UNIVERSITATEA TEHNICĂ "GHEORGHE ASACHI" IAŞI
FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
SPECIALIZAREA CALCULATOARE ŞI TEHNOLOGIA INFORMAŢIEI

DISCIPLINA BAZE DE DATE PROIECT

Gestiunea unei biblioteci

Coordonator,
Cătălin Mironeanu

Student,
Donici Leonardo-Mario
Grupa 1306A

Iaşi,2023

Titlu proiect: Gestiunea unei biblioteci.

Analiza, proiectarea si implementarea unei baze de date care să modeleze activitatea unei biblioteci.

Descrierea cerințelor și modul de organizare al proiectului

Pentru o bună activitate și organizare a unei biblioteci, va trebui să implementăm o aplicație care să înregistreze și să afișeze informații despre clienții care au imprumutat carti, și bineînțeles momentul returnarii lor. Fiecare carte este înregistrată, fiecare carte aparține unei categorii înregistrate în baza de date, iar fiecare client detine un card unic de membru.

Descrierea functionala a aplicatiei

Principalele functii care se pot intalni intr-o biblioteca sunt:

- ✓ Inregistrarea si evidenta membrilor
- ✓ Evidenta imprumuturilor efectuate
- ✓ Gestionarea cartilor din biblioteca

Informații despre entități

Autori:

Tabela Autori stochează informații despre autorii cărților din bibliotecă. Atributul **id_autor** reprezintă cheia unică pentru fiecare autor. Datele biografice și legăturile cu tabelele Carti și Categorii permit o gestionare eficientă a informațiilor referitoare la autori, facilitând căutarea și organizarea cărților în funcție de autor.

Carduri_de_biblioteca:

Tabela Carduri_de_biblioteca este responsabilă de gestionarea cardurilor de membru ai bibliotecii. Fiecare membru are un card unic identificat prin **id_card**. Atributul data_expirarii indică momentul până la care cardul este valabil, iar legătura cu tabela Membri asigură asocierea cardului cu informațiile membrului corespunzător.

Carti:

Tabela Carti conține informații despre fiecare carte din bibliotecă. Fiecare carte este identificată unic prin **id_carte**, iar detalii precum titlu, autor, editură, an de publicare, număr de exemplare disponibile și limbă sunt utile pentru gestionarea eficientă a colecției de cărți.

Categorii:

Tabela Categorii definește categoriile în care sunt încadrate cărțile. Aceasta asigură o organizare tematică a cărților, facilitând căutarea și prezentarea acestora pentru membrii bibliotecii. **id_categorie** servește drept cheie unică pentru fiecare categorie.

Imprumuturi:

Tabela Imprumuturi stochează informații despre fiecare împrumut de carte. Fiecare împrumut este identificat prin **id_imprumut**, iar datele legate de împrumut, precum data împrumutului, data scadenței și data returnării, ajută la monitorizarea și gestionarea împrumuturilor în bibliotecă. Legăturile cu tabelele Carti și Membri facilitează identificarea cărților împrumutate și a membrilor care le-au împrumutat.

Membri:

Tabela Membri conține detalii despre membrii bibliotecii. Atributul id_membru reprezintă cheia unică pentru fiecare membru. Detalii precum nume, prenume, număr de telefon, dată de înregistrare, dată de naștere și gen contribuie la gestionarea și comunicarea eficientă cu membrii bibliotecii.

Gestiunea Bibliotecii - Descriere Relații Cheie

Proiectul de bază de date pentru bibliotecă a fost meticulos proiectat pentru a gestiona eficient resursele și informațiile. Relațiile cheie dintre entitățile definitorii evidențiază complexitatea sistemului și modul în care acestea interacționează:

1. Relația 1:1 între Membri și Carduri_de_biblioteca:

În cadrul bibliotecii, fiecare membru este asociat cu un card de bibliotecă unic, înregistrând detaliile specifice ale membrului și informațiile cardului său. Această conexiune strânsă între entități asigură evidența precisă a datelor membrilor și a stării cardurilor lor.

2. Relația 1:n între Membri și Imprumuturi:

Pentru a gestiona eficient împrumuturile, fiecare membru are capacitatea de a efectua mai multe împrumuturi de cărți. Astfel, relația permite bibliotecii să urmărească istoricul împrumuturilor pentru fiecare membru în parte, facilitând gestionarea și respectarea regulilor de returnare.

3. Relația m:1 între Autori și Carti:

În cadrul bibliotecii, se observă o relație de tipul **m:1** între entitățile **Autori** și **Carti**.
Această conexiune permite ca un autor să fie asociat cu mai multe cărți, reflectând capacitatea fiecărui autor de a contribui la mai multe opere literare.
În același timp, această relație indică posibilitatea ca o carte să aibă mai mulți autori, oferind un cadru adecvat pentru colaborări extinse și proiecte literare comune.

4. Relația m:1 între Carti și Categorii:

Pentru a organiza și clasifica colecția de cărți, se folosește o relație **m:1** între **"Carti"** și **"Categorii".** Mai multe cărți pot aparține aceleiași categorii, facilitând categorizarea eficientă a resurselor bibliotecii și oferind utilizatorilor o modalitate structurată de a explora și a găsi cărți pe baza categoriilor tematice.

Descrierea constrângerilor

Tabela autori:

Cheia primară **(autori_pk)** asigură unicitatea identificatorului autorului.
Cheia străină **(autori_carti_fk)** stabilește legătura cu tabela carti, referindu-se la identificatorul cărții pentru care a scris autorul.

Tabela carduri_de_biblioteca:

Cheia primară **(carduri_de_biblioteca_pk)** asigură unicitatea identificatorului cardului.
Cheia străină **(cdb_membri_fk)** asigură că fiecare card este asociat cu un membru al bibliotecii, referindu-se la identificatorul membrului.
Secvența **carduri_id_card_seq** definește o secvență pentru auto-incrementarea identificatorului cărții care incepe de la 1 si incrementeaza cu 1.
Si check **data_expirarii** sa fie mai mare decat data actuala in care se inscrie membrul respectiv

Tabela carti:

Cheia primară **(carti_pk)** asigură unicitatea identificatorului cărții.
Cheia străină **(carti_categorii_fk)** leagă tabela de categorii, referindu-se la identificatorul categoriei cărții.
Constrângerea de unicitate **carti_titlu_un** garantează că titlul fiecărei cărți este unic.
Constrângerea **check_title** verifică dacă titlul are cel puțin două caractere și conține doar litere și spații.
Constrângerea **check_autor** asigură că numele autorului are cel puțin două caractere și conține doar litere și spații.
Constrângerea **check_data_publicare** verifică dacă data publicării este mai mică decât data curentă.
Secvența **carti_id_carte_seq** definește o secvență pentru auto-incrementarea identificatorului cărții care incepe de la 1 si incrementeaza cu 1.
Triglerul **carti_id_carte_trg** asigură auto-incrementarea identificatorului cărții înainte de inserarea unui nou rând în tabel.

Tabela categorii:

Cheia primară **(categorii_pk)** asigură unicitatea identificatorului categoriei.
Secvența **categorii_id_categorie_seq** definește o secvență pentru auto-incrementarea identificatorului cărții care incepe de la 1 si incrementeaza cu 1.

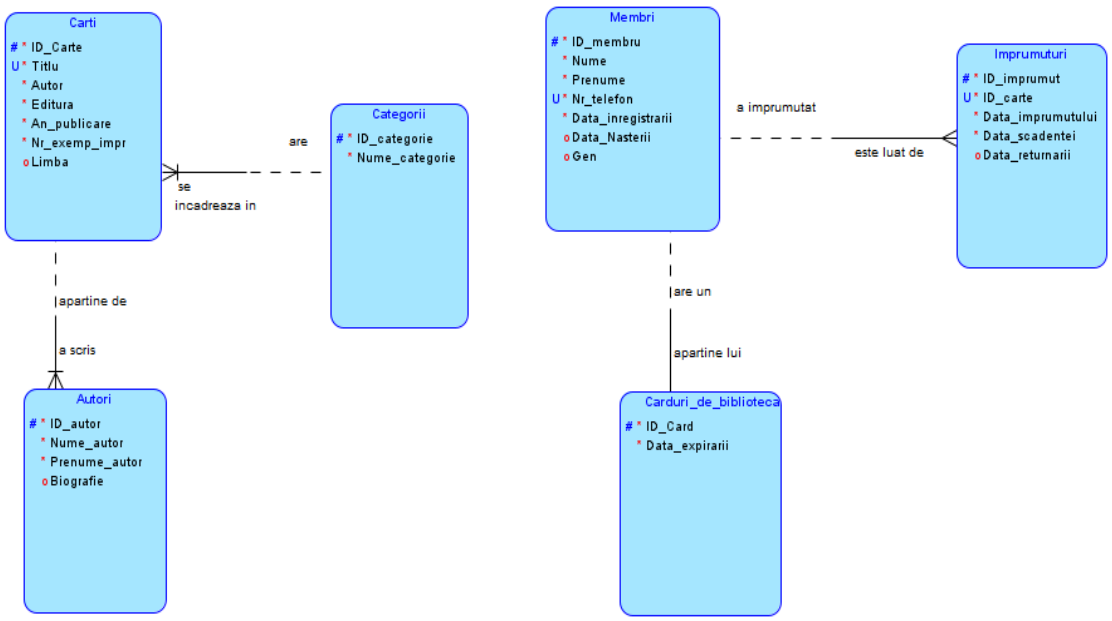
Tabela imprumuturi:

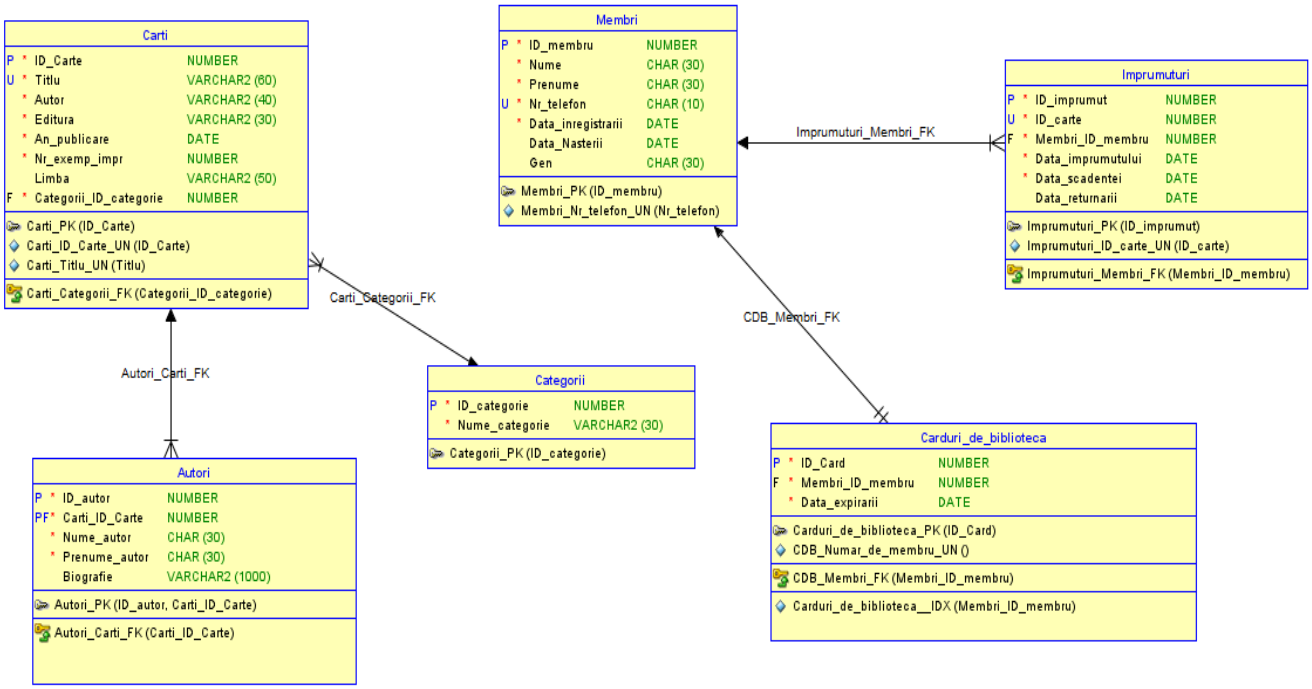
Cheia primară **(imprumuturi_pk)** asigură unicitatea identificatorului imprumutului.
Cheia străină **(imprumuturi_id_carte_un)** previne asocierea mai multor imprumuturi pentru aceeași carte.
Constrângerea **check_data_imprumutului** verifică dacă data imprumutului este mai mică decât data curentă.
Constrângerea **check_data_scadentei** asigură că data scadenței este mai mare decât data curentă.
Constrângerea **check_data_returnarii** verifică dacă data returnării este mai mare decât data curentă, doar în cazul în care cartea a fost returnată.
Secvența **imprumuturi_id_imprumut_seq** definește o secvență pentru auto-incrementarea identificatorului imprumutului care incepe de la 1 si incrementeaza cu 1.
Triglerul **imprumuturi_id_imprumut_trg** asigură auto-incrementarea identificatorului imprumutului înainte de inserarea unui nou rând în tabel.

Tabela membri:

Cheia primară **(membri_pk)** asigură unicitatea identificatorului membrului.
Constrângerea **check_nume** verifică dacă numele are cel puțin două caractere și conține doar litere și spații.
Constrângerea **check_prenume** asigură că prenumele are cel puțin două caractere și conține doar litere și spații.
Constrângerea **check_nr_telefon** verifică dacă numărul de telefon incepe cu 0, are exact 10 caractere și conține doar cifre.
Constrângerea **check_data_nastere** asigură că data nașterii este între 01-01-1948 și 01-01-2005 adica membrul are intre 18-75 de ani.
Constrângerea **check_data_inregistrarii** verifică dacă data înregistrării este mai mică decât data curentă.
Secvența **membri_id_membru_seq** definește o secvență pentru auto-incrementarea identificatorului membrului.
Triglerul **membri_id_membru_trg** asigură auto-incrementarea identificatorului membrului înainte de inserarea unui nou rând în tabel.

Diagrama Entitate Relatie





Tehnologii folosite pentru Front-end si Back-end

Pentru partea de Front-end Interfata prin care s-a facut legatura cu baza de date a fost implementata cu ajutorul limbajelor de programare python (unde am utilizat cx_Oracle pentru conectarea cu baza de date), flask, HTML, CSS.

Pentru partea de Back-end

Am utilizat limbajul SQL alaturi de Oracle SQL Developer Data Modeler(pentru a realiza modelul logic si relational) si Oracle SQL Developer(pentru a realiza scriptul de inserare, crearea bazei de date din codul generat de aplicatia Oracle SQL Developer Data Modeler si pentru a testa diferite comenzi pe care le-am implementat in aplicatie).

Conectarea la baza de date din aplicație

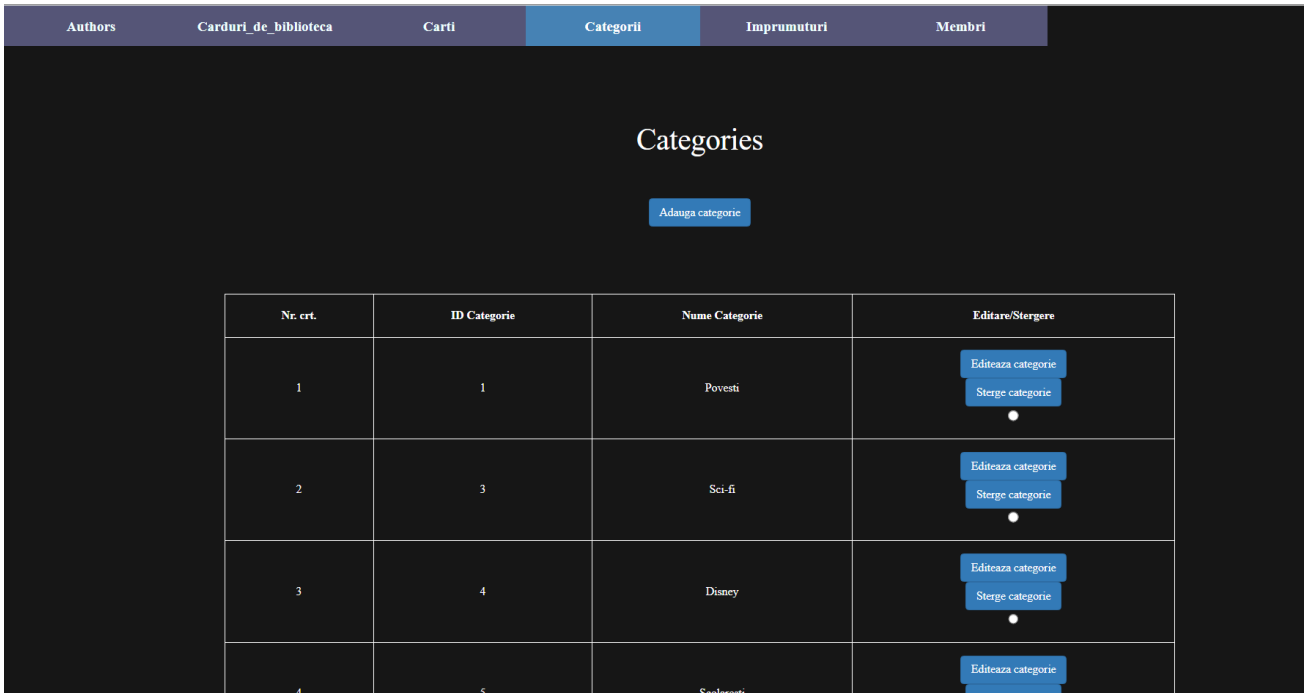
Pentru conectare s-a utilizat pachetul cx-oracle și Oracle client. Pentru a putea realiza conectarea, a fost necesară descărcarea de la companie a unui folder ce conține diferite fișiere și includerea lui în aplicație. Au fost necesare și informații legate de datele de autentificare.

```
10
11  cx_Oracle.init_oracle_client(os.path.join(os.getcwd(), "instantclient_21_7"))
12  dsn_tns = cx_Oracle.makedsn( *args: 'bd-dc.cs.tuiasi.ro', '1539', 'orcl')
13  con = cx_Oracle.connect("bd007", "parolaBD2023", dsn_tns)
14
```

Exemple si instructiuni SQL folosite si aspect in aplicatie

Select:

-folosit pentru a extrage datele din baza de data(de exemplu pentru a afisa lista cu categorii)

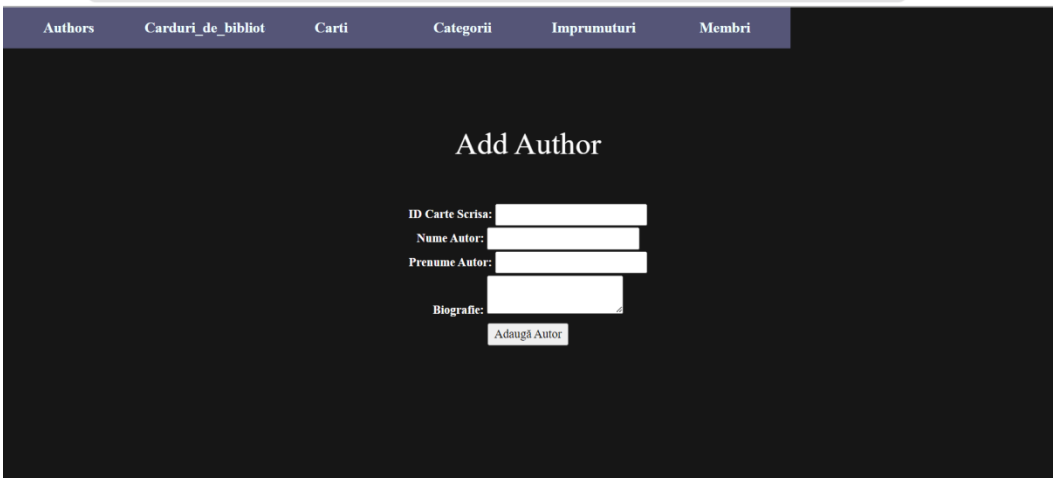


```python
@app.route('/categories')
def categories():
    categories = []

    cur = con.cursor()
    cur.execute('SELECT * FROM categorii')  # Modificati cu numele real al tabelului categorii
    for result in cur:
        category = {
            'id_categorie': result[0],
            'nume_categorie': result[1],
        }
        categories.append(category)
    cur.close()
    return render_template( template_name_or_list: 'categorii.html', categories=categories)
```

Insert:

-folosit pentru inserarea datelor in tabelele din baza de date(de exemplu sa adaugam noi autori)




```python
def save_author():
    error = None
    if request.method == 'POST':
        # Conectarea la baza de date și codul pentru inserarea datelor
        emp = 0
        cur = con.cursor()
        cur.execute('SELECT MAX(id_autor) FROM autori')
        for result in cur:
            emp = result[0]
        cur.close()

        emp += 1
        cur = con.cursor()
        values = []
        values.append("'" + str(emp) + "'")
        values.append("'" + request.form['carti_id_carte'] + "'")
        values.append("'" + request.form['nume_autor'] + "'")
        values.append("'" + request.form['prenume_autor'] + "'")
        values.append("'" + request.form['biografie'] + "'")

        fields = ['id_autor', 'carti_id_carte', 'nume_autor', 'prenume_autor', 'biografie']
        query = 'INSERT INTO autori (%s) VALUES (%s)' % (', '.join(fields), ', '.join(values))

        cur.execute(query)
        con.commit()


    return redirect('/authors')  # Aici redirecționăm înapoi la pagina autorilor după adăugare
```

Delete:
-folosit pentru a sterge unele inregistrari dintr-o tabela

```
71  @app.route( rule: '/delAuthor', methods=['POST'])
72  def del_author():
73      emp = request.form['author_id']
74      cur = con.cursor()
75      cur.execute('delete from autori where id_autor=' + emp)
76      cur.execute('commit')
77      return redirect('/authors')
78
```

S-a procedat la fel si pentru tabelele carti,imprumuturi,membri,carduri de biblioteca , iar acolo unde a fost nevoie datele s-au sters in cascada.