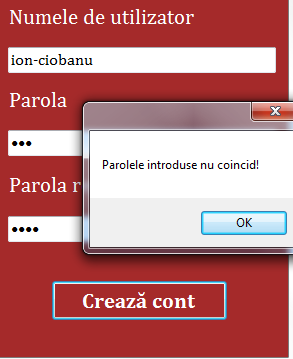
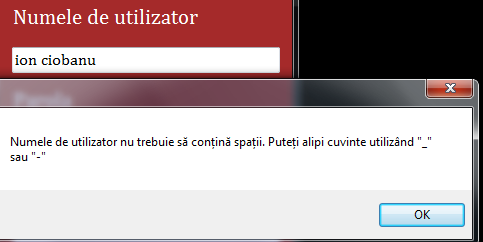


Figura 16. Eroare la inregistrare. Format de nume invalid

Figura 15. Eroare la inregistrare. Parolele nu coincid

Figura 14. Formularul Inregistrare

Așa va arăta pagina principală dupa o autentificare/înregistrare cu success:



Figura 17. Pagina principală după autentificare

Dacă, dorim să ne deconectăm de la contul curent, putem apăsa butonul [Deconectare](#4vhm7jw75cg9). Aplicația va ajunge la [starea inițială](#maxxqvro0b97).

Pentru a primi acces la pagina de Administrare este necesar să ne [autentificăm](#skg9tzcnqu4g) ca Manager.





Figura 18. Pagina Administrare accesibilă

Figura 18. Pagina Administrare inaccesibilă

## Funcționalitățile aplicației

### Pagina Catalog

Accesând butonul [Catalog](#ej6igasstbgb), primim acces în zona de conținut la 3 DataGridView-uri: Categorii de genuri de cărți, Genuri de cărți, Cărți. Făcând [click](#i86qdb8z3pv6) pe o celulă din Categorii de genuri de cărți, se vor încărca genurile de cărți. Făcând [click](#r83rqygvxjl1) pe un gen de carte se vor încărca cărțile din acest gen. Făcând [dublu click](#m7iy8pu0d8es) pe o carte din Cărți, se va deschide formularul [BookInformation](#odoorygoxkg1) cu detalii despre cartea accesată.

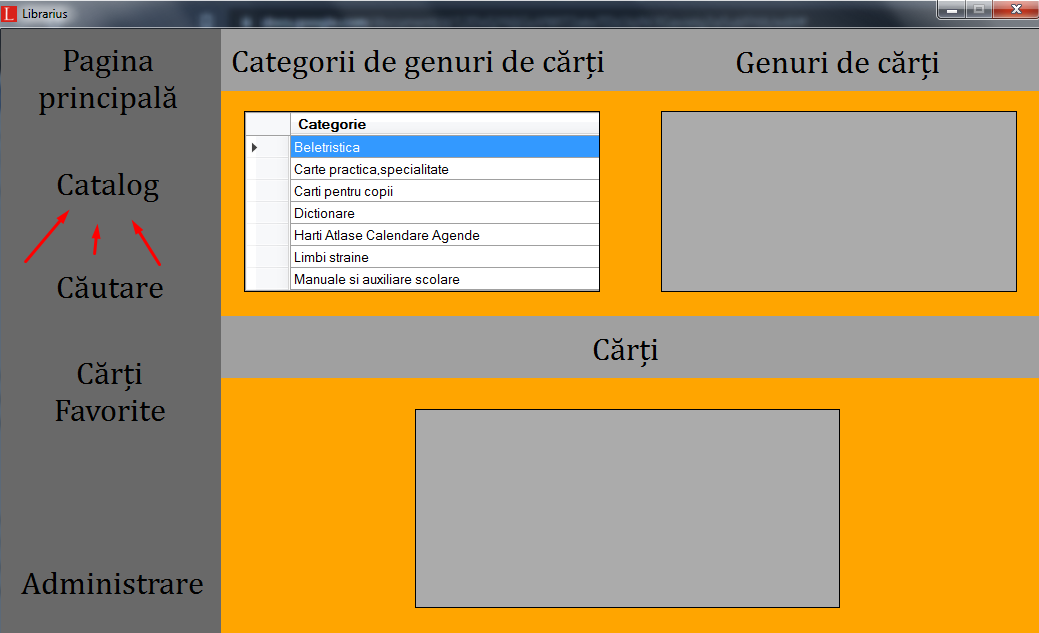


Figura 19. Pagina Catalog inițială

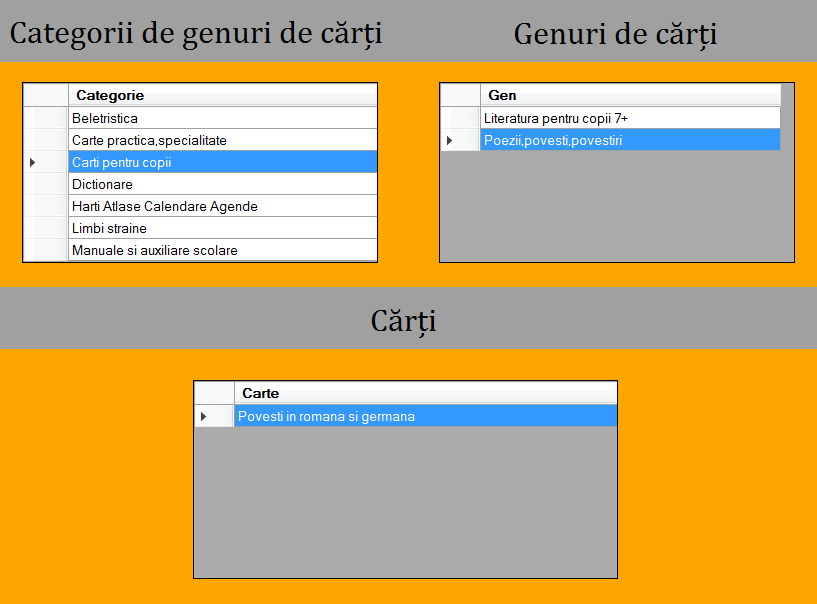


Figura 20. Pagina Catalog cu lista de cărți vizibilă

Formularul [BookInformation](#odoorygoxkg1), conține detaliile despre carte. Denumirea cărții este un link către carte de pe site-ul web [Librarius](https://librarius.md/ro/). În stânga putem vedeam imaginea cărții, în dreapta detaliile despre carte.

În colțul din dreapta jos observăm prețul cărții, iar dacă aceasta are reducere, putem vedea reducerea și prețul final.

În colțul stâng observăm butonul [Adaugă la Favorite](#h11b16elkhxw). Accesându-l cartea se va adăuga la lista de cărți favorite, iar în locul lui va apărea butonul [Șterge din Favorite](#ipq0v6n5w98p).

# 

Figura 21. Formularul BookInformation. Butonul Adaugă la Favorite

# 

# 

Figura 22. Formularul BookInformation. Butonul Șterge din Favorite

### Pagina Căutare

Accesând butonul [Căutare](#ivz13u4whkov), primim acces în zona de conținut la o [bară de căutare](#p70wivvmtjw8), la butonul [Caută](#efs5606g3t8a) și la un DataGridView. Deasemenea putem alege filtre de căutare: căutare în baza numelui autorului sau în baza denumirii cărții.

Selectând unul din filtre, introducem text în bară de căutare și apăsăm [Caută](#efs5606g3t8a) sau tasta Enter. În DataGridView vor apărea cărțile găsite.

[Butonul din stânga](#p70wivvmtjw8) barei este predestinat curățării acesteia de text.

# 

Figura 23. Pagina Căutare



Figura 24. Rezultatele căutării după Carte

# 

# 

Figura 25. Rezultatele căutării după Autor

# 

# 

### Pagina Cărți Favorite

Accesând butonul [Cărți Favorite](#2b1ommxt8ij4), primim acces în zona de conținut la lista cărților favorite. Pentru a adăuga alte cărți utilizăm paginile precedente și funcționalul formularului [BookInformation](#odoorygoxkg1). Pentru a șterge cărți din listă, putem utiliza și paginile precedente și pagina aceasta, făcând [dublu click](#1q3xqh8zn0aa) pe celulă și utilizând funcționalul formularului [BookInformation](#odoorygoxkg1).

Pentru a exporta lista în Excel, utilizăm butonul [Exportă în Excel](#ourycneb7hky). Fișierul cărți\_favorite.xlsx va fi salvat în folderul cu aplicația sau în (bin\Debug\cărți\_favorite.xlsx).

# 

Figura 26. Lista de cărți favorite. Exportarea datelor

# 

# 

# 

Figura 27. Lista de cărți favorite exportată în Excel

### Pagina Administrare

Fiind autentificat ca Manager, accesăm butonul [Administrare](#au7slopy1su5) și primim acces în zona de conținut la 2 butoane: [Adaugă Carte](#ay8sx3xcqwy5) și [Șterge Carte](#cxx0fjbj0f4t).

Butonul [Adaugă Carte](#ay8sx3xcqwy5) încarcă pagina de adăugare de noi cărți în baza de date.

Butonul [Șterge Carte](#cxx0fjbj0f4t) încarcă pagina de ștergere a cărților din baza de date.

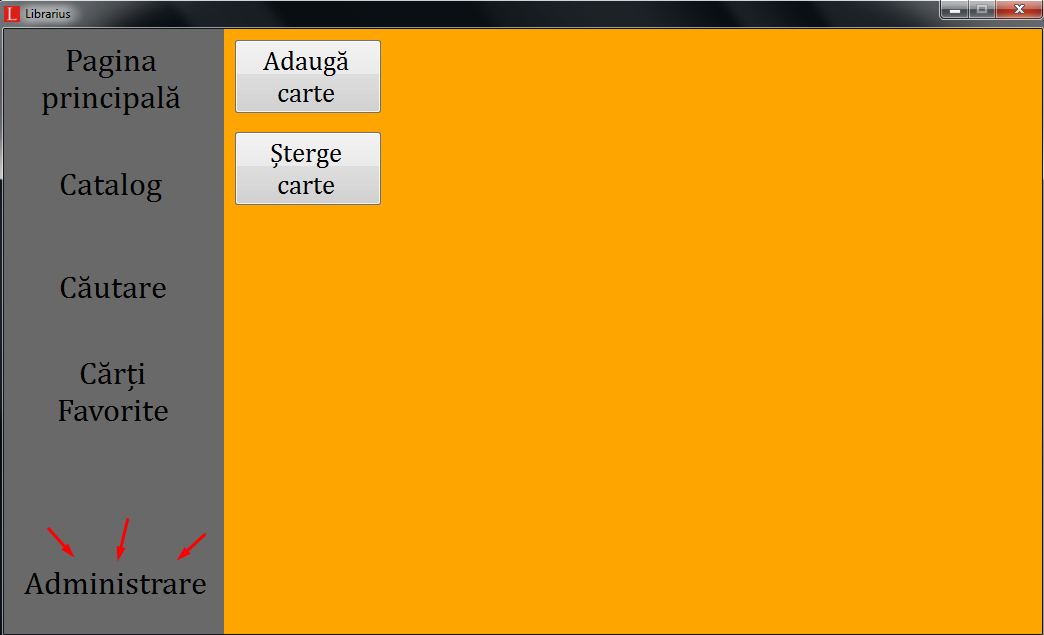


Figura 28. Pagina Administrare

# 

# 

### Pagina Adaugă Carte

Accesând Butonul [Adaugă Carte](#ay8sx3xcqwy5), primim acces în zona de conținut la câmpurile în care trebuie să introducem informațiile despre cartea pe care dorim să o adăugăm în baza de date. Odată ce completăm toate câmpurile, accesăm butonul [Adaugă](#ybbclxtsqf5h) pentru a introduce cartea în baza de date.

Nu vom putea accesa butonul dacă nu completăm toate câmpurile cu informațiile validă, despre a cărui format aflăm din diverse MessageBox-uri care apar la apăsarea butonului. La introducerea cărții în baza de date vom vedea un mesaj de succes în aplicație.



Figura 29. Pagina Adaugă Carte

# 

# 

Figura 30. Adăugare cărții în baza de date

# 

# 

# 

# 

Figura 31. Cartea a fost adăugată cu succes

### Pagina Șterge Carte

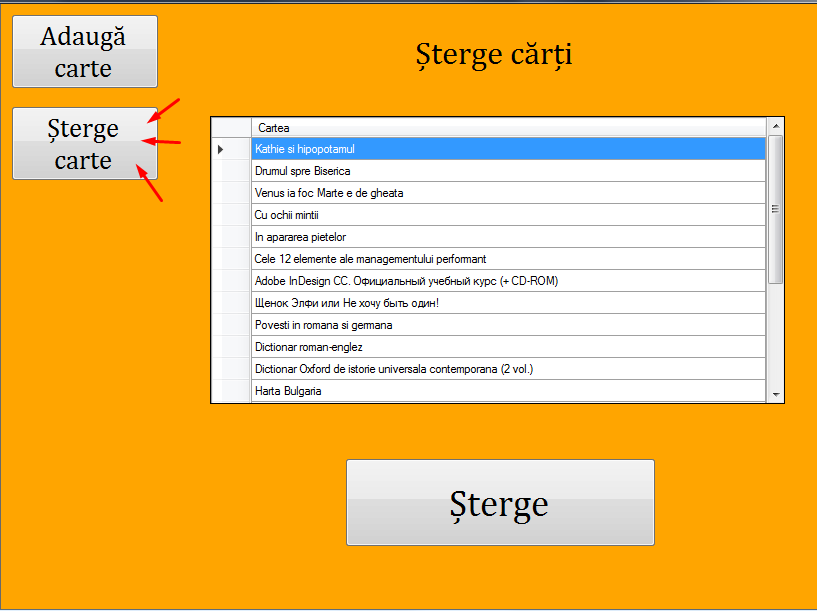
Accesând Butonul [Șterge Carte](#kix.8jkq9llpuvj2), primim acces în zona de conținut la un DataGridView ce conține toate cărțile din baza de date și la butonul [Șterge](#skg9tzcnqu4g). Pentru a șterge cărți este novoie de selectat cărțile dorite, făcând click pe celule (putem utiliza și tastele SHIFT sau CTRL + CLICK). Putem selecta atât o singură carte cât și mai multe cărți. La accesarea butonului [Șterge](#skg9tzcnqu4g), toate cărțile selectate se vor șterge din baza de date, iar lista de cărți se va reîmprospăta.

Figura 32. Pagina Șterge Carte

### 

### 

Figura 33. Pagina Șterge. Selectarea cărților spre ștergere

### 

### 

Figura 34. Pagina Șterge. Lista de cărți după ștergere.

# 

# Concluzie

Datorită practicii date am obținut experiență reală în dezvoltarea aplicațiilor desktop utilizând Windows Forms și C#. Am învățat cum să utilizez diverse controale în cel mai eficient mod, pentru a face experiența utilizatorului cât mai plăcută. Mi-am consolidat cunoștințele și în baze de date, integrând o bază de date în aplicație și realizând interogări conform unui API bine conceput.

Practica dată m-a ajutat să înțeleg cum e să lucrezi într-un colectiv de oameni zilnic. Am învățat să respect regulile impuse, și că menirea acestora este pentru eficienta dezvoltare a întregului colectiv în termenul propus. Am înțeles că este nevoie de multă concentrație și răbdare pentru a face față sarcinilor propuse.

De asemenea practica te impune să gestionezi propriul timp într-un mod care ar permite realizarea tuturor sarcinilor și al altor obligații în termenii prestabiliți, astfel învățând să gestionezi mai eficient deadline-urile.

Activitățile în cadrul acestui stagiu de practică mi-au oferit oportunități de dezvoltare a competențelor profesionale și totodată m-a făcut să înțeleg importanța structurii unei aplicații vizuale, a lizibilității codului și a diferitor principii care îți permit dezvoltarea și întreținerea unui produs de calitate înaltă care este iubit de utilizatorii săi.

# Bibliografie

1. <https://librarius.md/ro/>
2. <https://www.w3schools.com/cs/index.php>
3. <https://www.w3schools.com/sql/default.asp>
4. <https://stackoverflow.com/>

# Anexe

## Autentificarea ca Manager

Pentru a ne autentifica la etapa de autentificare ca Manager, este necesar să introducem următoarele date:

Nume de utilizator: admin

Parola: 123

Tipul contului: Manager

## Formularul Main

### Butonul “Autentificare”:

private void buttonMainPageAuthentification\_Click(object sender, EventArgs e)

{

Authentification a = new Authentification(this);

a.StartPosition = FormStartPosition.CenterScreen;

a.Show();

}

### Butonul “Deconectare”

private void buttonMainPageLogOut\_Click(object sender, EventArgs e)

{

labelNavAdmin.Enabled = false;

authentifiedUser = null;

buttonMainPageAuthentification.Visible = true;

buttonMainPageLogOut.Visible = false;

labelMainPageAuthentificationStatus.Text = "Vă rugăm să vă autentificați";

}

## Formularul Autentificare

### Butonul “Log In”

private void buttonLogIn\_Click(object sender, EventArgs e)

{

Dictionary<string, UserInfo> usersCredentials = LibraryDatabase.getSavedUsers();

string username = textBoxUserName.Text;

string password = textBoxUserPassword.Text;

if (String.IsNullOrWhiteSpace(username))

{

MessageBox.Show("Completați câmpul \"Numele de utilizator\"");

return;

}

if (String.IsNullOrWhiteSpace(password))

{

MessageBox.Show("Completați câmpul \"Parola\"");

return;

}

if (username.Split(' ').Length > 1)

{

MessageBox.Show("Numele de utilizator nu trebuie să conțină spații. Puteți alipi cuvinte utilizând \"\_\" sau \"-\"");

return;

}

if (!usersCredentials.ContainsKey(username))

{

MessageBox.Show("Numele de utilizator introdus nu există!");

return;

}

UserInfo foundUser = usersCredentials[username];

if (!password.Equals(foundUser.Password))

{

MessageBox.Show("Parola introdusă este incorectă!");

textBoxUserPassword.Clear();

return;

}

string loginType = comboBoxLoginType.SelectedItem.ToString();

if (!loginType.Equals(foundUser.Role))

{

MessageBox.Show("Acest cont nu are permisiuni de " + comboBoxLoginType.Text);

return;

}

this.mainSender.authentifyUser(foundUser);

this.Close();

}

### Butonul “Creare cont nou”

private void linkLabelSignUp\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

Registration r = new Registration(mainSender);

r.StartPosition = FormStartPosition.CenterScreen;

r.Show();

this.Close();

}

## Formularul Inregistrare

### Butonul “Sign Up:

private void buttonSignUp\_Click(object sender, EventArgs e)

{

string username = textBoxUserName.Text;

string password = textBoxUserPassword.Text;

string repeatedPassword = textBoxUserPasswordRepeated.Text;

if (string.IsNullOrWhiteSpace(username))

{

MessageBox.Show("Completați câmpul \"Numele de utilizator\"!");

return;

}

if (string.IsNullOrWhiteSpace(password))

{

MessageBox.Show("Completați câmpul \"Parola\"");

return;

}

if (string.IsNullOrWhiteSpace(repeatedPassword))

{

MessageBox.Show("Completați câmpul \"Parola repetat\"");

return;

}

if (username.Split(' ').Length > 1)

{

MessageBox.Show("Numele de utilizator nu trebuie să conțină spații. Puteți alipi cuvinte utilizând \"\_\" sau \"-\"");

return;

}

if (!password.Equals(repeatedPassword))

{

MessageBox.Show("Parolele introduse nu coincid!");

return;

}

Dictionary<string, UserInfo> users = LibraryDatabase.getSavedUsers();

if (users.ContainsKey(username))

{

MessageBox.Show("Numele de utilizator introdus este deja înregistrat. Vă rugăm să introduceți alt nume!");

return;

}

UserInfo newUser = new UserInfo(username, password, "Client");

saveUser(newUser);

mainSender.authentifyUser(newUser);

this.Close();

}

## UserControl-ul Catalog

### Evenimentul “dataGridViewCategoriiGenuri\_CellClick”

private void dataGridViewCategoriiGenuri\_CellClick(object sender, DataGridViewCellEventArgs e)

{

dataGridViewCarti.DataSource = null;

string categorieGen = dataGridViewCategoriiGenuri.SelectedCells[0].Value.ToString();

DataTable genuriTable = LibraryDatabase.getGenuriTable(categorieGen);

dataGridViewGenuri.ClearSelection();

dataGridViewGenuri.CurrentCell = null;

dataGridViewGenuri.DataSource = genuriTable;

dataGridViewGenuri.Columns[0].Width = 300;

}

### 

### Evenimentul “dataGridViewGenuri\_CellClick”

private void dataGridViewGenuri\_CellClick(object sender, DataGridViewCellEventArgs e)

{

string gen = dataGridViewGenuri.SelectedCells[0].Value.ToString();

DataTable cartiTable = LibraryDatabase.getCartiTable(gen);

dataGridViewCarti.ClearSelection();

dataGridViewCarti.CurrentCell = null;

dataGridViewCarti.DataSource = cartiTable;

dataGridViewCarti.Columns[0].Width = 450;

}

### Evenimentul “dataGridViewCarti\_CellMouseDoubleClick”

public void showBookInformation(object sender)

{

BookInformation bookInfo = new BookInformation(this, favoriteBooksPage);

bookInfo.Text = "Detalii Carte";

DataGridView dataGrid = (DataGridView)sender;

DataGridViewCell selectedCell = dataGrid.SelectedCells[0];

string bookName = (string)selectedCell.Value;

string[] list = {

"BookName", "Price", "Publisher", "PublishYear", "Pages", "StockState", "Discount", "Type", "Genre", "ISBN", "Author"

};

string[] bookAttributes = LibraryDatabase.getBookAttributes(bookName);

//string price = bookAttributes[1];

//string publisher = bookAttributes[2];

//string publishYear = bookAttributes[3];

//string pages = bookAttributes[4];

//string stockState = bookAttributes[5];

//string discount = bookAttributes[6];

//string type = bookAttributes[7];

//string genre = bookAttributes[8];

//string isbn = bookAttributes[9];

//string author = bookAttributes[10]

//string imagePath = bookAttributes[11]

//string link = bookAttributes[12]

string labelPrefix = "labelPlaceholder";

for (int i = 0; i <= 10; ++i)

{

Control[] labelControl = bookInfo.Controls.Find(labelPrefix + list[i], true);

Control ctrl = labelControl[0];

Label label = (Label)ctrl;

label.Text = bookAttributes[i];

}

string discount = bookAttributes[6];

string price = bookAttributes[1];

if (!string.IsNullOrEmpty((discount)))

{

formatDiscountControls(bookInfo, price, discount);

}

else

{

hideDiscountControls(bookInfo);

shrinkPriceSection(bookInfo);

}

Control[] pictureControl = bookInfo.Controls.Find("pictureBoxPlaceholderBookImage", true);

PictureBox picture = (PictureBox)pictureControl[0];

string imagePath = bookAttributes[11];

picture.Image = Image.FromFile("..\\..\\..\\" + imagePath);

string link = bookAttributes[12];

bookInfo.setLink(link);

bookInfo.Show();

}

private void dataGridViewCarti\_CellMouseDoubleClick(object sender, DataGridViewCellMouseEventArgs e)

{

showBookInformation(sender);

}

private void formatDiscountControls(BookInformation bookInfo, string price, string discount)

{

double priceNum = double.Parse(price);

double total = priceNum - (double.Parse(discount) \* 0.01) \* priceNum;

Control[] labelControl = bookInfo.Controls.Find("labelPlaceholderPriceWithDiscount", true);

Label label = (Label)labelControl[0];

label.Text = total.ToString();

labelControl = bookInfo.Controls.Find("labelPlaceholderDiscount", true);

label = (Label)labelControl[0];

label.Text += "%";

}

private void hideDiscountControls(BookInformation bookInfo)

{

Control[] labelControl = bookInfo.Controls.Find("labelPriceWithDiscount", true);

Label label = (Label)labelControl[0];

label.Hide();

labelControl = bookInfo.Controls.Find("labelPlaceholderPriceWithDiscount", true);

label = (Label)labelControl[0];

label.Hide();

labelControl = bookInfo.Controls.Find("labelDiscount", true);

label = (Label)labelControl[0];

label.Hide();

}

private void shrinkPriceSection(BookInformation bookInfo)

{

Control[] panelControl = bookInfo.Controls.Find("panelPriceSection", true);

Panel panel = (Panel)panelControl[0];

panel.Size = new Size(80, 82);

panel.Location = new Point(520, 416);

}

## Formularul BookInformation

### Butonul Adaugă la Favorite

private void buttonAddFavorite\_Click(object sender, EventArgs e)

{

if (catalog.addFavoriteBook(labelPlaceholderBookName.Text))

{

buttonAddFavorite.Hide();

buttonDeleteFavorite.Show();

}

}

public bool addFavoriteBook(string bookName)

{

return favouriteBooks.Add(bookName);

}

### Butonul Șterge din Favorite

private void buttonDeleteFavorite\_Click(object sender, EventArgs e)

{

if (catalog.removeFavoriteBook(labelPlaceholderBookName.Text))

{

buttonAddFavorite.Show();

buttonDeleteFavorite.Hide();

favoriteBooksPage.fillFavoriteBooksTable();

} else

{

throw new NotSupportedException("Unexpected behaviour.");

}

}

public bool removeFavoriteBook(string bookName)

{

return favouriteBooks.Remove(bookName);

}

## UserControl-ul Căutare

### Bara de căutare

private void textBoxSearch\_KeyPress(object sender, KeyPressEventArgs e)

{

if (e.KeyChar.Equals((char)13))

{

executeSearchQuery();

}

}

### Butonul “Caută”

### private void executeSearchQuery()

### {

### if (string.IsNullOrWhiteSpace(textBoxSearch.Text))

### {

### MessageBox.Show("Completați bara de căutare!");

### return;

### }

### 

### string request = textBoxSearch.Text;

### DataTable result = null;

### if (radioButtonFilterName.Checked)

### {

### result = LibraryDatabase.searchBooksByName(request);

### }

### if (radioButtonFilterAuthor.Checked)

### {

### result = LibraryDatabase.searchBooksByAuthor(request);

### }

### 

### dataGridViewResult.DataSource = result;

### dataGridViewResult.Columns[0].Width = 322;

### dataGridViewResult.Columns[1].Width = 202;

### }

### Butonul “ClearSearch”

private void buttonClearSearch\_Click(object sender, EventArgs e)

{

textBoxSearch.Clear();

}

## UserControl-ul Cărți Favorite

### Evenimentul [dataGridViewResult\_CellMouseDoubleClick](#_ku2576824mnm)

### Butonul “Exportă în Excel”

private WorkSheet initializeWorkBook()

{

WorkBook workBook = WorkBook.Create(ExcelFileFormat.XLSX);

var sheet = workBook.CreateWorkSheet("Cărți favorite");

string librarius = "D2:E2";

sheet[librarius].Value = "Librarius";

sheet[librarius].Style.Font.Bold = true;

sheet[librarius].Style.Font.Height = 20;

sheet.Merge(librarius);

sheet["B5"].Value = "Carte";

sheet["C5"].Value = "Autor";

sheet["D5"].Value = "Editura";

sheet["E5"].Value = "Anul";

sheet["F5"].Value = "Pagini";

sheet["G5"].Value = "Gen";

sheet["H5"].Value = "ISBN";

sheet["I5"].Value = "Preț";

string tableNameRange = "B4:I4";

var tableName = sheet[tableNameRange];

tableName.Value = "Cărți favorite";

tableName.Style.HorizontalAlignment = IronXL.Styles.HorizontalAlignment.Center;

tableName.Style.Font.Bold = true;

wrapCellWithBorders(sheet, tableNameRange, BorderType.Thin);

sheet.Merge(tableNameRange);

var header = sheet["B5:I5"];

header.Style.Font.Bold = true;

wrapCellWithBorders(sheet, "B5:I5", BorderType.Thin);

header.Style.ShrinkToFit = true;

return sheet;

}

private void writeDateToWorkSheet(WorkSheet sheet)

{

int startDataRow = 6;

int rowCounter = fetchFavoriteBooks().Length;

int offset = 2;

int dateRowNumber = rowCounter + startDataRow + offset;

char dateRowLetter = 'B';

string dateCellLocation = dateRowLetter.ToString() + dateRowNumber.ToString();

string date = DateTime.UtcNow.ToString("MM/dd/yyyy H:mm");

string dateCellContent = $"Data exportării: {date}";

sheet[dateCellLocation].Value = dateCellContent;

}

private string[] fetchFavoriteBooks()

{

var rows = dataGridViewFavoriteBooks.Rows;

string[] favoriteBooks = new string[rows.Count];

string bookName = "";

for (int i = 0; i < rows.Count; ++i)

{

bookName = rows[i].Cells[0].Value.ToString();

favoriteBooks[i] = bookName;

}

return favoriteBooks;

}

private string[] prepareAttributesForExport(string[] attributes)

{

string[] exportAttributes = new string[8];

int[] requiredAttributesIndex = new int[] { 0, 10, 2, 3, 4, 8, 9, 1 };

for (int i = 0; i < exportAttributes.Length; ++i)

{

exportAttributes[i] = attributes[requiredAttributesIndex[i]];

}

return exportAttributes;

}

private void wrapCellWithBorders(WorkSheet sheet, string cell, BorderType borderType)

{

var c = sheet[cell];

c.Style.LeftBorder.Type = borderType;

c.Style.TopBorder.Type = borderType;

c.Style.RightBorder.Type = borderType;

c.Style.BottomBorder.Type = borderType;

}

private void addRowToExportSheet(WorkSheet sheet, string[] exportAttributes, int rowIndex)

{

if (rowIndex < 0)

{

MessageBox.Show("Eroare: Index Negativ.");

return;

}

int firstRowNumber = 6;

int rowNumber = firstRowNumber + rowIndex;

string[] columnLetters = new string[] {"B", "C", "D", "E", "F", "G", "H", "I"};

for (int i = 0; i < columnLetters.Length; ++i)

{

string cell = columnLetters[i] + rowNumber;

sheet[cell].Value = exportAttributes[i];

wrapCellWithBorders(sheet, cell, BorderType.Thin);

// styling

var c = sheet[cell];

c.Style.VerticalAlignment = VerticalAlignment.Center;

}

string bookName = exportAttributes[0];

if (bookName.Length > 30)

{

int rows = sheet.Rows.Length;

sheet.Rows[rows - 1].Height = 1300;

sheet["B" + rowNumber].Style.WrapText = true;

}

}

private void buttonExport\_Click(object sender, EventArgs e)

{

var sheet = initializeWorkBook();

var favoriteBooks = fetchFavoriteBooks();

int rowCounter = 0;

foreach (var book in favoriteBooks)

{

string[] attributes = LibraryDatabase.getBookAttributes(book);

string[] exportAttributes = prepareAttributesForExport(attributes);

addRowToExportSheet(sheet, exportAttributes, rowCounter);

++rowCounter;

}

writeDateToWorkSheet(sheet);

try

{

sheet.SaveAs("carti\_favorite.xlsx");

MessageBox.Show("Datele au fost exportate!");

} catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

## 

## UserControl-ul “Administrare”

### Butonul “Adaugă Carte”

private void buttonNavAddBook\_Click(object sender, EventArgs e)

{

panelContent.Controls.Clear();

panelContent.Controls.Add(addBookPage);

}

### 

### Butonul “Șterge Carte”

private void buttonNavRemoveBook\_Click(object sender, EventArgs e)

{

panelContent.Controls.Clear();

panelContent.Controls.Add(removeBookPage);

}

## UserControl-ul “Adaugă carte”

### Butonul “Adaugă”

private void buttonAdd\_Click(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(tBoxName.Text))

{

MessageBox.Show("Completați câmpul Denumire!");

return;

}

if (!checkPrice(tBoxPrice.Text))

{

tBoxPrice.Clear();

return;

}

if (string.IsNullOrWhiteSpace(tBoxPublisher.Text))

{

MessageBox.Show("Completați câmpul Editura!");

return;

}

if (!checkPublishingYear(tBoxPublishingYear.Text))

{

tBoxPublishingYear.Clear();

return;

}

if (!checkPages(tBoxPages.Text))

{

tBoxPages.Clear();

return;

}

if (checkBoxNoDiscount.CheckState.Equals(CheckState.Unchecked))

{

if (!checkDiscount(tBoxDiscount.Text))

{

tBoxDiscount.Clear();

return;

}

}

if (!checkISBN(tBoxISBN.Text))

{

tBoxISBN.Clear();

return;

}

if (!checkAuthor(tBoxAuthor.Text))

{

tBoxAuthor.Clear();

return;

}

bool isPublisherAdded = false;

var publishers = LibraryDatabase.getBookPublishers();

if (publishers.Contains(tBoxPublisher.Text))

{

isPublisherAdded = true;

}

bool isAuthorAdded = false;

var authors = LibraryDatabase.getBookAuthors();

if (authors.Contains(tBoxAuthor.Text))

{

isAuthorAdded = true;

}

string id = findAvailableBookId().ToString();

string name = tBoxName.Text;

string price = tBoxPrice.Text;

string publisherId = (isPublisherAdded)

? LibraryDatabase.getPublisherId(tBoxPublisher.Text).ToString()

: LibraryDatabase.addPublisher(tBoxPublisher.Text).ToString();

string publishingYear = tBoxPublishingYear.Text;

string pages = tBoxPages.Text;

string stockStateId = LibraryDatabase.getStockStateId(cBoxStockState.Text).ToString();

string discount = tBoxDiscount.Text;

if (checkBoxNoDiscount.CheckState.Equals(CheckState.Checked))

{

discount = "0";

}

string typeId = LibraryDatabase.getBookTypeId(cBoxType.Text).ToString();

string genreId = LibraryDatabase.getGenreId(cBoxGenre.Text).ToString();

string isbn = tBoxISBN.Text;

string[] author = tBoxAuthor.Text.Split(' ');

string authorLastName = author[0];

string authorFirstName = author[1];

string authorId = (isAuthorAdded)

? LibraryDatabase.getAuthorId(authorLastName, authorFirstName).ToString()

: LibraryDatabase.addAuthor(authorLastName, authorFirstName).ToString();

LibraryDatabase.addBook(id, name, price, publisherId, publishingYear, pages, stockStateId, discount, typeId, genreId, isbn, authorId);

MessageBox.Show($"Cartea '{name}' a fost adăugată cu success în baza de date!");

clearInputControls();

}

## UserControl-ul “Adaugă carte”

### Butonul “Șterge”

private void buttonRemoveBook\_Click(object sender, EventArgs e)

{

var selectedCells = dataGridViewBooks.SelectedCells;

List<string> booksForRemoval = new List<string>();

foreach (DataGridViewCell cell in selectedCells)

{

string book = cell.Value.ToString();

booksForRemoval.Add(book);

}

foreach (string book in booksForRemoval)

{

LibraryDatabase.removeBook(book);

}

updateDataGridViewBooks();

}