

**UNIVERSITATEA „POLITEHNICA” din BUCUREȘTI**

**Facultatea de Electronică, Telecomunicații și Tehnologia Informației**

**Proiect1**

**Programarea Interfețelor pentru Baze de Date**

**Student: Zaharia Roxana-Florina Prof. coordonator:**

**Grupa 435B Ș.l. Dr. Ing. Pupezescu Valentin**

**București 2025**

Cuprins

[**1.** **Cerinta proiectului** 3](#_Toc157360688)

[**2.** **Tehnologii utilizare** 3](#_Toc157360689)

[**3.** **Descrierea aplicatiei** 5](#_Toc157360690)

[**5.** **Functionalitatea aplicatiei** 7](#_Toc157360691)

[**6.** **Implementarea functiilor** 10](#_Toc157360693)

[**7.** **Front-end** 18](#_Toc157360694)

[**8.** **Concluzii** 20](#_Toc157360695)

[**9.** **Bibliografie** 20](#_Toc157360696)

# **Cerinta proiectului**

Proiectul propus vizează crearea a două aplicații cu o bază de date implementată în sistemul de gestionare a bazelor de date MySQL și două interfețe pentru accesarea acestei baze de date. Interfețele pot fi dezvoltate folosind tehnologii la alegere, precum JSP, Hibernate, JPA, .NET, Python etc. Structura bazei de date va fi definită conform unei liste predefinite de tabele, cu relații și asocieri specificate între acestea.

Scopul aplicațiilor constă în facilitarea utilizatorilor pentru efectuarea operațiunilor de bază de date, precum vizualizarea, adăugarea, modificarea și ștergerea datelor. Se presupune că interfețele vor oferi o vizualizare eficientă a datelor interconectate între diferite tabele, evidențiind relațiile dintre acestea.

# **Tehnologii utilizare**

**Descrierea sistemului de gestiune a bazelor de date MySQL**

* *MySQL* este un sistem de gestionare a bazelor de date relaționale (RDBMS) open-source, dezvoltat pentru a oferi un mediu eficient și fiabil pentru stocarea și manipularea datelor.
* *MySQL* este un sistem de gestiune a bazelor de date (SGBD) open-source, care este apreciat pentru accesibilitatea, ușurința de utilizare și costuri reduse, datorită naturii sale gratuite. Este recunoscut pentru performanțele rapide și pentru fiabilitatea în gestionarea volumelor mari de date, fiind astfel adecvat atât pentru proiecte mici, cât și pentru utilizare la scară largă. MySQL suportă o varietate de limbaje de programare și rulează pe diverse sisteme de operare, oferind scalabilitate și flexibilitate în dezvoltarea aplicațiilor web. Securitatea robustă integrată și comunitatea vastă de utilizatori contribuie la popularitatea sa continuă ca o soluție de bază de date preferată în industrie.

**Tehnologia JSP**

* *JSP,* sau JavaServer Pages, este o tehnologie care permite dezvoltatorilor să creeze pagini web dinamice bazate pe Java, integrând cod Java în pagini HTML sau XML. Este o parte crucială a stack-ului tehnologic Java, fiind utilizată pentru a construi interfețe de utilizator pentru aplicații web. JSP este valoroasă pentru că facilitează separarea părții logice a aplicației (Java) de design-ul front-end (HTML/CSS), făcând codul mai ușor de gestionat și de întreținut.

**HTML & CSS**

* *HTML* (HyperText Markup Language) și *CSS* (Cascading Style Sheets) sunt două tehnologii fundamentale pentru crearea paginilor web. HTML este limbajul de bază utilizat pentru a structura conținutul pe web, permițând dezvoltatorilor să definească diferite părți ale unei pagini, cum ar fi paragrafe, titluri, link-uri și imagini. Pe de altă parte, CSS este utilizat pentru a controla designul vizual al acestor elemente HTML, inclusiv layout-ul, culorile, fonturile și animațiile.

# **Descrierea aplicatiei**

Ca temă am abordat gestionarea filmelor, a locatiilor si a protagonistilor , creând o interfață Web ce operează baza de date în care sunt stocate toate datele necesare. Funcționalitățile implementate în cadrul acestei etape a proiectului se enumeră vizualizarea datelor, adăugarea lor, modificarea și posibilitatea ștergerii acestora, într-o manieră elegantă și intuitivă.

**Implementare**

Baza de date a fost realizată cu ajutorul MySQL Workbench, fiind alcătuită din două tabele: *filme* și *locatii*. Pentru a realiza o asociere între aceste două tabele de tip M:N, a fost creată o tabelă intermediară numită *protagonist*.

**Structura coloanelor tabelelor:**

1. Tabela *trupa*
   * idtrupa ( INT, PK, NN, AI)
   * nume ( VARCHAR(45) )
   * gen\_muzical ( VARCHAR(45) )
   * nr\_membrii ( VARCHAR(45) )
   * producatorFilm ( VARCHAR(45) )
2. Tabela *festival*
   * idfestival ( INT, PK, NN, AI)
   * nume ( VARCHAR(45) )
   * data ( VARCHAR(45) )
   * pret\_bilet ( VARCHAR(45) )
3. Tabela *lista*
   * idlista ( INT, PK, NN, AI)
   * idfestival ( INT)
   * idtrupa ( INT)
   * ora\_trupa ( VARCHAR(45) )
   * dansatori ( VARCHAR(45) )
   * scena ( VARCHAR(45) )

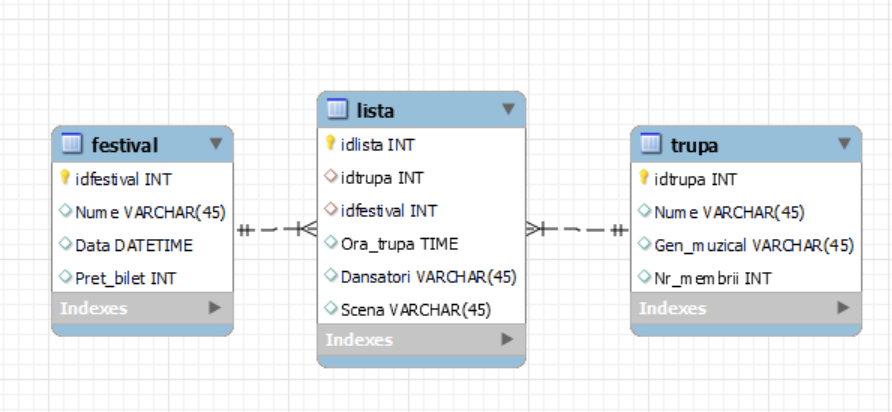


Fig. 1 – Diagrama EER a tabelelor și conținutului acestora

**Ce relatii exista intre cele trei tabele?**

* Intre **festival** si **lista** exista o asociere de tip 1:N
* Intre **trupa** si **festival** exista o asociere de tip M:N
* Intre **trupa** si **lista** exista o asociere de tip 1:N

Datorită asocierii M:N, *idtrupa* și *idfestival* sunt setate ca chei străine (foreign keys) în cadrul tabelei *lista*.

O imagine care conține text, număr, Font, linie

Descriere generată automat

**Diagrama UML pentru tehnologia JSP**

**O imagine care conține diagramă, schiță, desen, text

Descriere generată automat**

Fig. 2 – Diagrama implementarii fisierelor ce alcatuiesc pagina web

# **Functionalitatea aplicatiei**

**Arhitectura proiectului**

Proiectul realizat in tehnologia JSP are urmatoarea structura:

• Un pachet DB care conține o clasă JavaBean, având rolul de a furniza toate funcționalitățile principale ale interfeței bazei de date, inclusiv gestionarea conectivității și operațiile de afișare, adăugare, modificare și ștergere.

Fișierul **JavaBean.java** constituie "motorul" aplicației și se ocupă de operațiile necesare pentru conectarea la baza de date și gestionarea tabelelor și datelor. Acesta conține o singură clasă denumită *JavaBean*, în care sunt implementate metodele esențiale pentru funcționarea aplicației, inclusiv:

1. **Vizualizarea Tabelor:**
   * Metodele **vedeTabela** și **vedeLista** au rolul de a afișa conținutul tabelelor specificate.
2. **Inserarea de Date în Tabele:**
   * Metodele **adaugaLista**, **adaugaFestival** și **adaugaLista** permit adăugarea de date noi în tabelele corespunzătoare.
3. **Ștergerea Datelor Existente:**
   * Metodele **stergeDateTabela** și **stergeTabela** se ocupă de eliminarea datelor existente din tabelele respective.
4. **Modificarea Datelor:**
   * Metoda **modificareTabela** gestionează actualizarea sau modificarea datelor într-o anumită tabelă.
5. **Returnarea unei Linii Bazate pe ID:**
   * Metodele oferă posibilitatea de a returna o linie specifică dintr-o tabelă, utilizând ID-ul asociat acelei înregistrări.

• Un folder webapp care cuprinde toate paginile JSP. Aceste pagini au rolul de a importa funcționalitățile din clasa JavaBean, fiind interconectate și, implicit, conectate la pachetul DB. Acest aranjament realizează partea dinamică a proiectului, facilitând interacțiunea între interfața utilizatorului și baza de date.

**Index.html**, pagina principală a interfeței, este construită integral cu limbajul de marcare HTML și oferă acces la panoul de control al aplicației noastre web. Aici avem posibilitatea să adăugăm date noi, să modificăm informațiile existente sau să ștergem datele din tabela pe care dorim să o gestionăm.

O imagine care conține captură de ecran, text, software, Sistem de operare

Descriere generată automat

Fig. 3 – Pagina principala a interfetei

O imagine care conține text, captură de ecran, software, număr

Descriere generată automat

Fig. 4 – Pagina destinate vizualizarii tabelei *trupa* cu posibilitatea de stergere

# O imagine care conține text, captură de ecran, software, Font Descriere generată automat

Fig. 5 – Pagina destinate modificarii unei linii din tabela *trupa*

După cum se poate observa, utilizatorul are posibilitatea de a accesa pagina aferentă vizualizării și gestionării datelor pentru trupa, festival și lista, stocate în tabelele cu aceleași nume. Se remarca existenta butoanelor aferente operatiilor de : adaugare, stergere, modificare, precum si un buton destinat revenirii pe pagina principala.

# **Implementarea functiilor**

Toate paginile JSP conțin fragmente de cod Java care furnizează comportamentul dinamic al aplicației. Pentru a utiliza funcțiile din clasa Java, trebuie să folosim tag-urile JSP. Un exemplu poate fi găsit în pagina JSP numită "tabela\_filme":

O imagine care conține text, captură de ecran, Font

Descriere generată automat

Fig 1 – cod tabela\_Trupa

Obiectul "jb" reprezintă o instanță a clasei JavaBean, stabilind astfel o legătură direct între paginile JSP și funcțiile pe care dorim să le implementăm.

Instrucțiunea "property="\*" implementează utilizarea tuturor metodelor din clasa JavaBean." Aceasta înseamnă că toate metodele din clasa JavaBean pot fi accesate și utilizate în paginile JSP prin intermediul obiectului "jb", fără a fi nevoie să specificăm fiecare metodă în parte. Această abordare simplifică interacțiunea cu clasa JavaBean și permite paginilor JSP să acceseze și să utilizeze toate funcționalitățile disponibile în mod dinamic.

De asemenea, am creat un formular, in cadrul caruia prin instructiunea jb.connect() ne legam la functia connect() din clasa JavaBean:

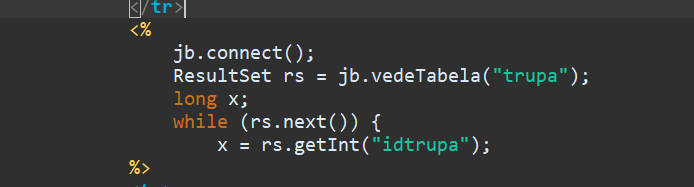


Fig 2 – cod tabela\_Trupa

* **Conexiunea**

Obiectul “con” este cel prin intermediul caruia se realizeaza toate operatiile pe baza de date. In metoda connect() : se incarca driver-ul de MySQL. DriverManager contine mai multe drivere de conectica, mai multe SGBD-uri. Prin metoda getConnection(…) apelam metoda de conectare si are ca atribute baza de date realizata in MySQL, user-ul si parola utilizatorului.

O imagine care conține text, captură de ecran, Font

Descriere generată automat

Fig 3 – cod JavaBean

O imagine care conține text, captură de ecran

Descriere generată automat

Fig 4 – cod JavaBean

Functia disconnect() realizeaza incheierea close() pe obiectul de conexiune con.

* **Afisarea**

Funcția din clasa JavaBean numită "vedeTabela()" primește ca atribut numele tabelei pe care dorim să efectuăm interogarea. Această funcție returnează un set de rezultate sub forma unui obiect de tip "results". La început, acest obiect "results" este declarat ca fiind null, adică inițial nu conține nicio informație. Funcția "vedeTabela()" va executa interogarea specifică asupra tabelului "filme" din baza de date și va obține rezultatele. Aceste rezultate, care reprezintă rânduri de date din tabelul "filme", vor fi stocate în obiectul "results". Astfel, prin apelarea acestei funcții și preluarea obiectului "results", paginile JSP pot accesa și afișa datele din tabela "filme" în mod dinamic.

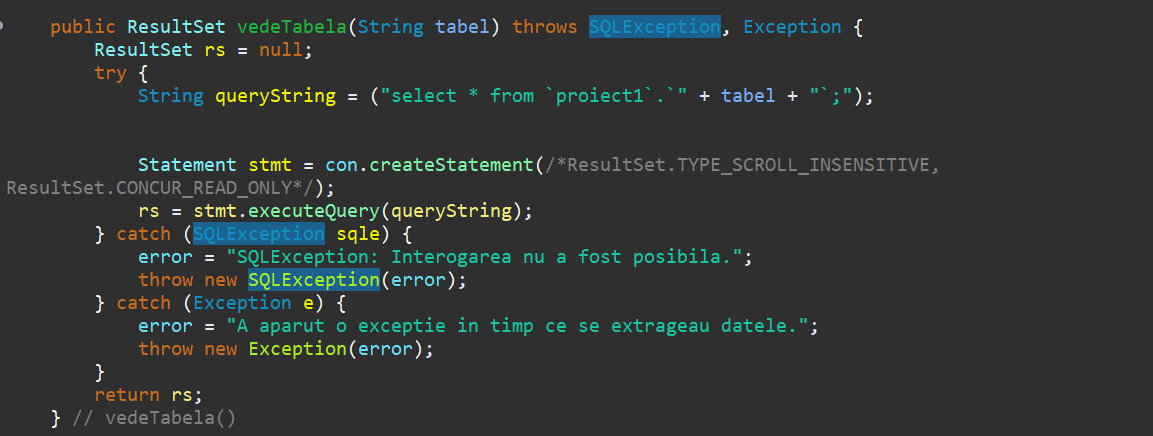


Fig 5 – cod JavaBean

O imagine care conține text, Font, captură de ecran

Descriere generată automat

Fig 6 – cod tabela\_Trupe

Checkbox-urile sunt asociate valorile idfilme specific fiecarui film din baza de date, astfel afisarea se face linie cu linie.

* **Stergerea**

Pentru a realiza stergerea, va trebui bifat checkbox-ul filmului care se doreste a fi sters apoi apasarea butonului *Sterge liniile marcate*, astfel se va face legatura cu fisierul sterge\_filme.jsp.

O imagine care conține text, Font, captură de ecran

Descriere generată automat

Fig 7 – tabela\_Trupe

Pentru a realiza stergerea, va trebui bifat checkbox-ul filmului care se doreste a fi sters apoi apasarea butonului *Sterge liniile marcate*, astfel se va face legatura cu fisierul sterge\_filme.jsp.

Prin metoda **POST**, valorile ascunse in spatele checkbox-urilor sunt transmise prin functi getParameter().

O imagine care conține text, captură de ecran, Font

Descriere generată automat

O imagine care conține text, captură de ecran, Font

Descriere generată automat

Fig 8 - sterge\_Trupe

Utilizatorul selectează date pe paginile JSP, stocate în vectorul "s". Funcția "stergeDateTabela" din clasa JavaBean este apelată cu numele tabelei și câmpul după care se efectuează ștergerea. Aceasta utilizează vectorul "s" pentru a identifica înregistrările de șters din baza de date, facilitând manipularea dinamică a datelor.

O imagine care conține text, captură de ecran, software

Descriere generată automat

Fig 9 - JavaBean

Instrucțiunea "if (con != null)" verifică dacă există o conexiune validă cu baza de date. Aceasta este o măsură de precauție pentru a asigura că se poate lucra cu baza de date înainte de a efectua orice operație asupra ei. Se declară un obiect numit "delete" de tip **PreparedStatement**. Această declarație se face pe conexiunea "con". PreparedStatement este utilizat pentru a pregăti o instrucțiune SQL care va fi ulterior executată. Instrucțiunea SQL include o interogare de tip DELETE care are o condiție parametrizată cu un semn întrebare ("?"). Acest semn întrebare indică o valoare care va fi înlocuită mai târziu cu valoarea specificată. Valorea "aux" este înlocuită în locul semnului întrebare și apoi instrucțiunea SQL este executată. Aceasta va efectua ștergerea înregistrărilor din baza de date care corespund criteriilor specificate în interogare.

* **Adaugarea**

Utilizatorii pot adăuga informații într-un formular. După ce completează formularul și apasă butonul "Adauga o noua trupa", datele sunt preluate din formular și stocate în variabile utilizând obiectul "request".

O imagine care conține text, captură de ecran, software

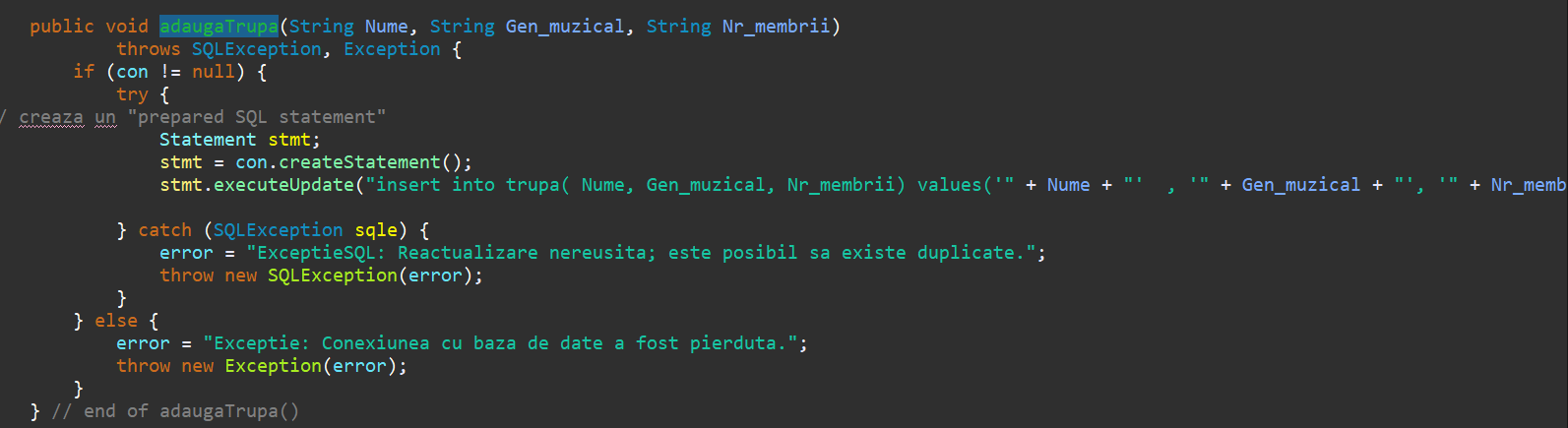
Descriere generată automat

Fig 10 – nou\_Trupa

Dacă utilizatorul furnizează datele corect, atunci se stabilește o conexiune cu baza de date și datele sunt adăugate folosind metoda "adaugaTrupa" din clasa JavaBean ("jb"). Dacă adăugarea reușește, utilizatorul primește un mesaj de confirmare. În caz contrar, formularul este afișat din nou pentru corectare. La finalul paginii, există un link către pagina principală "Home" pentru a permite utilizatorului să se întoarcă la pagina principală a aplicației.

La apasarea butonului *Adauga o noua trupa* se va face legatura cu functia adaugaTrupa() din clasa JavaBean :

Fig 11 - JavaBean



Asemenea operatiilor anterioare, se verifica daca avem conexiune cu baza de date si se executa o comanda specifica MySQL ( insert into trupa(...) ).

* **Modificarea**

Se realizeaza un formular in cadrul caruia se ruleaza aceleasi instructiuni ca in cazul operatiei de afisare a datelor. Prin structura repetitiva *while* se plaseaza cursorul pe prima linie, si se preia din “idfilme” valoarea id-ului ce se leaga de valorile pe care dorim sa le afisam. Singura diferenta este ca formularul are ca si “action” fisierul m1\_filme.jsp, ce ia locul operatiei de adaugare.

O imagine care conține text, captură de ecran

Descriere generată automatO imagine care conține text, captură de ecran, Font

Descriere generată automat

Fig 13 - JavaBean

Fig 12 – modifica\_Trupa

Metoda "intoarceLinieDupaId," implementată în clasa JavaBean, servește la selectarea datelor asociate unui film specific identificat după ID. De obicei, este folosită pentru a extrage informațiile unui film pe care dorim să le modificăm. În cadrul implementării, se utilizează o instrucțiune specifică MySQL, precum "SELECT \* FROM ...", pentru a recupera datele dorite din baza de date.

În fișierul m1\_filme.jsp, au fost create câmpuri de tip INPUT care sunt utilizate pentru introducerea noilor date. Acest formular este legat de fișierul m2\_filme.jsp, unde se efectuează procesul de modificare propriu-zis.

În m2\_filme.jsp, utilizatorii pot introduce noile date pentru film, iar aceste date vor fi procesate pentru a actualiza informațiile existente în baza de date. Această secțiune a aplicației permite utilizatorilor să modifice și să actualizeze detaliile filmului într-un mod interactiv.

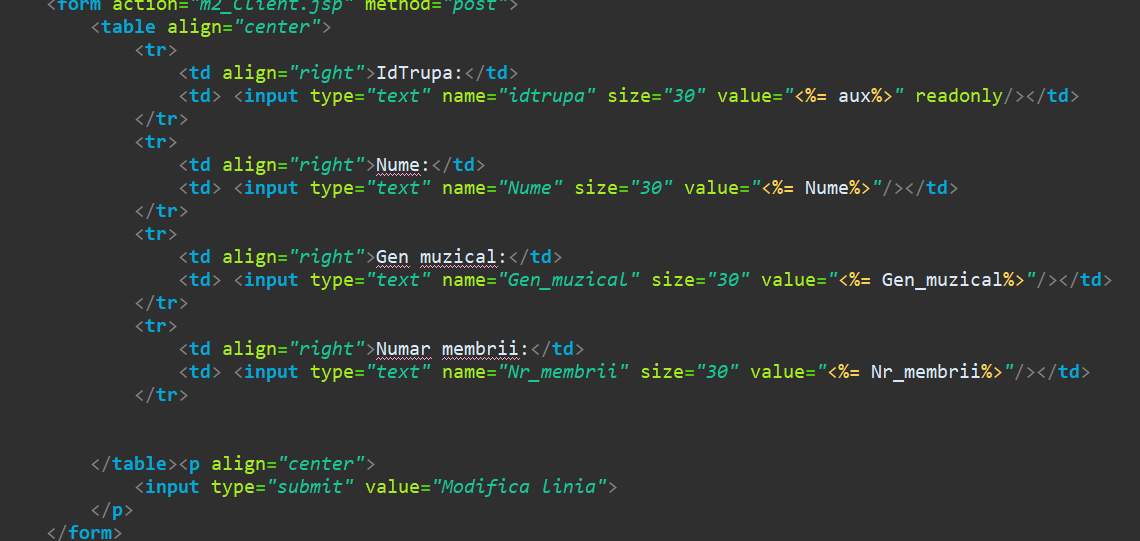


Fig 14 – m1\_Trupa

O imagine care conține text, captură de ecran, Font

Descriere generată automat

Fig 15 – m2\_Trupa

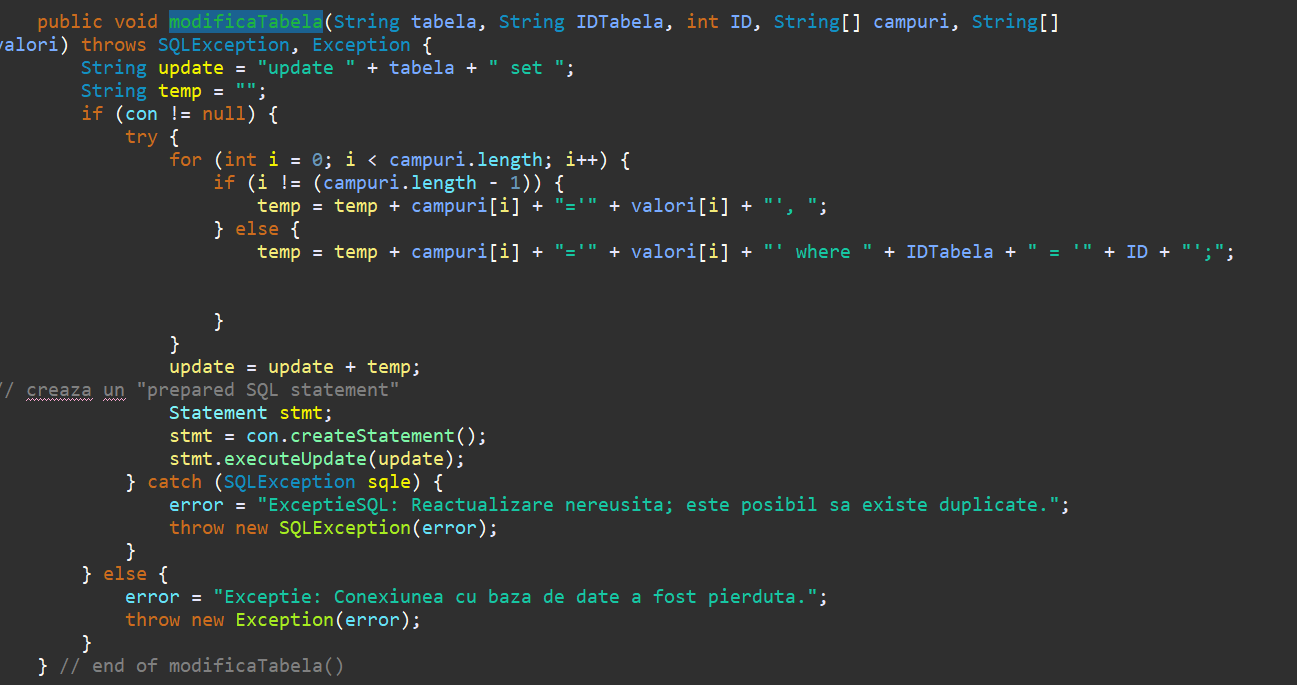
1

Fig 16 - JavaBean

# **Front-end**

Pentru interfata am ales un aspect simplu dar ingrijit, usor de inteles si utilizat folosind limbajul HTML combinat cu CSS.

O imagine care conține text, captură de ecran

Descriere generată automat

Fig 17 – index.html (1)

O imagine care conține text, captură de ecran, software, număr

Descriere generată automat

Fig. 6 – Pagina destinate vizualizarii tabelei *festival* cu posibilitatea de stergere

O imagine care conține text, captură de ecran, software, Sistem de operare

Descriere generată automat

Fig. 7 – Pagina destinate modificarii unei linii din tabela *trupa*

O imagine care conține text, captură de ecran, număr, Font

Descriere generată automat

Fig. 8 – Pagina destinate vizualizarii tabelei *lista* cu posibilitatea de stergere

# **Concluzii**

Bazele de date sunt esențiale în toate domeniile și sectoarele de afaceri. "Gestionare filme" o aplicație dezvoltată în tehnologia JSP, exemplifică o arhitectură eficientă, separând funcționalitățile între back-end și front-end. Aceasta permite utilizatorilor să gestioneze operațiuni cheie pe o bază de date MySQL, inclusiv afișarea, adăugarea, modificarea și ștergerea datelor. Tehnologia JSP se dovedește eficientă în dezvoltarea de soluții web, oferind o interfață intuitivă și facilitând gestionarea datelor. Prin separarea clară a componentelor, se obține un design modular și o administrare ușoară, contribuind la o experiență utilizator îmbunătățită și gestionare eficientă a datelor.

# **Bibliografie**

1. Cursurile PIBD
2. "JavaTpoint - Learn Tutorials Point with Examples," - https://www.javatpoint.com
3. "FreeCodeCamp Forum," [Online]. - https://forum.freecodecamp.org
4. "Stack Overflow - Where Developers Learn, Share, & Build Careers," - https://stackoverflow.com
5. “W3school” - https://www.w3schools.com/html/default.asp