

**Academia de Studii Economice**

**Facultatea de Cibernetică, Statistică și Informatică Economică**

**Proiect**

**Java**

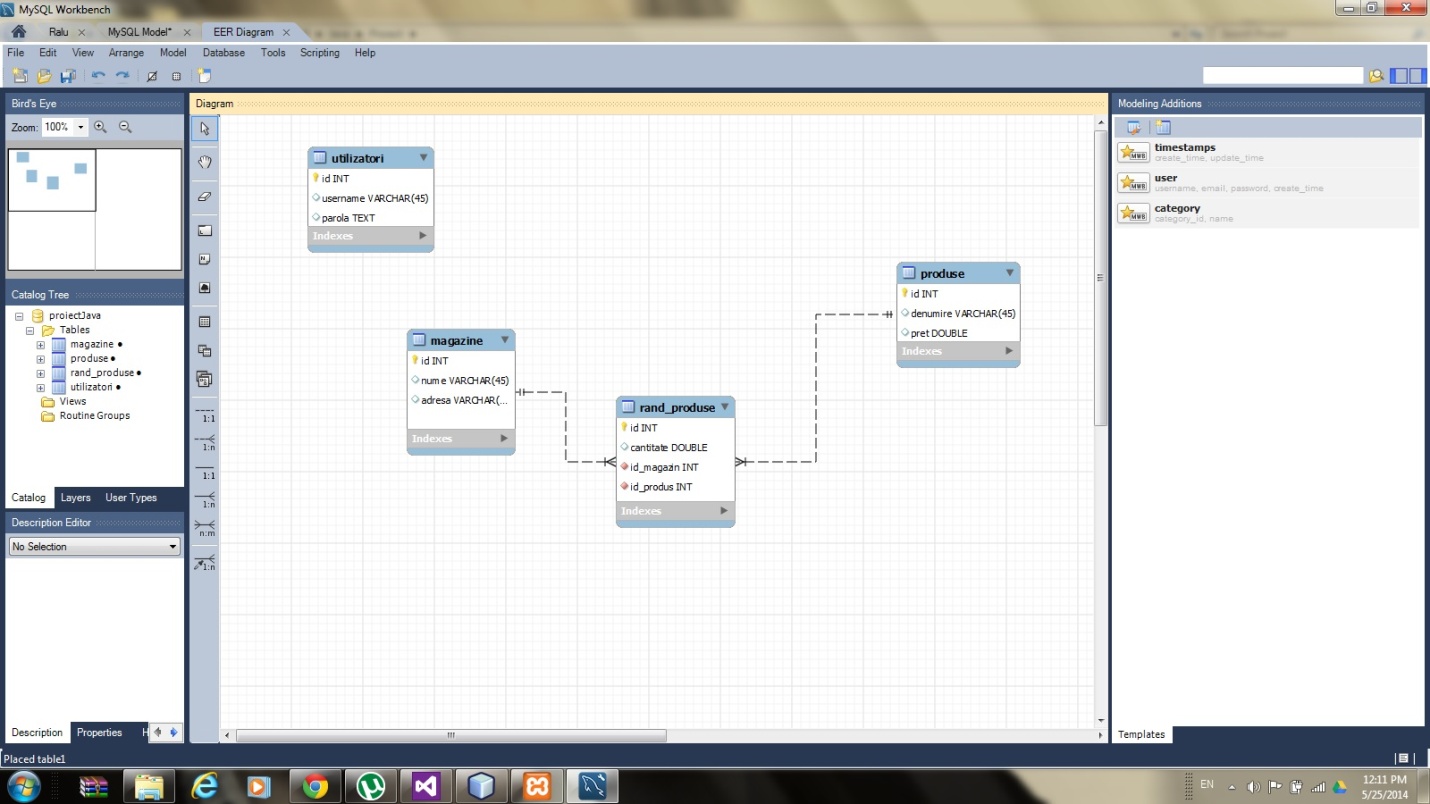
**Realizatori: Rusșindilar Raluca, Scripcaru Mihaela**

**Coordonator: Conf. univ. dr. Felix Titus Furtună**

**An II, Seria B, Grupa 1032**

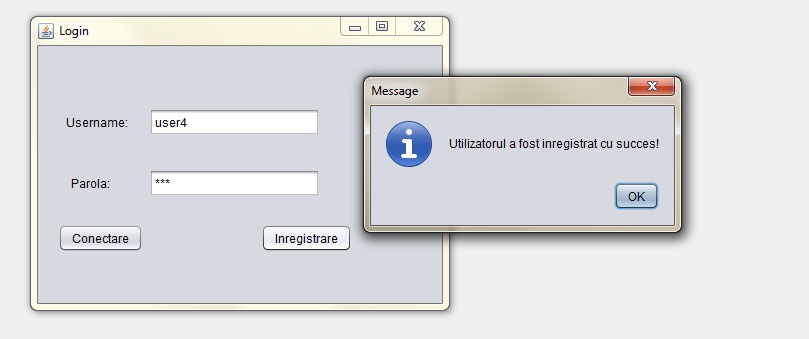
Proiectul ales de noi are drept scop facilitarea gestiunii comenzilor pentru o serie de magazine. Există posibilitatea conectării cu un anumit user (username și parola) sau înregistrării user-ului respectiv dacă acest lucru nu a fost deja făcut și gestionarea magazinelor și comenzilor de la magazinele existente.

Pentru gestionarea bazei de date am lucrat în MySQLWorkbench, iar pentru vizualizarea datelor existente pe măsura dezvoltării aplicației și pentru verificarea corectitudinii acestora am folosit aplicația PHP gratuită numită PHP myadmin ce se instalează o data cu instalarea aplicației XAMPP (aplicație ce instaleaza la rândul ei un server Apache, cât și unul MySQL). Schema bazei noastre de date este următoarea:

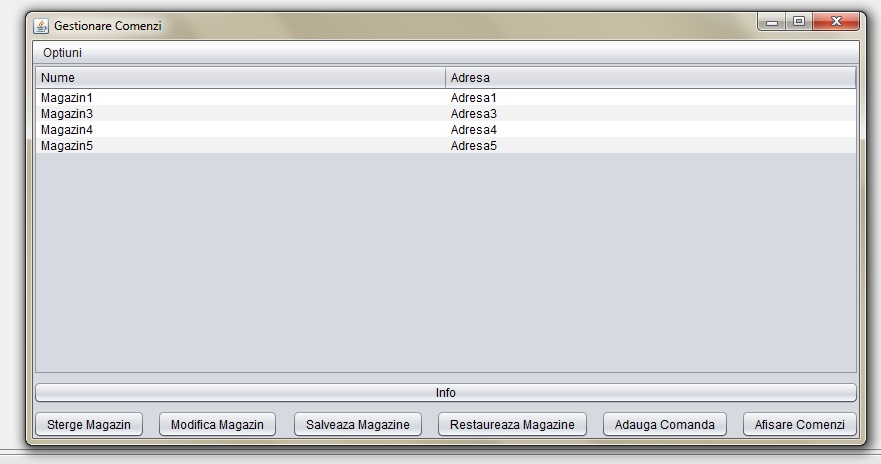


Aplicația noastră implementează următoarele facilități:

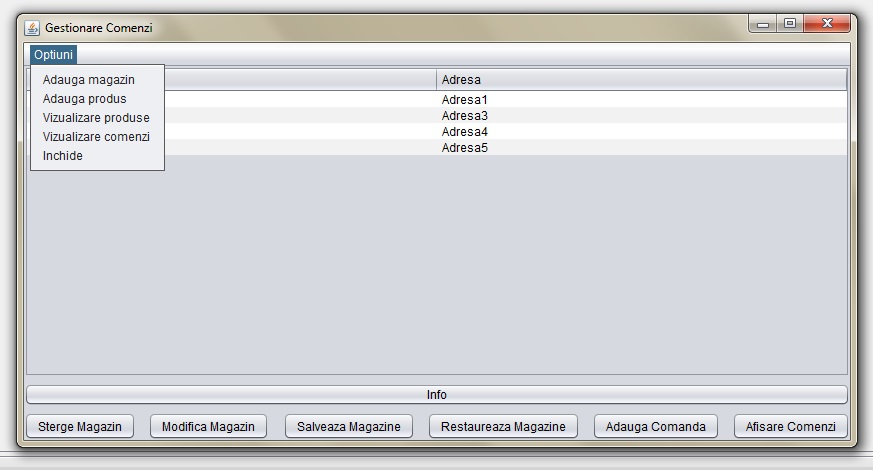
1. Conectarea sau înregistrarea unui user.

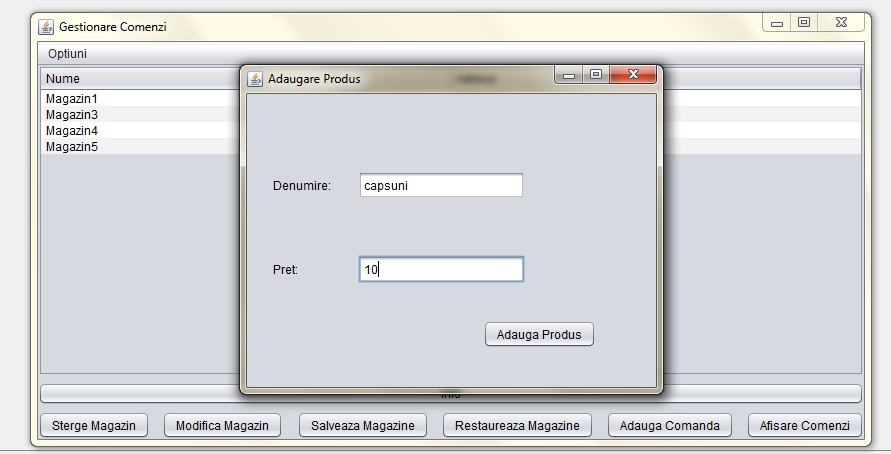


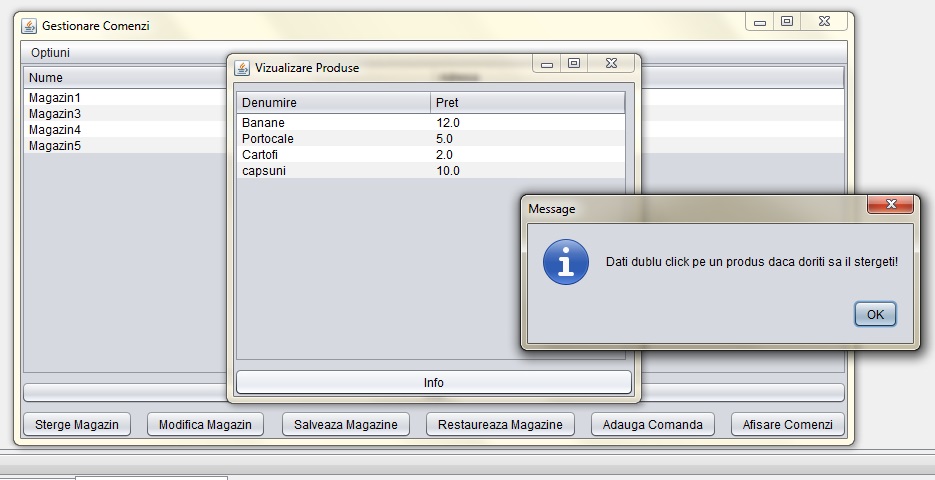
1. După conectare se deschide fereastra principală a aplicației în care se află un jTable ce gestionează toate magazinele pentru userul logat.

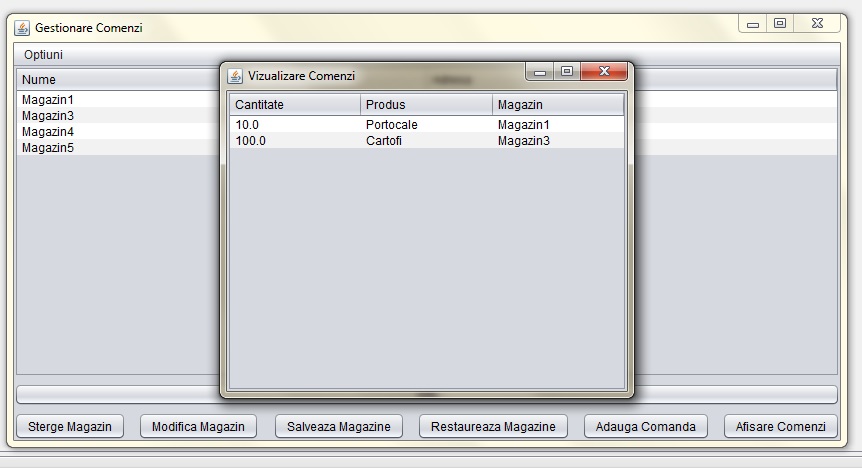


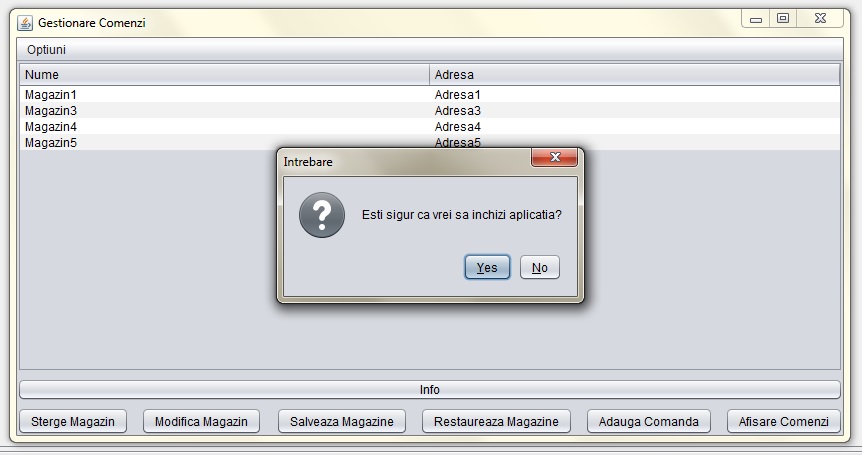
1. Meniul ferestrei principale permite adăugarea unui magazin nou, adăugarea unui produs nou, vizualizarea tuturor produselor adăugate, vizualizarea tuturor comenzilor înregistrate, închiderea aplicației.



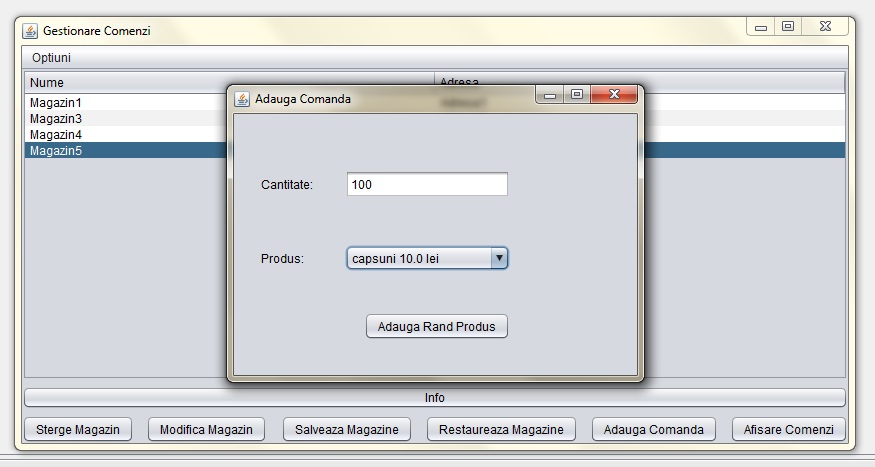


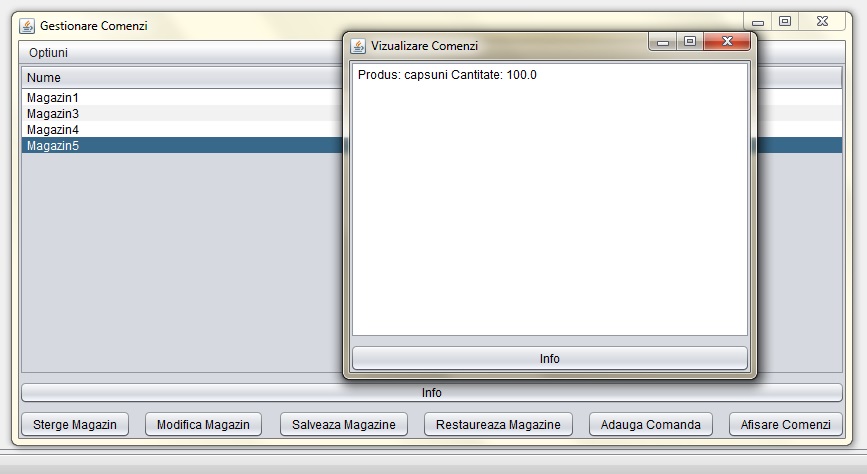






1. Butoanele de pe fereastra principală a aplicației au următoarele roluri: șterg un magazin selectat, modifică magazinul selectat, salvează toate magazinele într-un fișier, restaurează magazinele din fișier, adaugă o comandă la un magazin selectat, afișează comenzile unui magazin selectat.





Aplicația noastră este structurată conform design pattern-ului **Model View Controller**.

Pachetul **beans** reprezintă de fapt modelul și conține clasele care fac legătura cu baza de date: **Magazin**, **Produs**, **RandProdus**, **Utilizator**. Toate acestea implementează interfața Serializable pentru a putea fi trimise prin flux de către client serverului. Clasa **Mesaj**, care face parte tot din pachetul beans, îi spune serverului ce operație va face cu informația primită. Mesajele pot fi folosite atât pentru REQUEST cât și pentru RESPONSE. Deci clientul va comunica cu serverul prin intermediul mesajelor și ambele (atât clientul cât și serverul) sunt de tip TCP. De asemenea, am utilizat multithreading pentru a facilita conectarea mai multor clienți simultan la același server.

Pachetul **client.controllers** conține clasa **ClientController** care implementează toate metodele specifice clientului, metode în care sunt trimise/primite mesaje sau informații utile către/de la server pentru facilitarea lucrului cu baze de date si prelucrarea informațiilor.

Pachetul **server.controllers** conține câte un controller specific fiecărei clase (**Produs**, **Magazin**, **RandProdus**, **Utilizator**) care implementează metode ce lucrează efectiv cu interogările asupra bazei de date, precum și **ServerController**, un main controller care apeleaza controllerele claselor în cauză și în care este realizată conexiunea la baza de date **proiectjava.**

Clasa **ServerThread** citește mesajele primite de la client și în funcție de request-ul acestuia apelează metodele specifice din clasa **ServerController** după care îi răspunde clientului în funcție de cerințele sale.

De exemplu, în fereastra în care voi avea nevoie de o anumită funcționalitate se va apela o anumită metodă a ClientController-ului. Acea metodă va trimite serverului mesajul împreună cu alte date dacă este cazul și va primi un răspuns de la acesta. Serverul va primi mesajul și va testa în clasa ServerThread ce fel de mesaj este după care va apela funcționalitatea cerută din ServerController. ServerController-ul va apela la rândul său metoda dorită din controllerul clasei în cauză(unde se vor face interogările asupra bazei de date). După ce va obține datele necesare serverul îi va răspunde clientului. De asemenea, clasele **ClientController** și **ServerController**, sunt implementate ca și clase singleton pentru a putea fi instanțiate doar o singură dată.

În pachetul **tableModels** se află modelele tabelelor împeună cu metodele ce trebuie suprascrise.

Pachetul **client.gui** reprezintă interfața grafică și conține toate ferestrele ce vor fi deschise atunci când vor avea loc anumite evenimente de apăsare pe buton sau pe MenuItem.