O imagine care conține text, Font, siglă, simbol

Descriere generată automatSigla comerciantuluiU.N.S.T. POLITEHNICA București

## Facultatea de Electronică, Telecomunicații și Tehnologia Informației

**Proiect (1)**

**Programarea interfatelor pentru baze de date**

Student: NISTOR ELENA DANIELA

Grupa: 434D

Profesor coordonator: Ș.l. Dr. Ing. Pupezescu Valentin

An universitar 2023-2024

**Cuprins**

1. **CERINTA TEMEI...................................................................pag. 3**
2. **DESCRIEREA TEHNOLOGIILOR UTILIZATE.........................pag.3**
   1. **MySQL.......................................................................pag.3**
   2. **JSP.............................................................................pag.3**
   3. **ApacheTomcat..........................................................pag.3**
   4. **HTML.........................................................................pag.4**
   5. **CSS...........................................................................pag.4**
3. **IMPLEMENTARE.................................................................pag.4**
   1. **IMPLEMENTAREA BAZEI DE DATE...........................pag.4**
   2. **IMPLEMENTAREA CALSEI JavaBean........................pag.7**
   3. **IMPLEMENTAREA JSP-URILOR................................pag.11**
4. **CONCLUZII........................................................................pag.18**
5. **BIBLIOGRAFIE...................................................................pag.19**

**1.CERINTA TEMEI**

Creaţi două aplicaţii care să conţină o bază de date creată în sistemul de gestiune a bazelor de date MySql şi două interfeţe la aceasta (baza de date este comună). La crearea interfeţelor se vor folosi două tehnologii(la alegere - ex.: JSP, Hibernate, JPA, .NET, Python etc.).

Interfeţele vor trebui să permită utilizatorului să execute următoarele operaţii pe toate tabele(inclusiv pe cele de legătura dacă aveti asocieri de tip M:N): vizualizare, adăugare, modificare şi ştergere de date. Vizualizarea tabelelor de legătură va presupune vizualizarea datelor referite din celelalte tabele.

**2. DESCRIEREA TEHNOLOGIILOR UITILIZATE**

**2.1 MySQL**

MySQL este un sistem de gestionare a bazelor de date relaționale (RDBMS - Relational Database Management System) open-source, care oferă o platformă robustă pentru stocarea și gestionarea datelor. Este una dintre cele mai populare și larg utilizate baze de date în întreaga lume, folosită de la aplicații mici la cele de dimensiuni enterprise.

MySQL este o tehnologie fiabilă, având capacitatea de a gestiona sarcini critice și de a asigura disponibilitatea datelor, are funcții de recuperare a datelor și mecanisme de securitate pentru a proteja informațiile, performantă și flexibilă pentru gestionarea bazelor de date relaționale, utilizată într-o varietate de aplicații și scenarii, de la site-uri web și aplicații mobile la sisteme enterprise complexe. MySQL suportă multiple platforme și limbaje de programare, ceea ce îl face ușor de integrat în diverse medii de dezvoltare.

**2.2 JSP**

*Java Server Pages* este o simpla dar puternica tehnologie folosita pe partea de server pentru a genera continut HTML dinamic. JSP este o extensie directa a *Java Servlets* si furnizeaza o modalitate de a separa partea de procesare de cea de prezentare. Motorul JSP este doar un alt *Servlet*, mapat la extensia *\*.jsp*. Paginile JSP sunt create sa suporte mai multe tipuri de documente structurate, îndeosebi HTML si XML.

SP este compatibil cu tehnologii moderne de dezvoltare web, cum ar fi framework-urile JavaScript și librăriile de stilizare CSS, permițând dezvoltatorilor să creeze interfețe web avansate și atractive.

**2.3 Apache Tomcat**

Apache Tomcat este un server web open-source, funcționând ca un container servlet, dezvoltat de Apache Software Foundation. Acesta implementează specificațiile JavaServer Pages (JSP) și servlet, furnizând o platformă software pentru rularea aplicațiilor web dezvoltate în Java. În plus față de suportul pentru JSP și servlet, distribuția standard a Apache Tomcat include și funcționalitățile tradiționale ale unui server web, aliniate cu cele ale produsului Apache.

**2.4 HTML (Hyper Text Markup Language)**

HTML (HyperText Markup Language) este limbajul standard utilizat pentru crearea și structurarea paginilor web. Este compus dintr-o serie de elemente și tag-uri care definește structura și conținutul unei pagini web, cum ar fi titluri, paragrafe, imagini, link-uri etc. HTML permite dezvoltatorilor să creeze pagini web statice sau interactive prin adăugarea de interactivitate cu JavaScript și alte tehnologii.

**2.5 CSS (Cascading Style Sheets)**

CSS (Cascading Style Sheets) este un limbaj utilizat pentru stilizarea și formatarea paginilor web create în HTML. CSS permite dezvoltatorilor să definească aspectul și aspectul vizual al elementelor HTML, cum ar fi culorile, fonturile, marginile, dimensiunile etc. Utilizând CSS, dezvoltatorii pot crea pagini web estetice și atrăgătoare, și pot asigura coerența și consistența designului pe întreaga pagină sau întregul site.

**3. IMPLEMENTARE**

Ca temă am abordat gestionarea băncilor, a clienților acestora și a conturile pe care clienții le dețin la anumite bănci, creând o interfață Web ce operează baza de date în care sunt stocate toate datele necesare. Funcționalitățile implementate în cadrul acestei etape a proiectului sunt vizualizarea datelor, adăugarea lor, modificarea și posibilitatea ștergerii acestora.

Un sistem de gestiune al bazelor de date (SGBD) este o aplicație care gestionează toate aspectele ce țin de o bază de date. Un SGBD are următoarele funcții:

- Asigură crearea bazelor de date;

- Asigură salvarea în siguranță a datelor;

- Conține un set de instrucțiuni minime pentru transpunerea în limbaj de programare a unui algoritm elementar;

- Conține un set de comenzi detaliate pentru gestionarea datelor salvate în baza de date;

- Asigură o interfață ce poate fi utilizată atât de programatori, cât și de utilizatorii neexperimentați;

- Conține un set de instrucțiuni ce permite programatorului să creeze o interfață cu utilizatorul.

**3.1 IMPLEMENTAREA BAZEI DE DATE**

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

Structura coloanelor tabelelor:

* + Tabela banks:
    - bankid (tip BIGINT, primarykey, Not Null, Unsigned, Auto-Increment)
    - nume (tip VARCHAR(45))
    - adresa (tip VARCHAR(45))
    - persoanacontact (tip VARCHAR(45))
    - numartelefon ( tip VARCHAR(45))
    - email (tip VARCHAR(45))
  + Tabela clients:
    - clientid (tip BIGINT, primarykey, Not Null, Unsigned, Auto-Increment)
    - nume (tip VARCHAR(45))
    - prenume (tip VARCHAR(45))
    - telefon (tip VARCHAR(45))
    - email (tip VARCHAR(45))
    - datanasterii (tip VARCHAR(45))
    - sold (tip VARCHAR(45))
  + Tabela account:
    - accountid( tip BIGINT, primarykey, Not Null, Unsigned, Auto-Increment)
    - bankid (tip BIGINT, Unsigned)
    - clientid (tip BIGINT, Unsigned)
    - tip (tip VARCHAR(45))
    - startdate (tip DATE)
    - enddate (tip DATE)
    - lasttransactiondate (tip DATE)
    - lasttransactionamount (tip VARCHAR(45))
    - lasttransactiontype (tip VARCHAR(45))

O imagine care conține text, Font, număr, linie

Descriere generată automat

Fig1. Implementarea tabelei *banks*

O imagine care conține text, chitanță, Font, număr

Descriere generată automat

Fig. 2 Implementarea tabelei *banks*

O imagine care conține text, chitanță, număr, Font

Descriere generată automat

Fig3. Implementarea tabelei *account*

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

Descriere generată automat

Fig. 4 Asocierea cu tabela de legatura

Datorită asocierii M:N, *bankid* și *clientid* sunt setate ca chei străine (*foreign keys*) în cadrul tabelei *account*. Cheile primare corespunzătoare fiecărei tabele au fost setate cu proprietățile Not Null (NN), Auto-Increment (AI) și Unsigned (UN). Diagrama aferentă tabelelor și structurii acestora este reprezentată în Fig. 5.

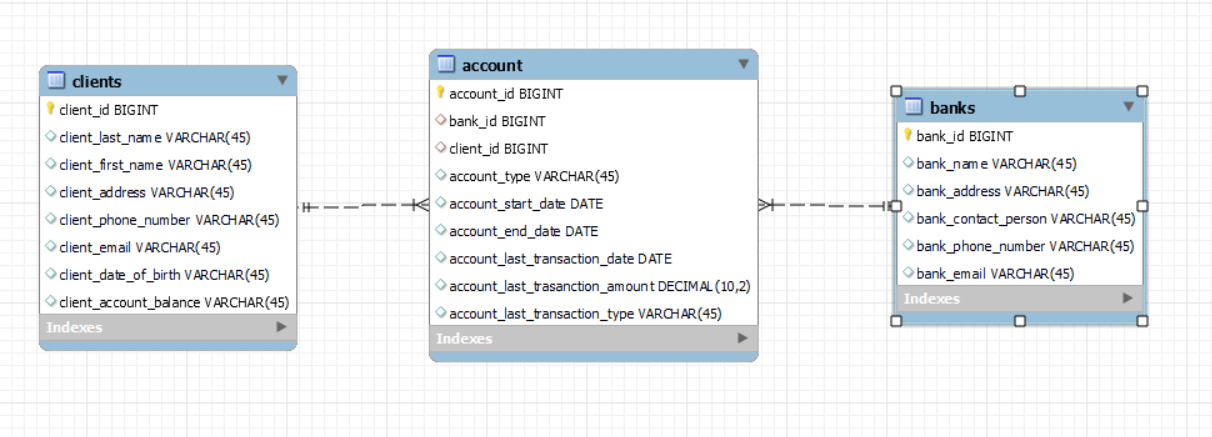


Fig. 5 Diagrama ERR

**3.2 IMPLEMENTAREA CLASEI JavaBean**

Pentru realizarea interfeței dintre utilizator și bază de date am ales tehnologia Java Server Pages (JSP), tehnologie ce facilitează crearea paginilor Web de dezvoltatorii software, pagini generate dinamic, bazate pe HTML, XML sau alte tipuri de documente. Cu toate că JSP este similar cu PHP (Hypertext Preprocessor) și ASP (Active Server Pages), acesta utilizează limbajul de programare Java.

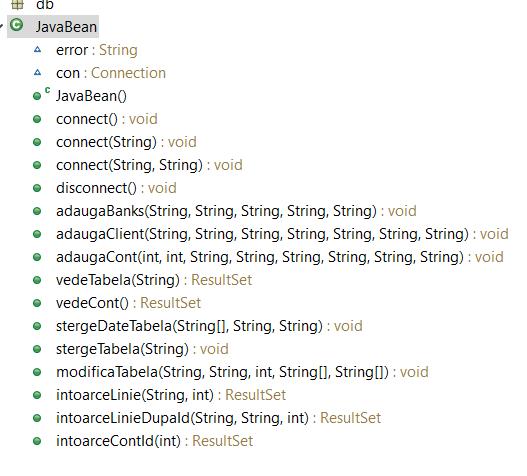


Fig. 6 Metodele de implementare ale clasei JavaBean

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

Descriere generată automatFișierul JavaBean.java, ce reprezintă “motorul” care realizează operațiile pentru conectarea bazei de date și gestionarea tabelelor și a datelor, conține o singură clasă numită JavaBean în care se regăsesc metodele implementate de care avem nevoie: vizualizare tabele (metodele vedeTabela și vedeCont), inserare de date în tabele (metodele adaugaBanks, adaugaClient și adaugaCont), ștergere date existente (metodele stergeDateTabela și stergeTabela), modificare date (metoda modificareTabela), dar și metode ce ne permit returnarea unei linii dintr-o tabelă folosind ID-ul acesteia ( intoarceLinie, intoarceLinieDupaId, intoarceContId).

Fig. 7 Realizarea conexiunii la baza de date

Aceste trei metode numite ***connect***, sunt folosite pentru a stabili o conexiune la o bază de date MySQL utilizând driver-ul JDBC (Java Database Connectivity). Toate cele trei metode utilizează driverul MySQL pentru încărcarea clasei și apoi folosesc DriverManager pentru a obține conexiunea la bază de date, iar în cazul în care apar excepții de tipul ClassNotFoundException, SQLException sau alte excepții, se returneaza un mesaj de eroare.

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

Descriere generată automatFig.8 Metoda ***adaugaBanks***

Aceasta metoda este folosita pentru a adăuga date despre bănci in tabela din baza de date MySQL. Metoda începe prin verificarea conexiunii la baza de date. Daca nu exista o conexiune valida metoda returnează un mesaj de eroare corespunzător. In cazul in care conexiunea la baza de date este activa, metoda continua prin crearea unei instrucțiuni SQL folosind obiectul ***Statement***. Aceasta instrucțiune este utilizata pentru a adăuga datele băncii in tabelul ***banks***. Daca se returnează un mesaj de eroare de tipul SQLException, acest lucru se întâmplă in cazul in care operația de inserare nu poate fi finalizata din cauza unor erori precum duplicare de cheie primara, neconcordanta intre denumirile sau tipurile atributelor.

O imagine care conține text, captură de ecran, Font

Descriere generată automatFig. 9 Metoda ***vedeTabela***

Aceasta metoda este utilizata pentru a extrage si returna datele dintr-un tabel specificat dintr-o baza de date MySQL. In primul rand se face o interogare SQL bazata pe numele tabelei primit ca parametru. Dupa interogare se creaza un obiect de tip Statment utilizand conexiune la baza de date, apoi interogarea SQL este executata utilizand metoda executeQuery, iar rezultatul este stocat intr-un obiect de tip ResultSet, care este returnat la finalul codului.

O imagine care conține text, captură de ecran, Font

Descriere generată automatFig. 10 Metoda ***stergeDateTabela***

O imagine care conține text, captură de ecran, Font

Descriere generată automat Acest cod defineste o metoda care este utilizata pentru a sterge inregistrarile dintr-un tabel specificat dintr-o baza de date MySQL, utilizand o cheie primara data ca parametru. Metoda incepe prin verificarea conexiunii la baza de date. In cazul in care conexiunea la baza de date este activa, se creeaza obiectul PeparedStatment pentru a sterge inregistrarile specificate din tabela pe baza unei chei primare. Bucla for este folosita pentru a itera prin toate cheile primare furnizate.

Fig. 11 Metoda ***modificaTabela***

Acest cod defineste o metoda care este utilizata pentru a modifica inregistrarile dintr-un tabel specificat dintr-o baza de date MySQL. Se construieste o instructiune SQL care este formata din numele tabelului si dintr-o serie de perechi de campuri si valori care urmeaza sa fie actualizate. Daca exista o conexiune activa la baza de date se creaza o instructiune de actualizare SQL folosind obiectul Statment.

**3.3 IMPLEMENTAREA JSP-URILOR**

**O imagine care conține text, captură de ecran, Font, document

Descriere generată automat** Pagina principală a interfeței ***index.html***, realizată exclusiv cu limbajul de marcare HTML și limbajul descriptiv CSS, ne oferă acces la panoul de control al aplicației noastre Web, prin care putem adăugate date noi, putem modifica datele existente sau le putem șterge din tabela pe care dorim să operăm

Fig. 12

După cum se poate observa, utilizatorul are posibilitatea de a accesa pagina aferentă vizualizării și gestionării datelor pentru bănci, clienți și conturi, stocate în tabelele cu aceleași nume.

**O imagine care conține text, captură de ecran, diagramă, Font

Descriere generată automat**

**Fig. 13 Interfata Web**

**Pentru a vedea detaliile aferente fiecarei tabele se va apasa butonul „Banks”, „Clients” sau „Account” in functie de dorintele utilizatorului.**

**Dupa efectuarea acestei selectii se va deschide pagina aferenta fiecarei tabelei ce va afisa datele acesteia si va permite efectuarea operatiilor CRUD.**

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

Descriere generată automat**

**Fig. 14 Tabela *clients***

Codul pentru cearea tabelei clienti si pentru legaturile cu operatiile de adaugare, stergere si modificare :

<body>

<h1 align=*"center"*> Tabela Clienti:</h1>

<div class=*"container"*>

<div class=*"box"*>

<i class=*"fas fa-user-plus icon"*></i>

<h3>Adauga client nou</h3>

<p><a href=*"nou\_client.jsp"*><button type=*"button"*>Adauga client</button></a></p>

</div>

<div class=*"box"*>

<i class=*"fas fa-edit icon"*></i>

<h3>Modifica client</h3>

<p><a href=*"modifica\_client.jsp"*><button type=*"button"*>Modifica client</button></a></p>

</div>

<div class=*"box"*>

<i class=*"fas fa-home icon"*></i>

<h3>Home</h3>

<p><a href=*"index.html"*><button type=*"button"*>HOME</button></a></p>

</div>

</div>

<hr>

<form action=*"sterge\_client.jsp"* method=*"post"*>

<table border=*"1"* align=*"center"*>

<tr>

<th>Mark</th>

<th>IdClient</th>

<th>Nume</th>

<th>Prenume</th>

<th>Adresa</th>

<th>Numar de telefon</th>

<th>Email</th>

<th>Data nasterii</th>

<th>Sold</th>

</tr>

<%

jb.connect();

ResultSet rs = jb.vedeTabela("clients");

**long** x;

**while** (rs.next()) {

x = rs.getInt("clientid");

%>

<tr>

<td><input type=*"checkbox"* name=*"primarykey"* value=*"*<%= x%>*"* /></td>

<td><%= x%></td>

<td><%= rs.getString("nume")%></td>

<td><%= rs.getString("prenume")%></td>

<td><%= rs.getString("adresa")%></td>

<td><%= rs.getString("telefon")%></td>

<td><%= rs.getString("email")%></td>

<td><%= rs.getString("datanasterii")%></td>

<td><%= rs.getString("sold")%></td>

<%

}

%>

</tr>

</table><br/>

<p align=*"center"*>

<input type=*"submit"* value=*"Sterge liniile marcate"*>

</p>

</form>

<%

rs.close();

jb.disconnect();

%>

</body>

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

Descriere generată automat

Fig. 15

**Codul pentru adaugarea unui nou client:**

<%

String Nume = request.getParameter("nume");

String Prenume = request.getParameter("prenume");

String Adresa = request.getParameter("adresa");

String Telefon=request.getParameter("telefon");

String Email = request.getParameter("email");

String Data\_nastere = request.getParameter("datanasterii");

String Sold = request.getParameter("sold");

**if** (Nume != **null**) {

jb.connect();

jb.adaugaClient(Nume, Prenume, Adresa, Telefon, Email, Data\_nastere,Sold);

jb.disconnect();

%>

<p>Datele au fost adaugate.</p><%

} **else** {

%>

Acesta apeleaza functia adaugaClient din clasa JavaBean.

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

Descriere generată automat

**Fig. 16**

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

Descriere generată automat**

**Fig. 17 Pagina aferenta modificarii datelor din tabela clienti**

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

Descriere generată automat**

**Fig. 18 Codul aferent modificarii datelor dintr-o tabela**

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

Descriere generată automat**

**Fig. 19 Codul JSP pentru efectuarea modificarilor din tabela clienti**

În schimb, dacă utilizatorul dorește să șteargă date, acesta va trebui să bifeze una sau mai multe intrări prin selectarea checkbox-urilor intrărilor ce se doresc a fi șterse, urmând apoi să apese butonul “Sterge liniile marcate”. Dacă operația de ștergere a fost realizată cu succes, utilizatorul va fi redirecționat către o pagina de informare a faptului ca datele au fost sterse cu succes. O imagine care conține text, captură de ecran, Font

Descriere generată automat

**Fig. 20 Codul JSP corespunzator paginii de stergere a datelor despre clienti**

**In acest cod JSP se apeleaza functia din clasa JavaBean „stergeDateTabela()”**

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

Descriere generată automat**

**Fig. 21**

**4. CONCLUZII**

În cadrul acestui proiect de programare a interfețelor pentru baze de date, am utilizat programul MySQL pentru a crea și gestiona o bază de date, iar apoi am dezvoltat o interfață utilizând limbajul de programare Java și tehnologia JSP (JavaServer Pages) în mediu Eclipse. Acest proiect ne-a oferit oportunitatea de a înțelege și de a aplica concepte cheie în dezvoltarea de aplicații web bazate pe baze de date.

Am invatat sa proiectam si sa implementam o baza de date folosint MySQL, inclusiv crearea tabelelor, definirea relatiilor si adaugarea datelor.

Am utilizat tehnologia JSP pentru a dezvolta interfata web a aplicatiei noastre. Am creat pagini JSP care au intercationar cu baza de date pentru a afisa, adauga, actualiza si sterge date.

Cu ajutorul limbajului de programare Java am gestionat conexiunea la baza de date, interogarea SQL si procesarea datelor din baza de date in cadrul aplicatiei noastre web.

In concluzie, acest proiect ne-a oferit o experienta valoroasa in dezvoltarea de aplicarii web bazate pe baza de date folosind tehnologii precum MySQL, Java si JSP. Am dobandit cunostinte practice despre proiectarea si implementarea schemelor de baza de date, dezvoltarea interfetei web si gestionarea conexiunilor si operatiilor CRUD cu baza de date. Acest proiect ne-a pregatit sa abordam si sa rezolvam provocarile reale din domeniul dezvoltarii de software si sa aplicam abilitatile dobandite in proiectele viitoare.

Partea superioară a formularului

**5. BIBLIOGRAFIE**

1. <https://www.scritub.com/stiinta/informatica/java/Tehnologia-JSP22577.php> - accesat la data de 3.02.2024
2. <https://koaha.org/wiki/Apache_Tomcat> - accesat la data de 3.02.2024
3. Prof. Pupezescu V., Curs de Programarea Interfețelor pentru Baze de Date 2023-2024
4. <https://www.w3schools.com/css/> - accesat la data de 5.02.2024
5. <https://www.w3schools.com/html/default.asp> - accesat la data de 5.02.2024
6. [Newest Questions - Stack Overflow](https://stackoverflow.com/questions) – accesat la data de 3.02.2024