



**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**  
Lect. Univ. Dr. Marius-Iulian 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 to optimizing operations and enhancing the user experience.

# Cuprins

## Cuprins

<b>1. Introducere .....</b>	<b>5</b>
1.1 Obiectivul general al lucrării .....	5
1.2 Obiectivele specifice ale lucrării .....	5
1.3 Contextul lucrării .....	6
1.4 Istoria aplicațiilor desktop .....	6
1.5 Provocări în dezvoltarea aplicațiilor desktop.....	7
1.6 De ce am ales această temă? .....	7
1.7 Tehnologii utilizate .....	8
1.8 Flux de lucru .....	8
<b>2. Servicii .....</b>	<b>9</b>
2.1. Serviciul identity .....	9
2.2. Serviciul biblioteca-backend .....	9
2.3. Arhitectura Serviciului de Frontend.....	10
<b>3. Design .....</b>	<b>11</b>
3.1. Design-ul formulelor .....	11
3.2. Design-ul elementelor .....	11
<b>4. Funcționalități.....</b>	<b>15</b>
4.1 Pagina de logare .....	15
4.2 Pagina principala Utilizator .....	16
4.3 Pagina principala Admin .....	18
4.4 Funcționalitatea căutare .....	19
4.5 Pagina Profil.....	21
4.5.1 Funcționalități .....	21
4.5.2 Pop-up Schimbă Numele.....	22
4.6 Pagina cereri utilizatori .....	22
4.7 Pagina toate recenziile.....	23
4.8 Pagina adaugă carte .....	24
4.8.1 Modul adăugare.....	24
4.8.2 Modul editare .....	26

# Cuprins

4.8.3 Adaugă Autor .....	27
4.8.4 Adaugă Autor .....	28
4.8.4 Funcție Adaugă Autor .....	29
4.9 Vizualizare carte .....	30
4.9.1 Vizualizare.....	30
4.9.2 Afișare recenzii.....	31
4.9.3 Împrumută carte .....	32
4.9.4 Lasă Recenzie.....	33
4.10 Prezentare Generală.....	34
4.11 Schemă bază de date.....	35
<b>5. Controlul versiunii .....</b>	<b>36</b>
<b>Concluzie .....</b>	<b>37</b>
<b>Bibliografie .....</b>	<b>38</b>

# Capitolul 1

## 1. Introducere

### 1.1 Obiectivul general al lucrării

Obiectivul general al lucrării este de a dezvolta o aplicație desktop modernă pentru gestionarea eficientă a bibliotecilor. Această aplicație are scopul de a îmbunătăți experiența utilizatorilor reducând timpul petrecut în bibliotecă și facilitând accesul la resursele disponibile.

### 1.2 Obiectivele specifice ale lucrării

Obiectivele specifice ale lucrării sunt:

- Crearea unui sistem sigur de autentificare și autorizare pentru utilizatori și administratori.
- Dezvoltarea unei interfețe intuitive pentru gestionarea cărților, împrumuturilor, vizionarea cărților, gestionarea recenziilor și vizualizarea acestora.
- Dezvoltarea unui sistem de căutare și filtrare al cărților
- Asigurarea integrității și securității datelor stocate în baza de date.

## **1.3 Contextul lucrării**

Proiectul se încadrează în contextul modernizării sistemelor de gestionare a bibliotecilor prin utilizarea tehnologiilor de ultimă generație, acest lucru contribuind la o experiență din ce în ce mai fluidă a procesului de închirieri și o experiență net superioară a utilizatorilor.

## **1.4 Istoria aplicațiilor desktop**

O aplicație desktop, de obicei, necesită instalare pentru a rula, permițând astfel în anumite condiții utilizatorilor să interacționeze cu acestea fără a fi conectați la internet. [\[1\]](#)

### **1.4.1 Primele aplicații software (Anii 1960)**

Primele programe și aplicații software dezvoltate în anii 1960 includ limbajele de programare de bază și primele sisteme de operare. Acestea au pus bazele dezvoltării ulterioare a aplicațiilor software. [\[1\]](#)

### **1.4.2 Microcalculatoare și primele aplicații comerciale (Anii 1970)**

Apariția microcalculatoarelor în anii 1970 a dus la dezvoltarea primelor aplicații comerciale și a software-ului vândut separat de hardware. Acest lucru a permis utilizatorilor să adauge diverse programe pe calculatoarele lor personale. [\[1\]](#)

### **1.4.3 Era PC-ului și apariția GUI (Anii 1980)**

Introducerea PC-urilor și a interfețelor grafice cu utilizatorul (GUI) în anii 1980 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. [\[1\]](#)

### **1.4.4 Revoluția internetului și distribuția software-ului (Anii 1990)**

Apariția CD-ROM-urilor și a internetului în anii 1990 a revoluționat distribuția software-ului, facilitând descărcarea și actualizarea programelor. Web browser-ele au făcut internetul accesibil maselor, schimbând modul în care utilizatorii interacționează cu tehnologia. [\[1\]](#)

### **1.4.5 Aplicații integrate și mobilitatea software-ului (Anii 2000+)**

În anii 2000, aplicațiile integrate și mobilitatea software-ului au devenit tot mai importante. Aplicațiile pentru dispozitive mobile și integrarea tehnologiilor web în aplicațiile desktop au continuat să evolueze, menținând relevanța acestora. [\[1\]](#)

## **1.5 Provocări în dezvoltarea aplicațiilor desktop**

Dezvoltarea proiectului a întâmpinat diverse provocări, cum ar fi:

- Interconectarea serviciilor între ele
- Portabilitatea între dispozitive
- Migrarea bazei de date
- Realizarea unui design intuitiv și funcțional

## **1.6 De ce am ales această temă?**

Am ales această temă deoarece oferă o mare flexibilitate în realizarea proiectului și abordează metode inovatoare pentru îmbunătățirea experienței utilizatorilor de bibliotecă. Alegerea acestei teme se bazează pe dorința de a explora soluții moderne și eficiente pentru gestionarea bibliotecilor.

## **1.7 Tehnologii utilizate**

### **1.7.1 Windows Forms Application**

Utilizând C#, această bibliotecă GUI din .NET Framework este folosită pentru a crea aplicația desktop.

### **1.7.2 Entity Framework Core**

Utilizat pentru maparea entităților, facilitând lucrul cu baza de date.

### **1.7.3 ASP.NET Core**

Un framework open-source și cross-platform pentru dezvoltarea aplicațiilor web moderne.

### **1.7.4 Identity Framework**

Utilizat pentru crearea serviciului de autentificare și gestionare a utilizatorilor, inclusiv generarea token-urilor de autentificare.

### **1.7.5 SQL Server**

Utilizat pentru gestionarea bazei de date a proiectului prin interfața din Visual Studio.

### **1.7.6 Tehnologii Suport**

NuGet: Manager de pachete pentru platforma .NET.

Git: Sistem de control al versiunii

Visual Studio: Mediu de dezvoltare integrat de la Microsoft.

## **1.8 Flux de lucru**

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



# Capitolul 2

## 2. Servicii

### 2.1. Serviciul identity

Serviciul de identity din cadrul proiectului meu, are ca scop gestionarea autentificării și autorizării utilizatorilor. Acesta a fost creat cu utilizând framework-ul ASP.NET Core Identity. [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.

### 2.2. 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 trimite răspunsurile corespunzătoare. Mai jos avem exemplificat unul dintre end-pointuri.

```

[HttpGet]
public ActionResult Get(long id)
{
    var result = _dbContext.Books
        .Include(x => x.Author)
        .Include(x => x.Language)
        .FirstOrDefault(x => x.Id == id);

    if (result == null)
        return NotFound($"Book with Id = {id} not found");

    return Ok(result);
}

```

- 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

## 2.3. 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. [\[3\]](#)

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.

# Capitolul 3

## 3. Design

### 3.1. Design-ul formularelor

Pentru designul formularelor am utilizat numai componente puse la dispoziție de Visual Studio, modificând proprietățile 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]. (Fig 3.1)

Font-ul utilizate în proiect este PoorRichard. [5].

### 3.2. 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

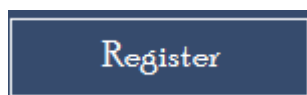


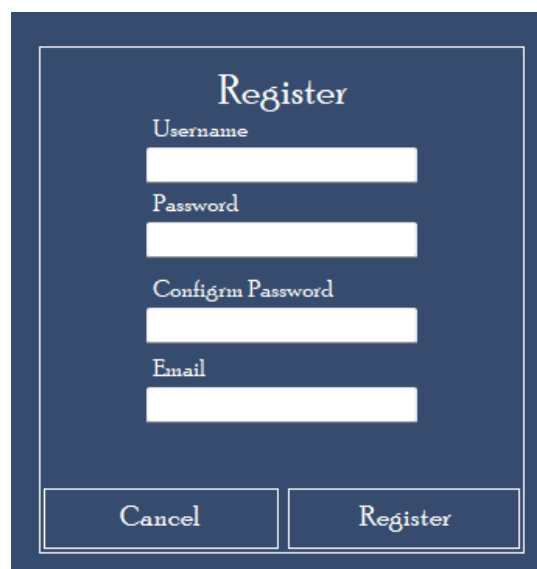
Fig. 3.1 Buton

## 2. Panel

Panelul are o interfață ușor de vizionat păstrând aceeași textură ca form-ul. (Fig 3.2)

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.



Register

Username

Password

Confirm Password

Email

Cancel Register

Fig. 3.2 Panel

### 3. DataGridView

DataGridView-ul pastrează același model, preia culoarea formularului, la fel și pentru rândurile inserate. (Fig 3.3)

Proprietăți:

- BackColor: 54,75,109
- AlternatingRowsDefaultCellStyle
  - BackColor: 54,75,109
  - ForeColor: White
- DefaultCellStyle
  - BackColor: 54,75,109
  - ForeColor: White
- Dock: Fill (în panelul parinte)

Status carti:				
Titlu	Autor	Confirmat	Data Inchiriere	DataReturnare
Povestind Copiilor	Creanga Ion	False	2024-06-01	
capra cu 3 capre	Cosbuc George	False	2024-06-01	

Fig. 3.3 DataGridView

## 4. Vizualizare Cărți 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)



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)



Fig. 3.5. Căutare cărți

# Capitolul 4

## 4. 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. (Fig 4.1.1) La prima logare utilizatorul va fi inițializat în baza de baze, primind și un mesaj de confirmare. Dacă utilizatorul este admin, acesta va primi acces la funcții speciale în aplicație. (Fig 4.1.2)

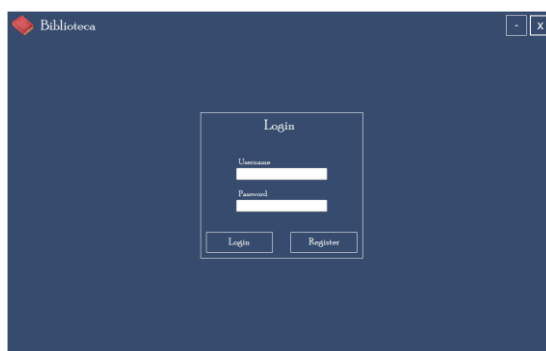


Fig. 4.1.1 Logare



Fig. 4.1.2 Inregistrare

## 4.2 Pagina principală 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 vizibilă și numărul total de cărți ce au fost aduse din baza de date.

(Fig 4.1.2.1)

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 după bunul plac.
- Actualizare în timp real: O dată la câteva secunde, aplicația trimite requesturi către pe backend pentru a putea vedea toate modificările aduse cărților.
- Buton “minimizare”: fereastra 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.



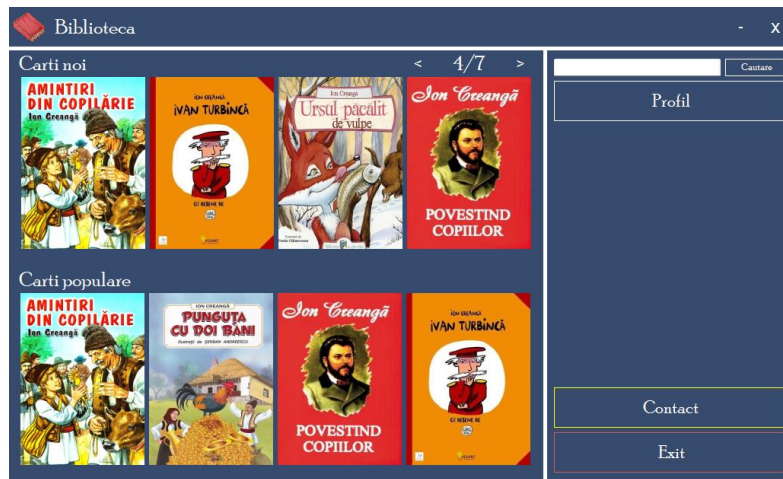


Fig. 4.1.2.1 Pagina Principala Utilizator

Mai jos 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.

```
public async void PopularBooks()
{
    try
    {
        _popularBooks = await _bookService.GetPopularBooks();
        popularBooksNumber = _popularBooks.Count();
        if (popularBooksNumber > 4)
        {
            popularBooksCurrentNumber = 4;
            changePopularBooks();
        }
    }
    catch
    {
    }
```

```
MessageBox.Show("Eroare");
```

## 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 funcțiile de bază ale utilizatorilor. (Fig 4.1.3)

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 Funcționalitatea căutare

Funcția de căutare îi oferă utilizatorului posibilitatea să caute cărțile după 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 apelată partea de backend pentru aducerea datelor. (Fig. 4.1.4.1) Lista în care sunt afișate sugestii legate de posibilele căutări este realizată pe același principiu, aceasta dispune de următoarele proprietăți:

- Se deschide după ce scriem ceva în componenta textBox.
- Dacă 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 „Exemplu cod 1”, 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 „Exemplu cod 2”, avem codul ce îi permite programului să transmită textul selectat din listBox către textBoxul în care este inițializată funcționalitatea de căutare.

#### Exemplu cod 1

```
private void TextBox1_TextChanged(object sender, EventArgs e)
{
    string searchText = textBox1.Text.ToLower();
    var filteredBooks = _books.Where(book => book.Title.ToLower().Contains(searchText)).ToList();

    listBox1.Items.Clear();
    foreach (var book in filteredBooks)
    {
        listBox1.Items.Add(book.Title);
    }
    listBox1.Visible = filteredBooks.Count > 0;
}
```

#### Exemplu cod 2

```
private void listBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    if (listBox1.SelectedItem != null)
```

```

{
    string selectedTitle = listBox1.SelectedItem.ToString();
    textBox1.Text = selectedTitle;
}
}

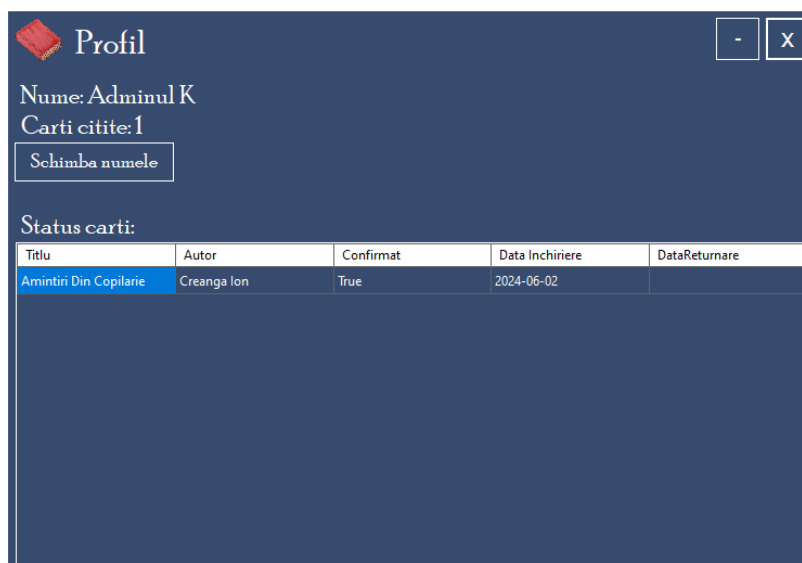
```

## 4.5 Pagina Profil

### 4.5.1 Funcționalități

În pagina de Profil, utilizatorul poate vizualiza cererile de închiriere și statusul lor într-un dataGridView. (Fig 4.1.5) 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



The screenshot shows a window titled 'Profil' with a dark blue background. At the top left is a small icon of a book. The window contains the following elements:

- Header:** 'Profil' title and window control buttons (minimize, maximize, close).
- User Information:** 'Nume: Adminul K' and 'Carti citite: 1'.
- Action:** A button labeled 'Schimba numele'.
- Table:** A table titled 'Status carti:' with the following data:
 

Titlu	Autor	Confirmat	Data Inchiriere	DataReturnare
Amintiri Din Copilarie	Creanga Ion	True	2024-06-02	

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. (Fig 4.1.5.2)

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

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

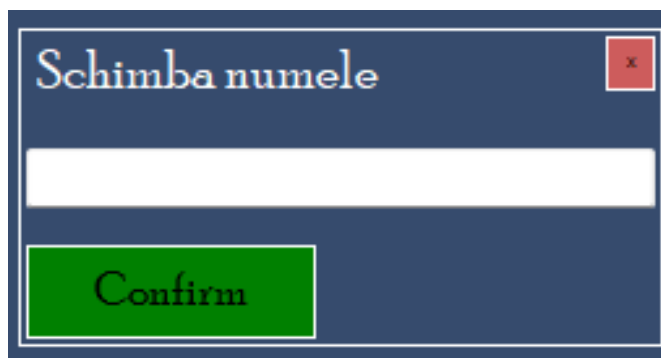


Fig. 4.1.5.2 Pop-up Schimba Numele

## 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. (Fig 4.1.6)

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

Alte funcții 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. (Fig 4.1.7)

Pagina prezintă mai multe funcționalități:

- Vizualizare recenzie: prin intermediul unui richTextBox ce conține recenzia propriu zisă.
- Vizualizare nume
- Vizualizare notă: numărul de stelute este egal cu nota lăsată de către utilizator în aplicații, unde nota maximă este 10 și cea minimă 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.

(Fig 4.1.8)

Formularul beneficiază de următoarele funcții:

- 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.
- DatePicker 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 fi actualizat în timp real.
- RichTextBox pentru descriere cu validatoare pentru lungime.



Fig. 4.1.8 Adaugă Carte

- 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.(Fig 4.1.9)

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 funcție prezintă:

- 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 Completa

### 4.8.3 Adaugă Autor

Formularul “Adaugă Autor” se află în componența “Adaugă carte”. Prin intermediul acestuia, adminul poate adăuga un nou Autor în baza de date.(Fig. 4.1.10)

- TextBox pentru nume cu validările relevante
- TextBox pentru prenume cu validările relevante
- DatePicker 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”

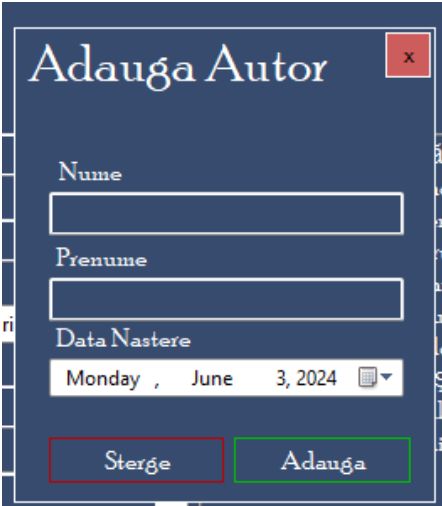


Fig. 4.1.10 Adaugă Autor

#### 4.8.4 Adaugă Autor

Formularul “Adaugă Limbă” se află în componența “Adaugă carte”. Prin intermediul acestuia, adminul poate adăuga o nouă Limbă în baza de date. (Fig, 4.1.11)

Formularul 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”

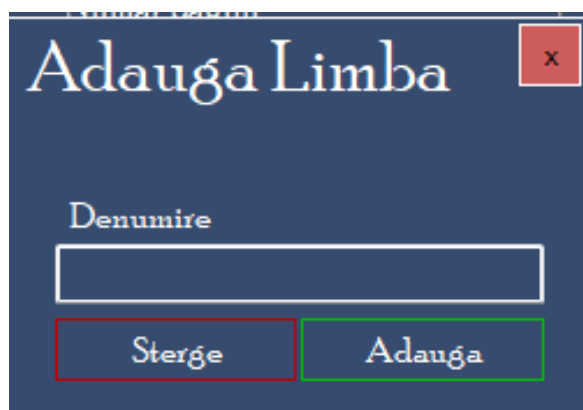


Fig. 4.1.11 Adaugă Limba

#### 4.8.4 Funcție Adaugă Autor

Mai jos este exemplificat modul în care 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

```
var sourceFile = openFileDialog1.FileName;
System.IO.File.Copy(sourceFile, _selectedImagePath, true);

await _bookService.CreateBook(new CreateBookDTO
{
    Title = textBox1.Text,
    PageNumber = int.Parse(textBox2.Text),
    Available = true,
    AppearDate = dateTimePicker1.Value,
    Description = richTextBox1.Text,
    Publisher = textBox3.Text,
    Photo = _selectedImagePath != null ? System.IO.Path.GetFileName(_selectedImagePath) :
    "",
    AuthorId = AuthorId,
    LanguageId = LanguageId,
});

comboBoxBooks();
```

## 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 după id-ul transmis din formularul precedent. Acesta dispune de:

Afișare cărți

- PictureBox pentru imagine
- Pentru afișarea obiectelor au fost afișate label-uri simple cu cărora le-a fost setată culoarea textului la alb. (Fig. 4.1.12)
- 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. (Fig 4.1.13)

- Funcționalitate ce specifică momentul 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. (Fig. 4.1.14) Requesturile utilizatorilor pot fi văzute și 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 – Înciriere



#### 4.9.4 Lasă Recenzie

Formularul “Lasă recenzie” îi permite utilizatorului să ofere un feedback privind cartea vizualizată. (Fig. 4.1.15) 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”

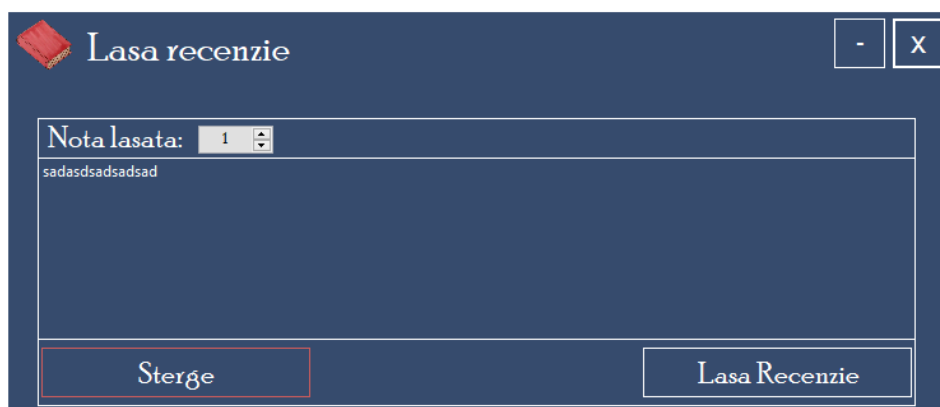


Fig. 4.1.15 Vizualizare Carte – Închiriere

## 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 situațiile în care aplicația s-ar putea închide automat într-un mod neașteptat.

(Fig. 4.1.16)

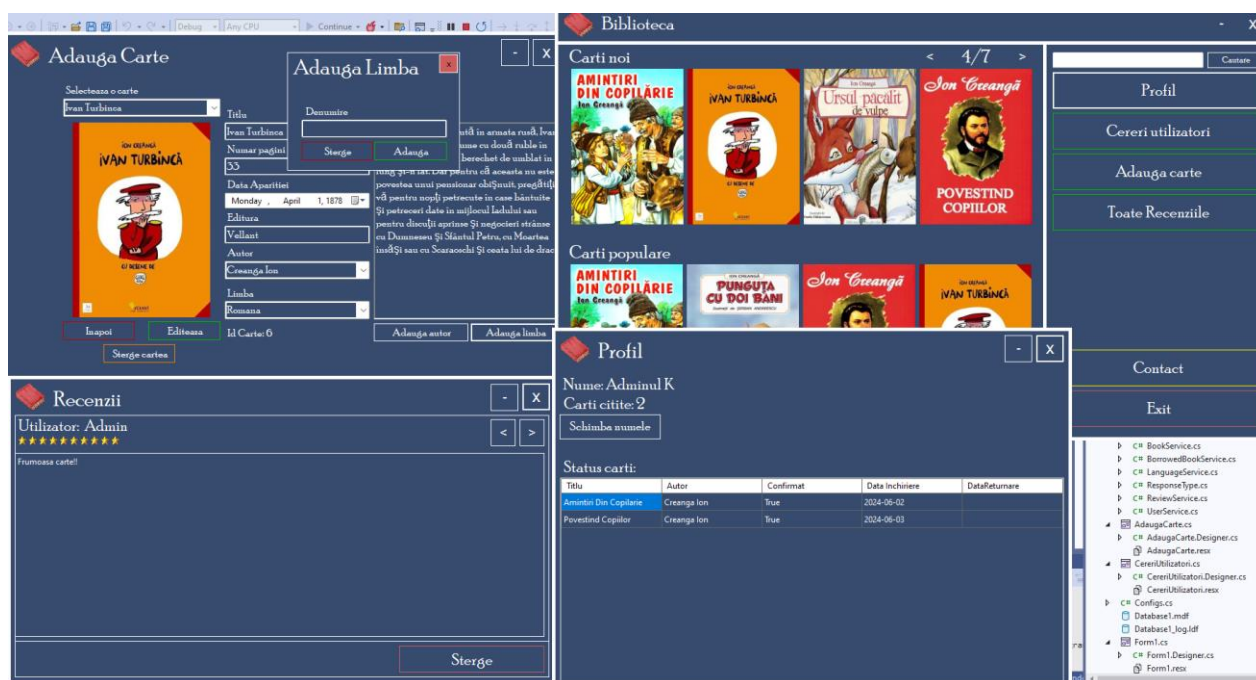


Fig. 4.1.16 Prezentare Generală

## 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)

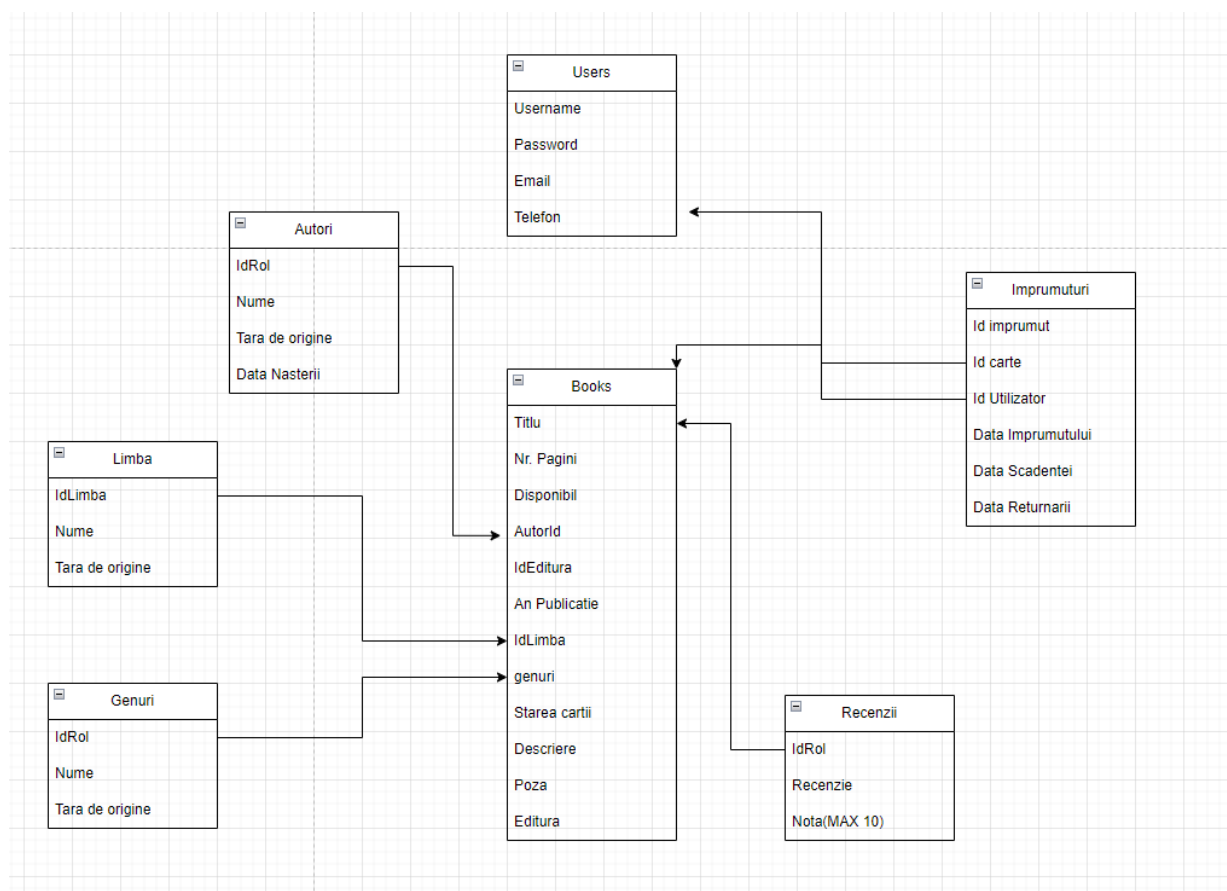


Fig. 4.1.15 Prezentare Generală

# 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)

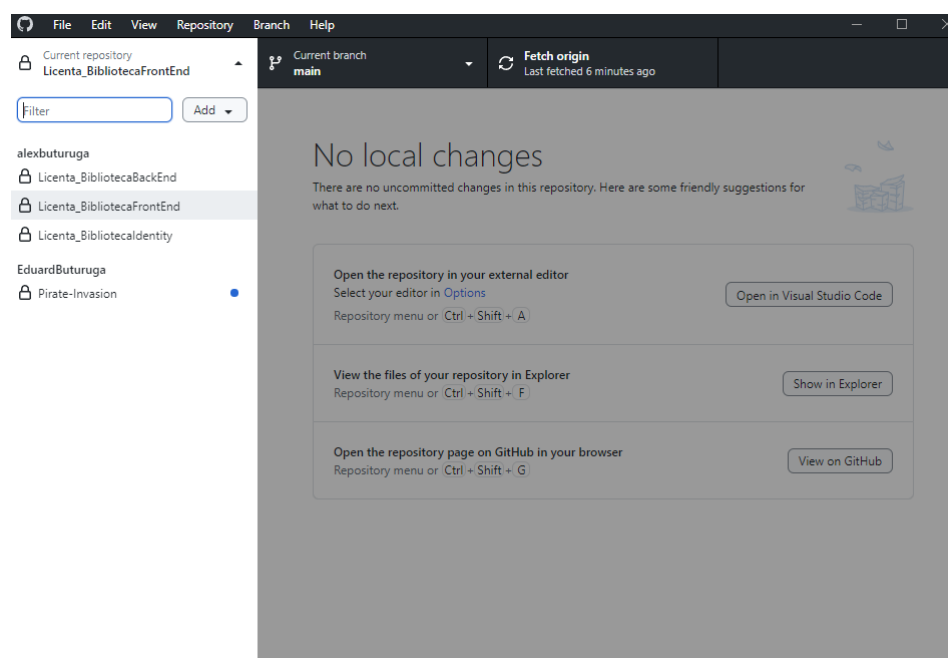


Fig. 4.1.16 Controlul versiunii

# 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 acumulat în facultate, aducând și lucruri în plus.

# 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://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 documentation: <https://dotnet.microsoft.com/en-us/apps/aspnet>
- [6] Entity Framework Core documentation: <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>
- [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/>