UNIVERSITATEA TEHNICĂ „Gheorghe Asachi” din IAȘI
FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DOMENIUL: Calculatoare și tehnologia informației

Market Database

Proiect la disciplina Baze de date

Studenti:
Robert-Cosmin Roșu 1309A
Adrian Burbulea 1309A
Coordonator: Matei-Victor Tizu

Iași, 2024

Cuprins:

Capitolul 1. Introducere

In cadrul acestui proiect am dezvoltat un program in Java cu ajutorul caruia se gestioneaza baza de date a unui magazin alimentar.

Prin aceasta aplicatie am implementat vizual un sistem de gestiune al produselor unui magazin alimentar. Programul poate fi folosit atat de catre client, cat si de administrator.

Administratorului ii stau la dispozitie actiuni precum impartirea produselor pe categorii elementare, modificarea stocului acestora, inserarea atat produse noi, cat si tipuri de produse, vizualizarea vanzarilor, vizualizarea stocului si adaugarea in stoc, dupa necessitate.

Cu ajutorul aplicatiei, clientul vede produsele pe care magazinul le are in stoc, adauga produse in cos si cumpara.

Capitolul 2. Descrierea proiectului si aplicatii utilizate.

2.1 Resurse folosite:

Aplicatia prin care utilizatorul interactioneaza cu baza de date a fost creata in Java, cu sdk Oracle OpenJDK version 19.0.2, iar baza de date a fost create in SQLite 3.

Baza de data contine 4 table, respectiv:

- tipProduse
- produs
- cosCump
- vanzar3

2.2 Descrierea tabelelor:

Tabela *tipProduse* este tabela care contine detalii de baza despre un produs, precum campul tipId de tip intreg (primary key cu autoincrement), campul

description de tip varchar(25) si doua campuri de tip real care ajuta la calcularea pretului final al unui produs: adaosComercialPct si TVA.

Tabela *produs* contine un camp produsId (foreign key la tipId din tabela *tipProduse*), doua campuri de tip intreg tipProdus si stoc, doua campuri de tip real, respective pretDeBaza si pretFinal si un camp denumit nume de tip text care reprezinta numele produsului.

Campul pretFinal este calculat cu ajutorul unui trigger prin formula

pretFinal = pretDeBaza * (1 + adaosComercialPct + TVA).

Tabela *cosCump* contine 4 campuri: un camp produsID (foreign key la produsId din *produs*), un camp nrBucati de tip intreg (constrangere not null) si un real care reprezinta pretul toal denumit pretTotal.

Tabela *vanzare* este gandita sa stea in ajutorul administratorului, permitandu-i acestuia sa vizualizeze detalii precum valoarea totala a cumparaturilor unui client stocata in campul valoareTotala si numarul de produse cumparate de acesta nrProduse.

Capitolul 3. Functionalitatea aplicatiei

3.1 Front-end

La pornirea programului trebuie ales tipul de utilizator, anume Client sau Administrator.

Dupa alegerea tipului de utilizator, in cazul in care este ales tipul “Client”, utilizatorului i se va deschide o ferestra in care va vedea produsele pe care magazinul le are in stoc si din care isi va putea face cumparaturile.

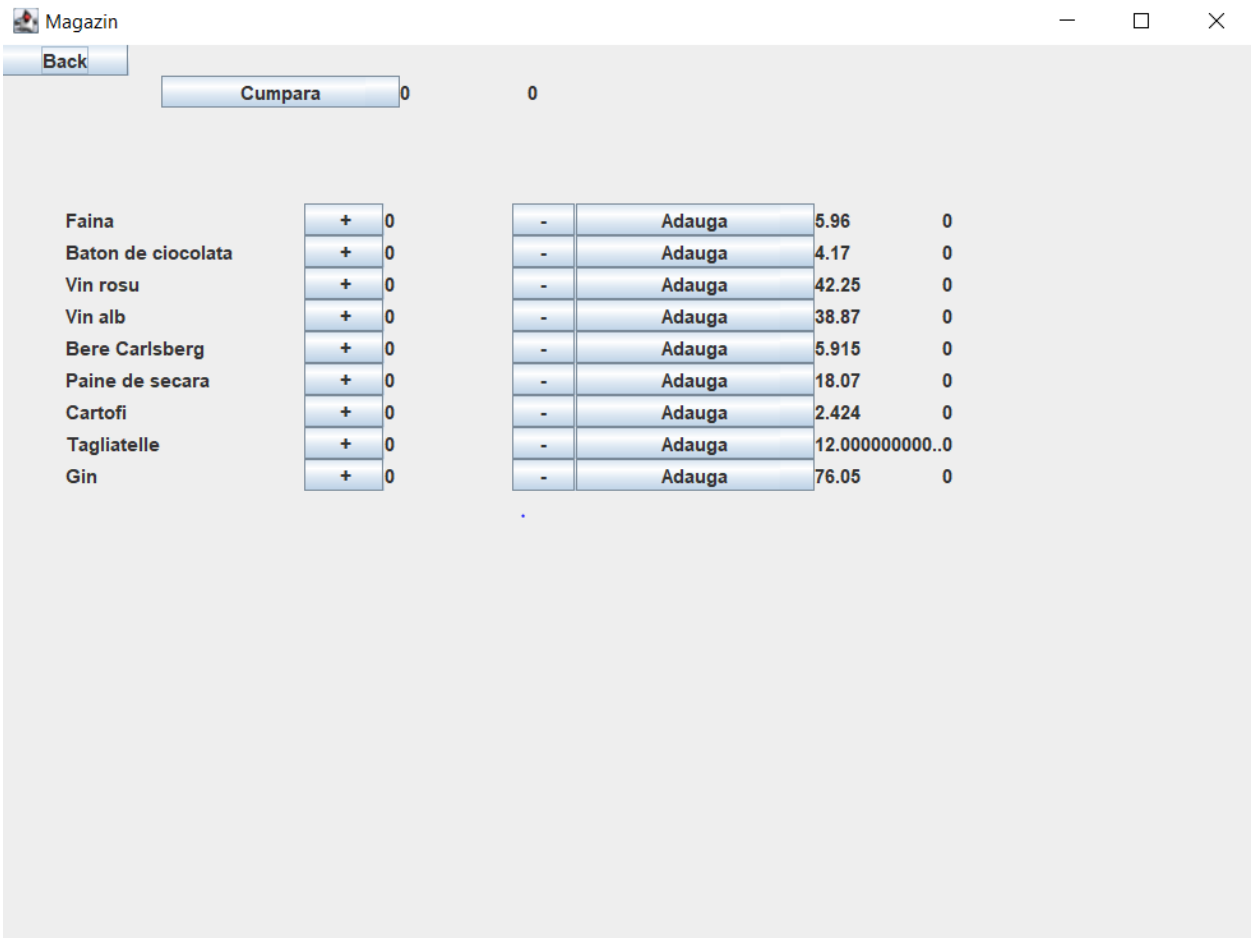


Fig. 3.1 Meniu Client

In cazul opus in care utilizatorul doreste proprietati de aministrator, se va deschide urmatoarea fereastra:

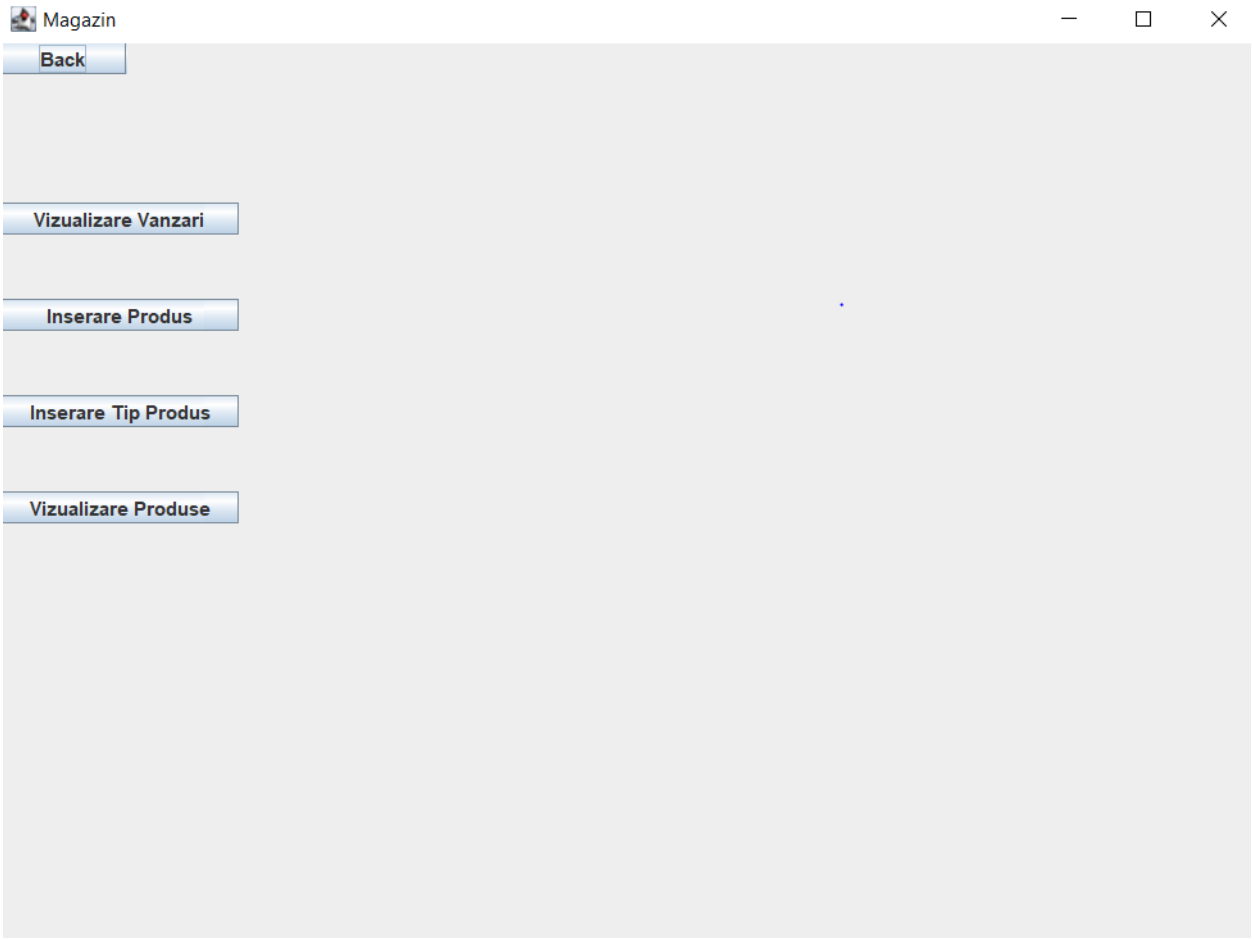


Fig 3.2 Meniu Administrator

Cu ajutorul butoanelor din Fig 3.2 administratorul gestioneaza marfa magazinului.

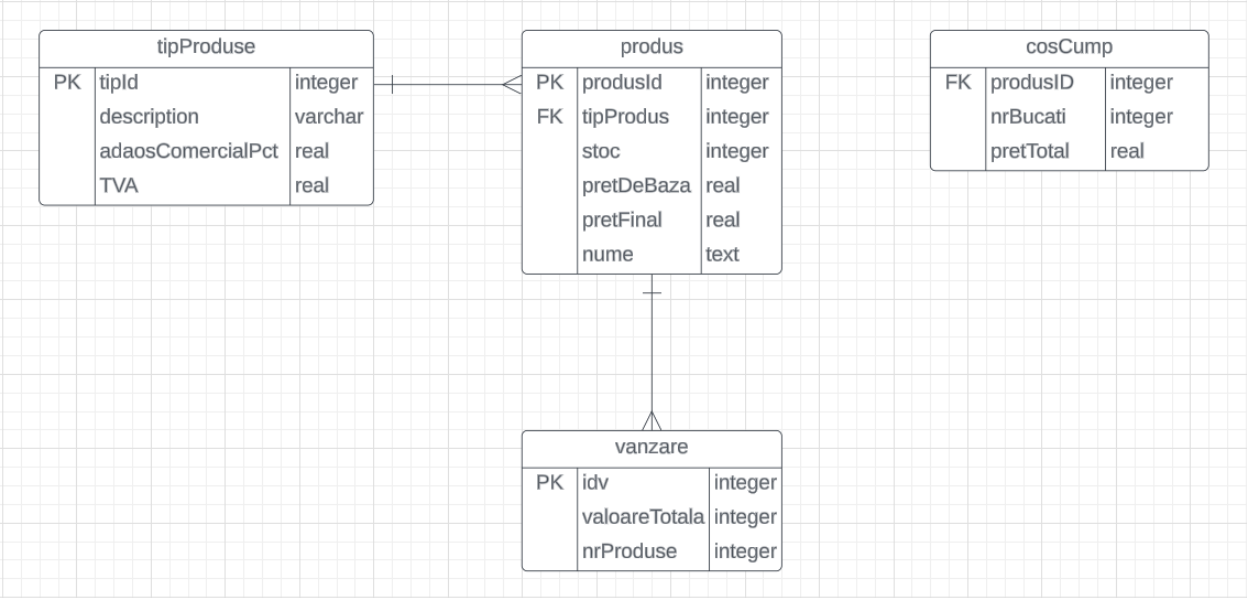
3.2 Back-end

Conexiunea interfetei grafice cu baza de date s-a realizat cu ajutorul driver-ului jdbc care este o componenta software care permite unei aplicatii Java sa interactioneze cu o baza de date.

Operatiile asupra tabelelor din baza de date sunt realizate cu ajutorul unor functii precum:

```
public void insertOrUpdateProdus (int tipProdus, int stoc, double pretDeBaza,
String nume)
{
    try {
        ResultSet resultSet;
        Statement statement = c.createStatement();

        String arg1 = Double.toString(tipProdus);
        String arg2 = Double.toString(stoc);
        String arg3 = Double.toString(pretDeBaza);

        statement.executeUpdate("DELETE FROM produs WHERE nume = '" + nume +
"';");
        statement.executeUpdate("INSERT INTO produs(tipProdus, stoc,
pretDeBaza, nume) VALUES(" + arg1 + ", " + arg2 + ", " + arg3 + ",
'"+nume+"');");
        System.out.println("Produs adaugat: "+nume+"");
    }
    catch (Exception E) { }
}

public void UpdateStoc(int Id,int stoc)
{
    try {
        ResultSet resultSet;
        Statement statement = c.createStatement();
        String arg1=Integer.toString(Id)+";";
        String arg2=Integer.toString(stoc);
        statement.executeUpdate("Update produs set stoc="+arg2+" where
produsId="+arg1);
    }
    catch (Exception E) {}
}

public void DeleteProdus(String nume)
{
    try{
        Statement statement = c.createStatement();
        statement.executeUpdate("DELETE FROM produs WHERE nume = '" + nume +
"';");
    }
    catch (Exception e){}
}
```

3.3 Diagrama ER



Impartirea task-urilor:

Adrian Burbulea:

- realizarea aplicatiei
- interactiunea cu baza de date

Robert-Cosmin Rosu:

- crearea bazei de date
- interactiunea cu baza de date
- documentatie