

**PROGRAMAREA INTERFETELOR PENTRU**

**BAZE DE DATE**

STUDENT: **Sava Gabriel**

# GRUPA: **432B**

Profesor coordonator: **Ș. I. Dr. Ing. Pupezescu Valentin**  **CUPRINS**

1. **Tema proiectului**………………………………………………..…pag. 3
2. **Descrierea sistemului de gestiune a bazelor de date MySQL**……..pag. 3
   1. Ce este MySQL………………………………………………………
   2. Cum administreaza MySQL bazele de date………………………….
   3. De ce este atat de util MySQL………………………………………..
3. **Tehnologia Hibernate utilizata in dezvoltarea aplicatiei**………..pag. 4
   1. Ce este Hibernate………………………………………………………
   2. Arhitectura tehnologiei Hibernate…………………………………….
4. **Limbajul HTML si utilizatea sa in aplicatii**……………………….pag. 5
   1. Ce este HTML…………………………………………………………
   2. In ce scop utilizam limbajul HTML……………………………………
5. **Descrierea aplicatiei**…………………………………………………pag. 5
   1. **Baza de date**………………………………………………………pag. 5
   2. **Diagrama logica a bazei de date (Diagrama ERD)**…………….pag. 6
   3. **Functionalitatea aplicatiei**………………………………………pag. 7

4.**Arhitectura proiectului**…………………………………………pag. 8

5.**Implementarea functiilor**…………………………………….. pag. 9

6.**Partea de front-end**……………………………………………pag. 16

1. **Concluzii**……………………………………………………………pag. 17
2. **Bibliografie**………………………………………………………….pag. 18

## 1) Tema proiectului

Proiectul se concentrează pe crearea unei aplicații care include o bază de date gestionată prin MySQL. Se pot folosi diverse tehnologii, cum ar fi JSP, Hibernate. Interfețele aplicației vor permite utilizatorului să efectueze operații precum vizualizarea, adăugarea, modificarea și ștergerea datelor în toate tabelele. În cazul tabelelor de legătură, vizualizarea acestora va include datele asociate din celelalte tabele. Pentru tema individuală, am ales să utilizez două tehnologii distincte: JSP și Hibernate. În cadrul acestei prezentări, ne vom concentra pe tehnologia Hibernate, iar relațiile dintre tabelele bazei de date vor fi de tip M:N .

## 2) Descrierea sistemului de gestiune a bazelor de date MySQL

**2.1. Ce este MySQL?**

MySQL este un sistem de gestionare a bazelor de date relaționale open-source, utilizat în principal pentru aplicațiile online. Acesta poate crea și administra baze de date utile, cum ar fi cele care conțin informații despre angajați, inventar și multe altele, similar cu alte sisteme, cum ar fi Microsoft Access. Deși MySQL, Microsoft Access și alte sisteme de gestionare a bazelor de date au scopuri similare, și anume stocarea datelor, modul în care sunt utilizate diferă semnificativ.

MySQL face parte integrantă din platformele LAMP sau WAMP (Linux/WindowsApacheMySQL-PHP/Perl/Python). Popularitatea sa ca aplicație web este strâns legată de utilizarea PHP-ului, acestea fiind adesea folosite împreună și denumite „Duo-ul Dinamic”. În multe lucrări de specialitate, se menționează că MySQL este mult mai ușor de învățat și folosit decât alte aplicații de gestionare a bazelor de date, un exemplu în acest sens fiind comanda simplă de ieșire: „exit” sau „quit”.

**2.2. Cum administreaza MySQL bazele de date?**

Pentru a administra bazele de date MySQL se poate folosi modul linie de comanda sau, prin descarcare de pe internet, o interfata grafica: [MySQL Administrator](https://ro.wikipedia.org/w/index.php?title=MySQL_Administrator&action=edit&redlink=1) [și MySQL Query Browser. U](https://ro.wikipedia.org/w/index.php?title=MySQL_Query_Browser&action=edit&redlink=1)n alt instrument de management al acestor baze de date este aplicatia gratuita, scrisa în PH[P, phpMyAdmin. [](https://ro.wikipedia.org/wiki/PhpMyAdmin)2]

**2.3. De ce este atat de util MySQL?**

Baza de date MySQL este folosita in principal ca mijloc de a stoca date pentru aplicatii mari, bazate pe web. Site-uri precum WordPress, iStock, GitHub, Facebook, NASA, Marina SUA, Tesla, Scholastic, Spotify, YouTube, Netflix, Glasses Direct, Symantec (si multe altele) folosesc baza de date MySQL ca mijloc de stocare a datelor pe din interiorul sau exteriorul site-urilor web și serviciilor interne. [1]

## 3) Tehnologia JSP utilizata in dezvoltarea aplicatiei

1. **Tehnologia JSP utilizata in dezvoltarea aplicatiei**

* 1. **Ce este 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. [5]

Paginile JSP sunt create sa suporte mai multe tipuri de documente structurate, indeosebi HTML si XML. În general, JSP-urile folosesc anumite informatii pe care le trimit la server intr-o cerere HTTP care interactioneaza cu datele existente pe acesta si creaza dinamic un raspuns organizat intrun format standard (HTML, DHTML, XML, etc.) sau intr-un format text sau neorganizat ce va fi trimis înapoi clientului. [2]

* 1. **Ce este SERVLET?**

Un servlet este un fișier java care poate solicita clientul, proceseaza-l și furnizează un fisier HTML ca raspuns. Exista mai multe servlete in interiorul containerului web. De asemenea, este posibila maparea mai multor cereri la un servlet. Prin urmare, toate aceste configuratii sunt incluse in acest fisier special, care este fisierul web.xml.[5]

* 1. **Care este diferenta majora intre JSP si SERVLET?**

Principala diferentaintre JSP și Servlet este caJSP este un limbaj de scripting al paginii web care poate genera continut web dinamic în timp ce servletul este un program Java care este deja compilat și folosit pentru a crea conținut dinamic de web.

Intr-o aplicatie web obisnuita, clientul solicita o pagina Web de la server și serverul raspunde inapoi cu pagina necesara. Aceste pagini pot fi fie statice, fie dinamice. Continutul unei pagini statice este deja creat. Continutul unei pagini dinamice este creat in timpul rularii. JSP și Servlet sunt două metode în Java pentru a crea pagini web dinamice. JSP este tradus si compilat intr-un servlet de catre containerul web. Pe de alta parte, un servlet este un program Java care este gestionat de containerul web.[5]

## 4) Limbajul HTML si utilitatea sa in aplicatii

**4.1. Ce este HTML?**

Unul din primele elemente fundamentale ale WWW ( World Wide Web ) este HTML (Hypertext Markup Language ), care descrie formatul primar în care documentele sunt distribuite și văzute pe Web. Multe din trasaturile lui, cum ar fi independenta fata de platforma, structurarea formatării și legaturile hypertext, ilustreaza utilizarea sa drept un foarte bun format pentru documentele Internet și Web.[4]

**4.2. In ce scop utilizam limbajul HTML?**

Scopul HTML este mai degraba prezentarea informatiilor – paragrafe, fonturi, tabele

ș.a.m.d. – decat descrierea semanticii documentului. In cadrul [dezvoltarii web de](https://ro.wikipedia.org/wiki/Dezvoltare_web) tip [front-end,](https://ro.wikipedia.org/wiki/Front-end_%C8%99i_back-end) HTML este utilizat impreuna cu [CSS](https://ro.wikipedia.org/wiki/CSS) [și JavaScript.[2](https://ro.wikipedia.org/wiki/JavaScript)]

## 5) Descrierea aplicatiei

### 5.1. Baza de date

Tema individuala se bazeaza pe crearea unei baze de date ce are 3 tabele in asociere M:1, 1:N Tabelele sunt farmacii, angajati si locatii:. Pentru cele 3 tabele am ales cateva atribute caracteristice:

Pentru tabela farmacii: am ales ca si cheie primara idFarmacie. Celelalte atribute sunt:numeFarmacie, telefon, email.

Pentru tabela locatii: am ales ca si cheie primara idLocatie. Celelalte atribute sunt: adresa, oras.

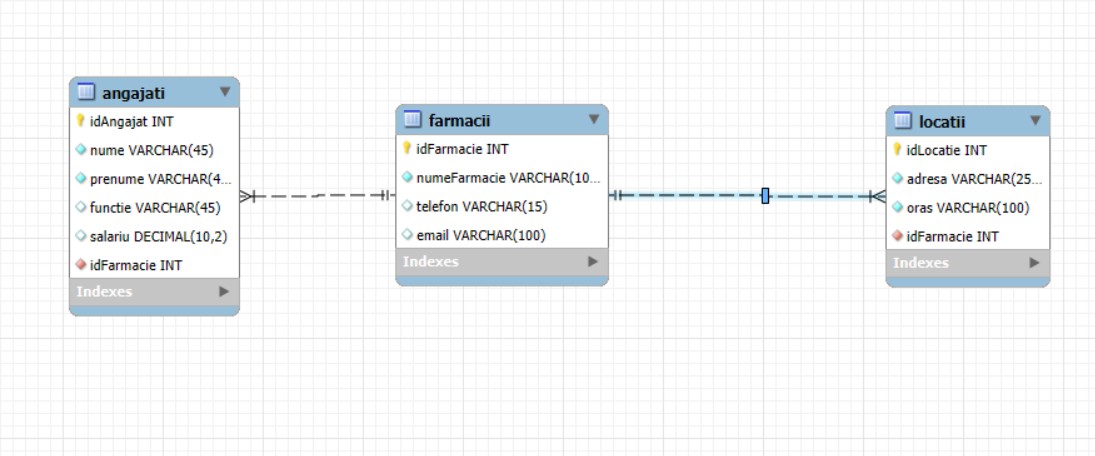
Pentru tabela angajati, am ales ca si cheie primada idAngajat. Celelalte attribute sunt: nume, prenume, functie si salariu **Ce este asocierea M:1?**

O asociere **M:1** (sau „mulți la unu”) este o relație între două tabele într-o bază de date, în care mai multe înregistrări din primul tabel (cel din stânga, „M”) pot fi asociate cu o singură înregistrare din al doilea tabel (cel din dreapta, „1”).

**Ce este asocierea 1:N?**

1:N reprezintă o relație de tip „unu la mulți” între două tabele într-o bază de date. Aceasta înseamnă că pentru fiecare înregistrare din primul tabel (cel din stânga, „1”), pot exista mai multe înregistrări asociate în al doilea tabel (cel din dreapta, „N”), dar pentru fiecare înregistrare din al doilea tabel există o singură înregistrare asociată în primul tabel.

### 5.2. Diagrama logica a bazei de date (Diagrama ERD)



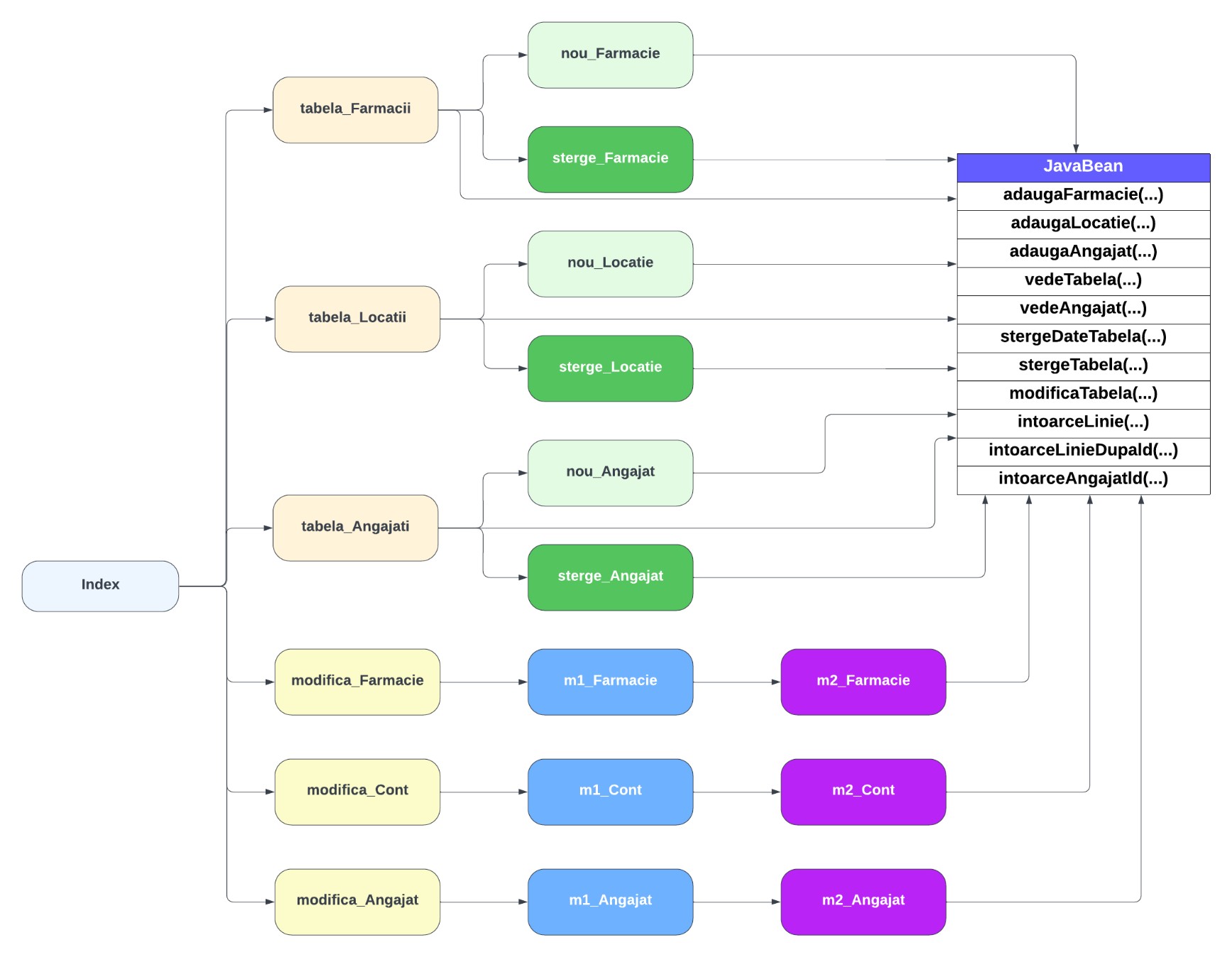
**De ce este importanta diagrama logica a bazei de date?**

Aceasta ofera o reprezentare logica detaliata a datelor celor 3 tabele, a relatiilor dintre ele.

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

* Intre **angajati** si **farmacii** exista o asociere de tip M:1
* Intre **farmacii** si **locatii** exista o asociere de tip 1:N

### 5.3. Diagrama implementarii fisierelor ce alcatuiesc pagina web



**5.4. Functionalitatea aplicatiei**

**5.4.1. Arhitectura proiectului**

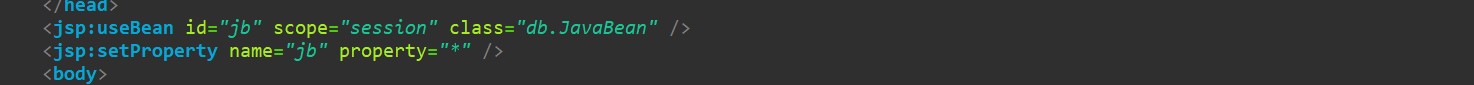
Proiectul realizat in tehnologia JSP are urmatoarea structura:

* Un pachet DB ce continue clasa JavaBean, ce are rolul de a oferi toate functionalitatile principale ale interfetei bazei de date si anume: conectica, operatiile de afisare, adaugare, modificare, stergere.
* Folder-ul webapp ce continue toate paginile JSP, ce au rolul de a importa functionalitatile din clasa JavaBean, fiind conectate intre ele si implicit conectate la pachetul DB, ce realizeaza partea dinamica a proiectului.

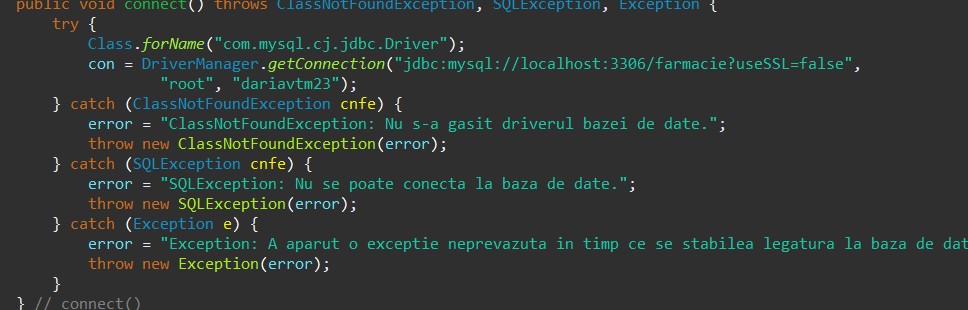
**5.4.2. Implementarea functiilor**

Toate paginile JSP contin structuri de cod java, ce ofera aplicatiei comportamentul dinamic. Ca sa utilizam functiile din clasa java, trebuie sa utilizam tag-urile JSP.

Un exemplu putem regasi in pagina JSP tabela\_Farmacii:



Jb este un obiect instanta la clasa JavaBean, astfel facandu-se legatura directa intre paginile JSP si functiile pe care dorim sa le implementam. .



Obiectul “con” este cel prin intermediul caruia se realizeaza toate operatiile pe baza de date. In functia connect() : se incarca driver-ul de MySQL. DriverManager contine mai multe drivere de conectica, mai multe SGBD-uri.

Prin functia getConnection(…) apelam functia de conectare si are ca atribute baza de date realizata in MySQL, user-ul si parola utilizatorului.

Totodata, exista si o functie disconnect() ce realizeaza close() pe obiectul de conexiune con.

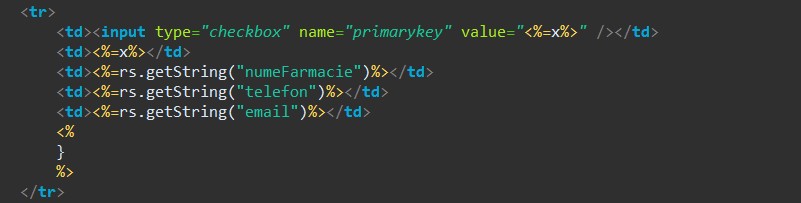
**Cum se afiseaza datele?**

In primul rand alegem ca si exemplu tabela farmacii. Functia din clasa JavaBean este vedeTabela() ce primeste ca si atribut numele tabelei pe care dorim sa facem interogarea si returneaza un set de rezultate: linii de tip ‘medici’, ce le implementeaza intr-un obiect de tip results. Acesta este declarat initial null.



Se creeaza instructiunea de interogare SQL: select \* from `farmacie`

Se realizeaza un statement pe conexiunea la baza de date, apoi se executa interogarea efectiva. Si astfel putem pune in evidenta partea de ‘’back-end’’ din spatele comenzii din tabela de clienti:





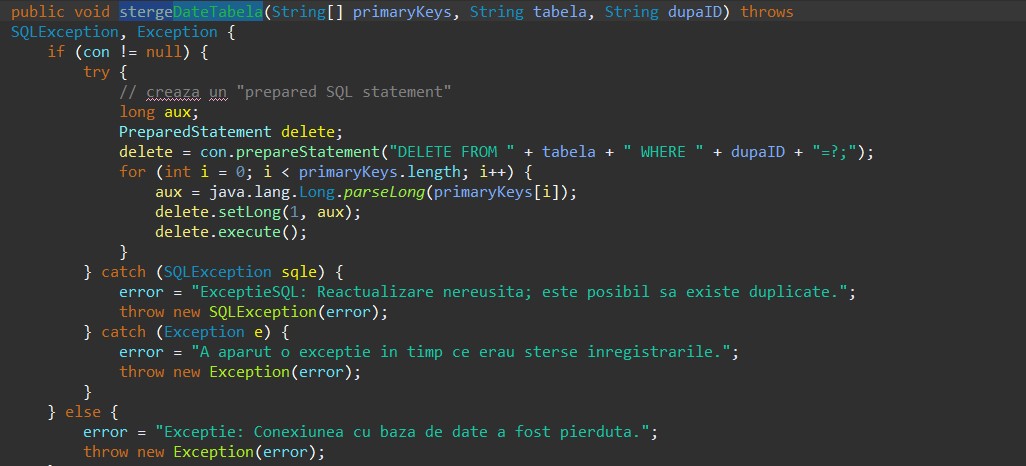
In spatele checkbox-urilor vom avea asociate valorile “idFarmacie” specific fiecarei farmacii inregistrate. S-a realizat afisarea linie cu linie. **Cum se realizeaza stergerea datelor?**

Luam drept exemplu stergerea datelor unui medic. Asadar ne indreptam atentia catre fisierul sterge\_Medic.jsp.

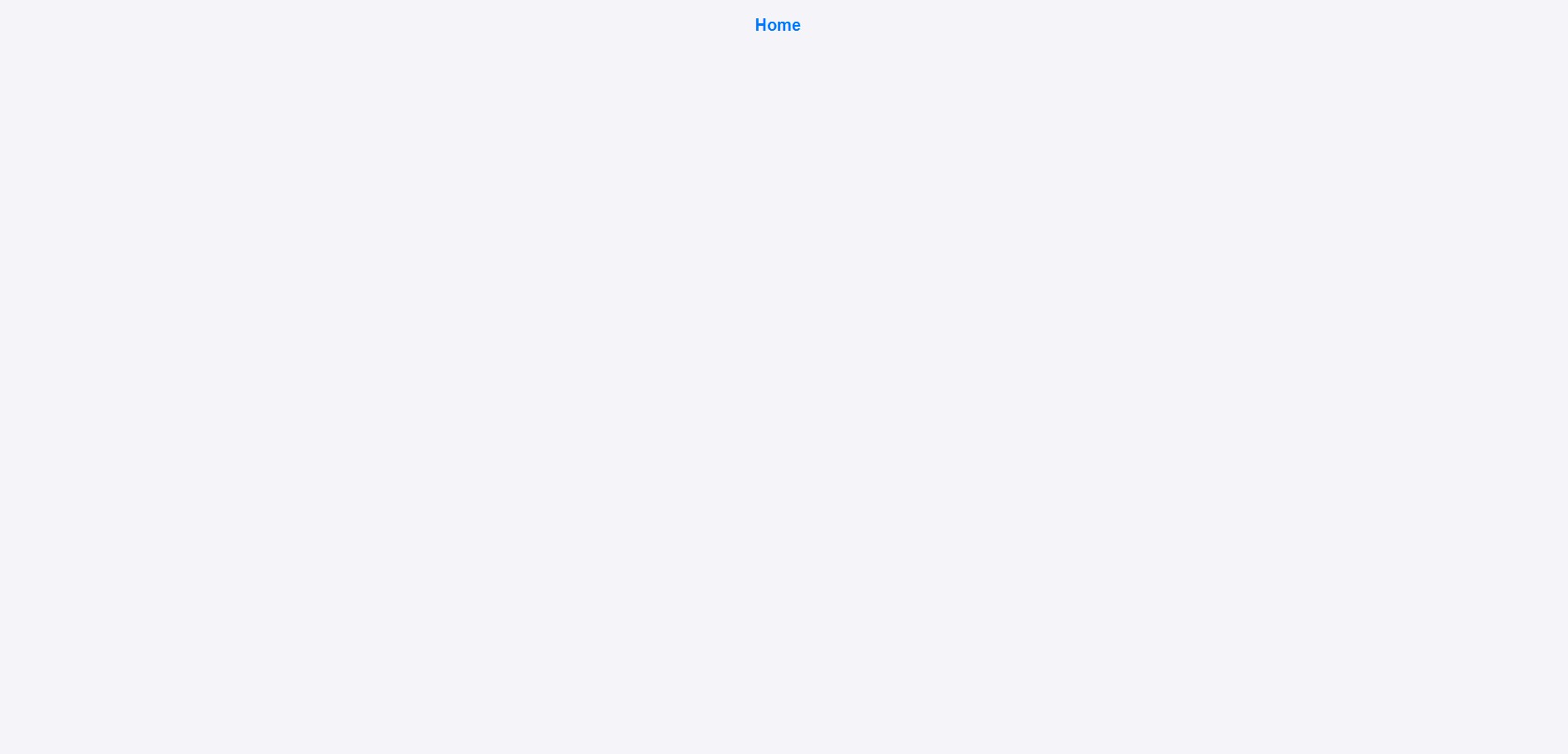


Daca alegem sa apasam butonul “Sterge liniile marcate” din cadrul paginii ce contine datele conturilor inregistrate, se va face legatura cu fisierul sterge\_Farmacii.jsp.

Se memoreaza toate datele pe care le-am bifat si se plaseaza in vectorul s, si prin obiectul jb se apeleaza la functia stergeDateTabela(…) din clasa JavaBean, functie ce are ca atribute tabela si totodata campul dupa care se doreste sa se faca stergerea.

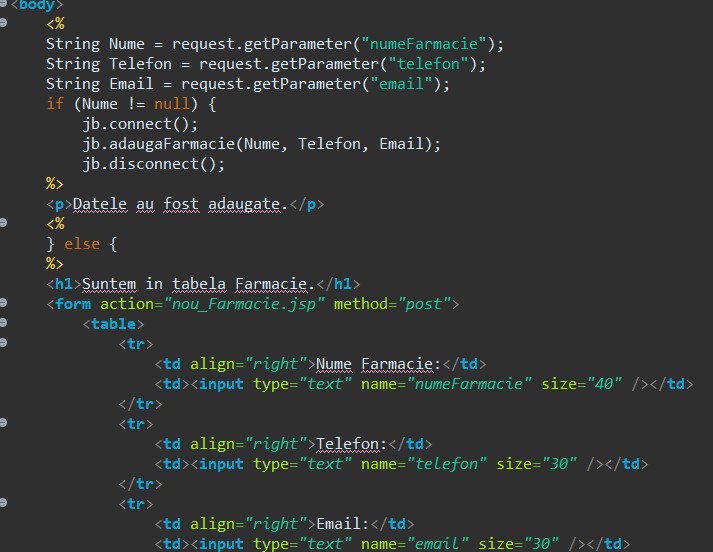


Prin instructiunea if (con != null) se verifica daca avem conexiune cu baza de date. Apoi se declara un obiect “delete” de tip PreparedStatement, declaratie ce se realizeaza pe conexiunea con. Se insereaza instructiunea specifica MySQL urmata de "=?;" ce pune in evidenta faptul ca noi nu cunoastem cate id-uri au fost selectate. Prima valoare ce se ascunde sub “?” se inlocuieste cu valoarea aux, dupa care se executa si apoi se sterge din baza de date intreaga linie. Acest ciclu se continua pana cand se sterg toate datele ce au fost selectate. La finalul operatiei, se va afisa urmatoarea pagina:



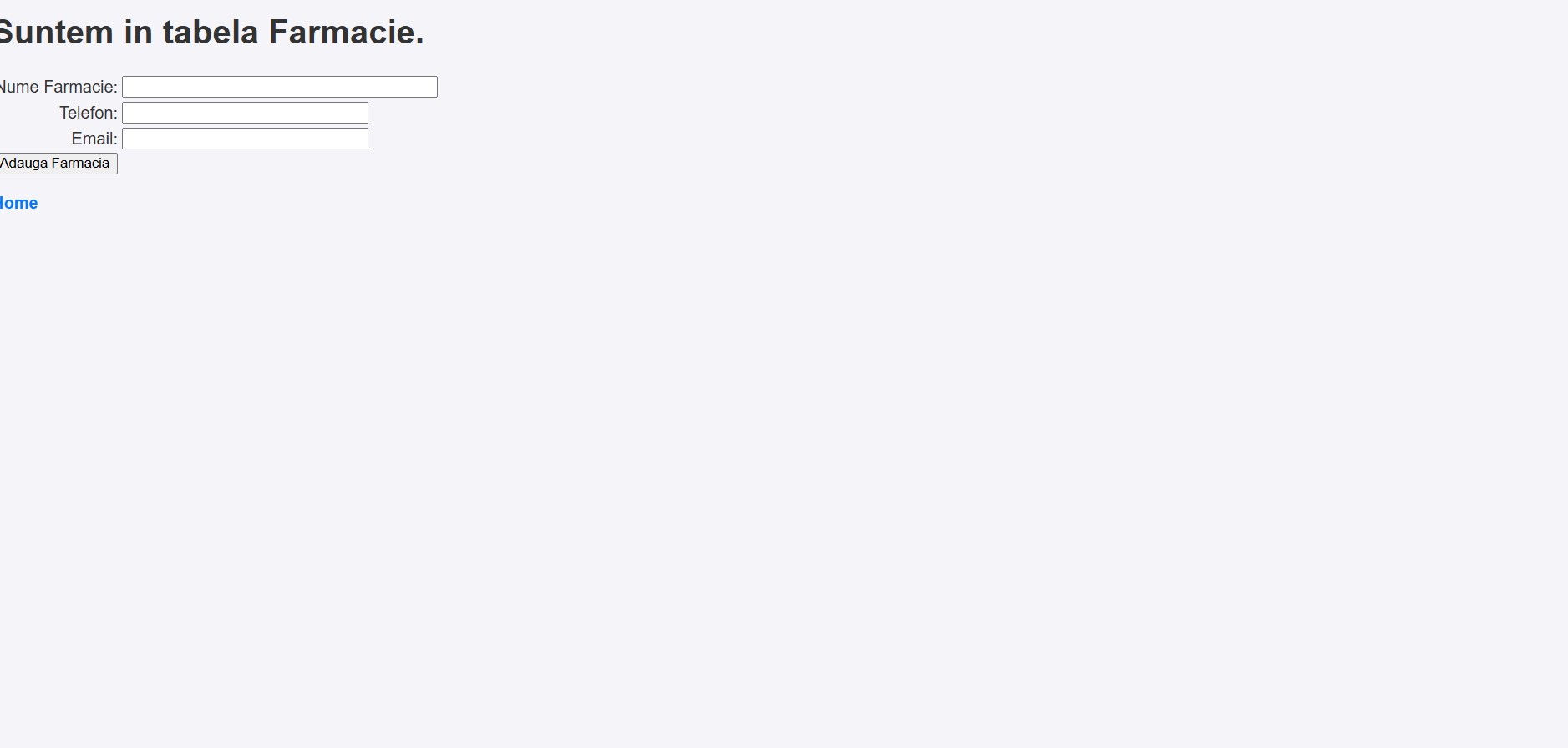
**Cum se realizeaza adaugarea datelor?**

Pentru aceasta operatie vom lua ca exemplu tabela farmacii. Deci ne concentram atentia asupra fisierului nou\_Farmacie.jsp.

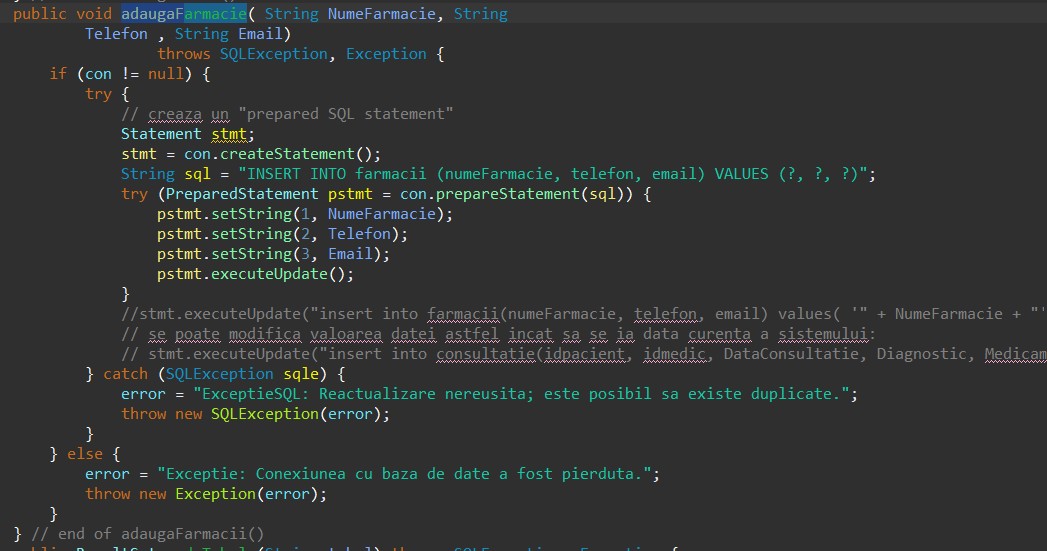


Se declara initial elementele pentru campurile pe care dorim sa le completam. Se preiau parametrii ce actioneaza ca si foreign-keys cu functiile de getParameter(…) si anume: idFarmacie.

Pagina in cadrul careia se adauga datele este:



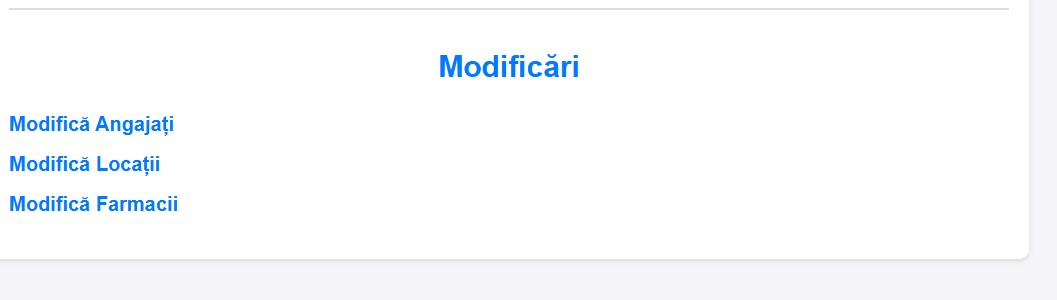
O data apasat butonul “Adauga Farmacia”, se va face legatura cu functia adaugaFarmacie(…) din clasa JavaBean:



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

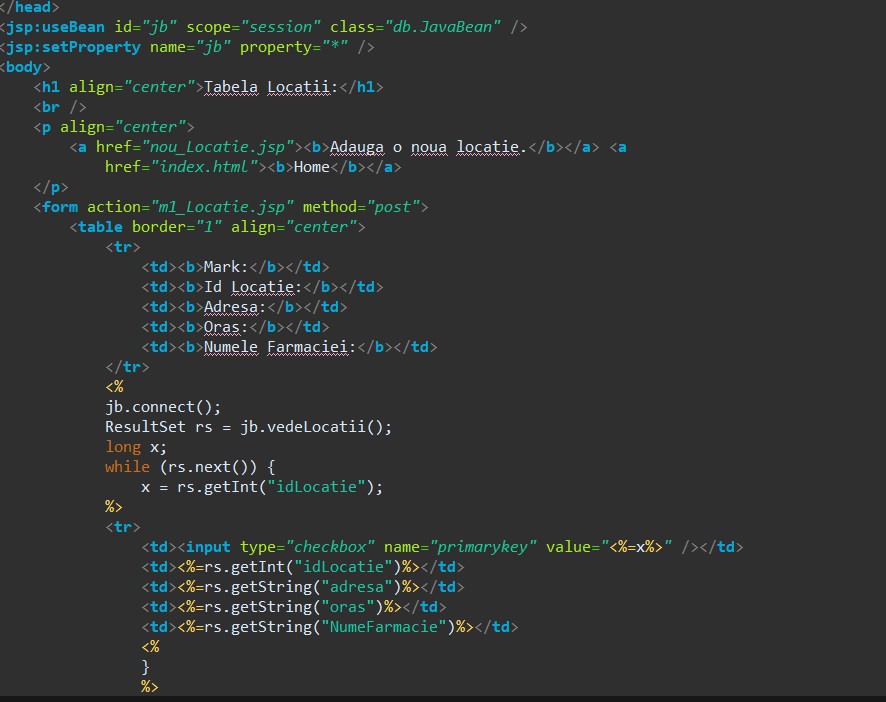
**Cum se realizeaza modificarea datelor?**

Am ales ca pe interfata site-ului , operatia de modificare sa fie separata fata de celelalte. Asadar vom lua exemplu tabela de locatii, deci ne vom indrepta atentia catre index, unde am introdus butonul pentru operatia de modificare:



Apasand butonul vom fi redirectionati catre fisierul JSP modifica\_Locatie.

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



Aici se implementeaza metoda intoarceLinieDupaId(…) din clasa JavaBean. Aceasta este utilizata pentru a selecta datele ce sunt in legatura cu id-ul contului pe care dorim sa il modificam.

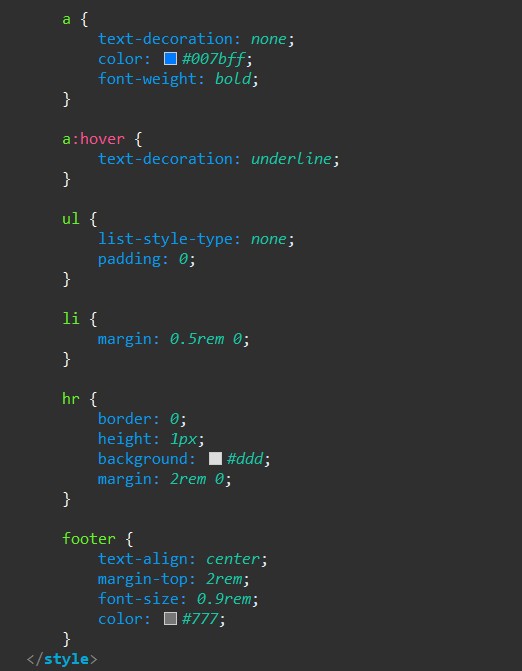
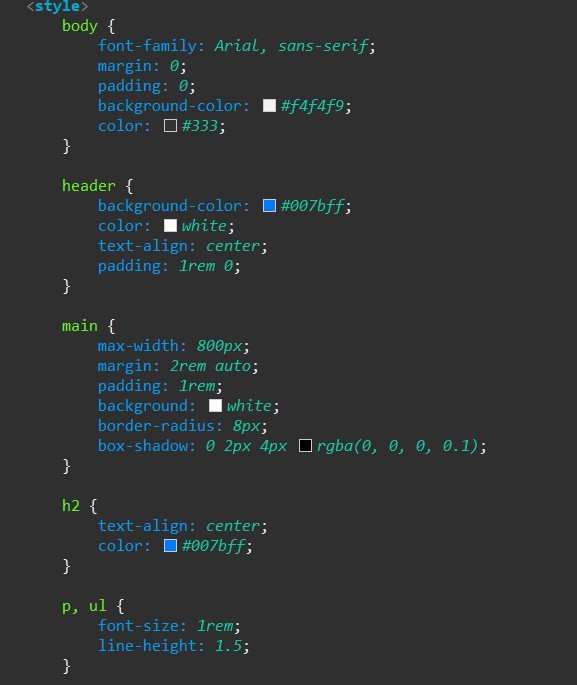
Utilizam din nou instructiunea specifica MySQL “SELECT \* FROM …. “.

In fisierul m1\_Locatii, am creat campuri de tip INPUT pentru introducerea noilor date. Acestea au fost create in cadrul formularului ce este legat de fisierul m2\_Locatii.jsp, unde se realizeaza modificarea propriu-zisa.

**5.4.3. Partea de front-end**

Pentru un aspect mai placut al interfetei, am folosit limbajul HTML combinat cu CSS.

De exemplu am ales sa creez un aspect mai placut in ceea ce priveste butoanele, divs si background folosind instructiuni in blocul STYLE:



Principalul rol al limbajului HTML folosit in cadrul acestei aplicatii este de a plasa componentele specifice design-ului paginii web la anumite coordonate si totodata pentru a seta dimensiunile corespunzatoare pentru div-uri ce compun pagina web.

## 6) Concluzii

In prezent bazele de date sunt utilizate practic peste tot, fiecare companie, intreprindere mai ales cu caracter de producere, comercializare are nevoie si la sigur implimenteaza in sistemul lor o baza de date. Pentru o utilizare fiabila și corecta a unei baze de date este important sa se urmareasca realizarea unei arhitecturi ce ofera posibilitatea de a separa functionalitatile, si anume partea de back-end de partea de front-end.

Totodata, aplicatia realizata in tehnologia JSP si prezentata in cadrul acestui proiect, numita “Banca-Management Platform”, permite utilizatorului sa efectueze operatiile cerute pe baza de date creata in MySQL: afisare, adaugare, modificare si stergere.

## 7) Bibliografie

* [https://www.nav.ro/blog/ce-este-mysql/ [](https://www.nav.ro/blog/ce-este-mysql/)1]
* [https://ro.wikipedia.org/ [](https://ro.wikipedia.org/)2]
* Cursurile PIBD [3]
* [https://web.ceiti.md/lesson.php?id=1 [](https://web.ceiti.md/lesson.php?id=1)4]
* <https://ro.sawakinome.com/>[5]