DOCUMENTATIE

TEMA 3

NUME STUDENT: Rujac Roxana
GRUPA: 30224

CUPRINS

1. Obiectivul temei

1.1 Obiectivul principal

Scopul proiectului este dezvolatarea unei aplicații de management al comenzilor pentru un depozit. Utilizatorul poate adăuga, șterge, edita și vizualiza clienți și produse, și poate crea comenzi. Datele sunt stocate într-o bază de date. Comenzile sunt filtrate pentru a verifica disponibilitatea produselor în depozit.

1.2 Obiectivele secundare

-Analiza detaliată a cerințelor funcționale și non-funcționale pentru a înțelege în profunzime scopul și funcționalitatea aplicației.

-Proiectarea unei arhitecturi eficiente și scalabile pentru aplicație, folosind principiile de proiectare orientată pe obiecte.

-Implementarea aplicației de gestionare a comenzilor depozitului.

-Crearea de metode generice.

-Testarea extensivă a aplicației pentru a verifica corectitudinea funcționării.

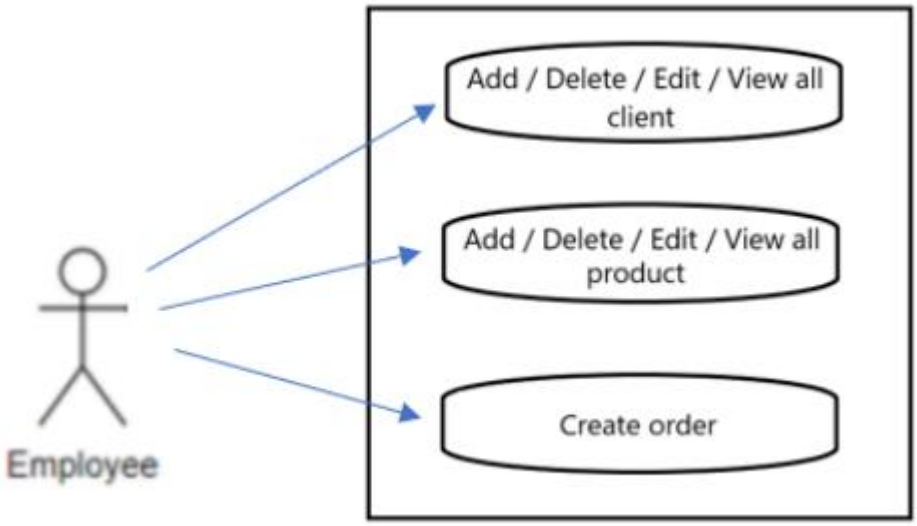
2.Analiza problemei, modelare, scenarii, cazuri de utilizare

2.1 Cerinte functionale

2.1.1 Clasificarea MoSCoW

- Must have: O conexiune la o bază de date care stochează toate informațiile relevante despre fiecare client, produs și comandă. Interfață grafică pentru utilizator
- Should have: Clase și metode generice care permit utilizarea acelorași metode pe diferite tipuri de obiecte
- Could have:
- Won’t have: Pagină de autentificare pentru client/angajat

2.1.2 Diagrama use-case



2.1.3 Scenarii de utilizare

Utilizatorul poate alege între operațiile legate de clienți sau produse sau poate crea o comandă.

• Pentru primele două opțiuni, utilizatorul poate: adăuga un client / produs nou, șterge aceste informații din baza de date, să le editeze sau să le actualizeze. Atunci când este selectată opțiunea „View”, apare o tabelă cu toate înregistrările.

• Pentru operația de creare a unei comenzi, cu ajutorul a două JComboBox, una pentru clienți și cealaltă pentru produse, angajatul va putea să aleagă o opțiune (un client / produs din listă). Un alt câmp trebuie completat în care utilizatorul este întrebat câte unități dorește să achiziționeze.

2.2 Cerinte non-functionale

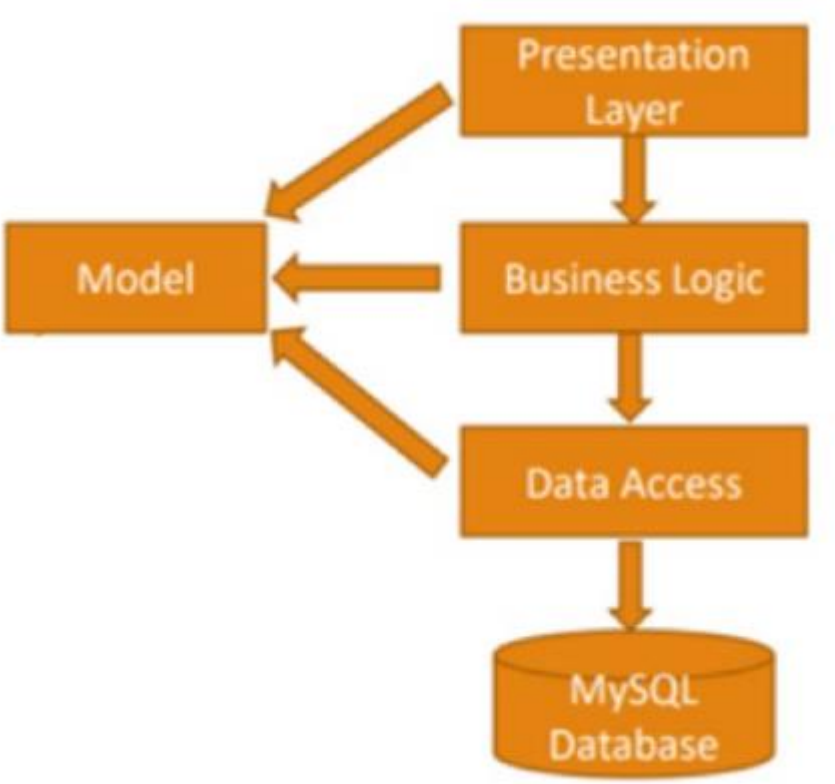
- Interfața trebuie să fie ușor de utilizat, cu instrucțiuni clare pentru utilizatori.
- Designul ar trebui să fie compact, fără elemente suplimentare, dar suficient de spațios pentru a afișa toate informațiile necesare.
- Timpul de răspuns al aplicației pentru fiecare acțiune nu ar trebui să depășească 2 secunde și ar trebui să fie rapid pentru majoritatea operațiunilor.
- Baza de date folosită trebuie să fie fiabilă, pentru a asigura securitatea și integritatea datelor.

3.Proiectare

Codul este scris în stilul OOP, urmând cele 4 piloni ai paradigmelor de Programare Orientată pe Obiecte: Încapsulare, Abstractizare, Polimorfism, Moștenire.

4.Design si implementare

4.1 Diagrama pachetelor



Arhitectura stratificată este un cadru comun și larg utilizat în dezvoltarea software-ului. Este structurat în mai multe straturi orizontale, fiecare cu rolul său bine definit și independență față de celelalte. În această arhitectură, avem patru straturi principale:

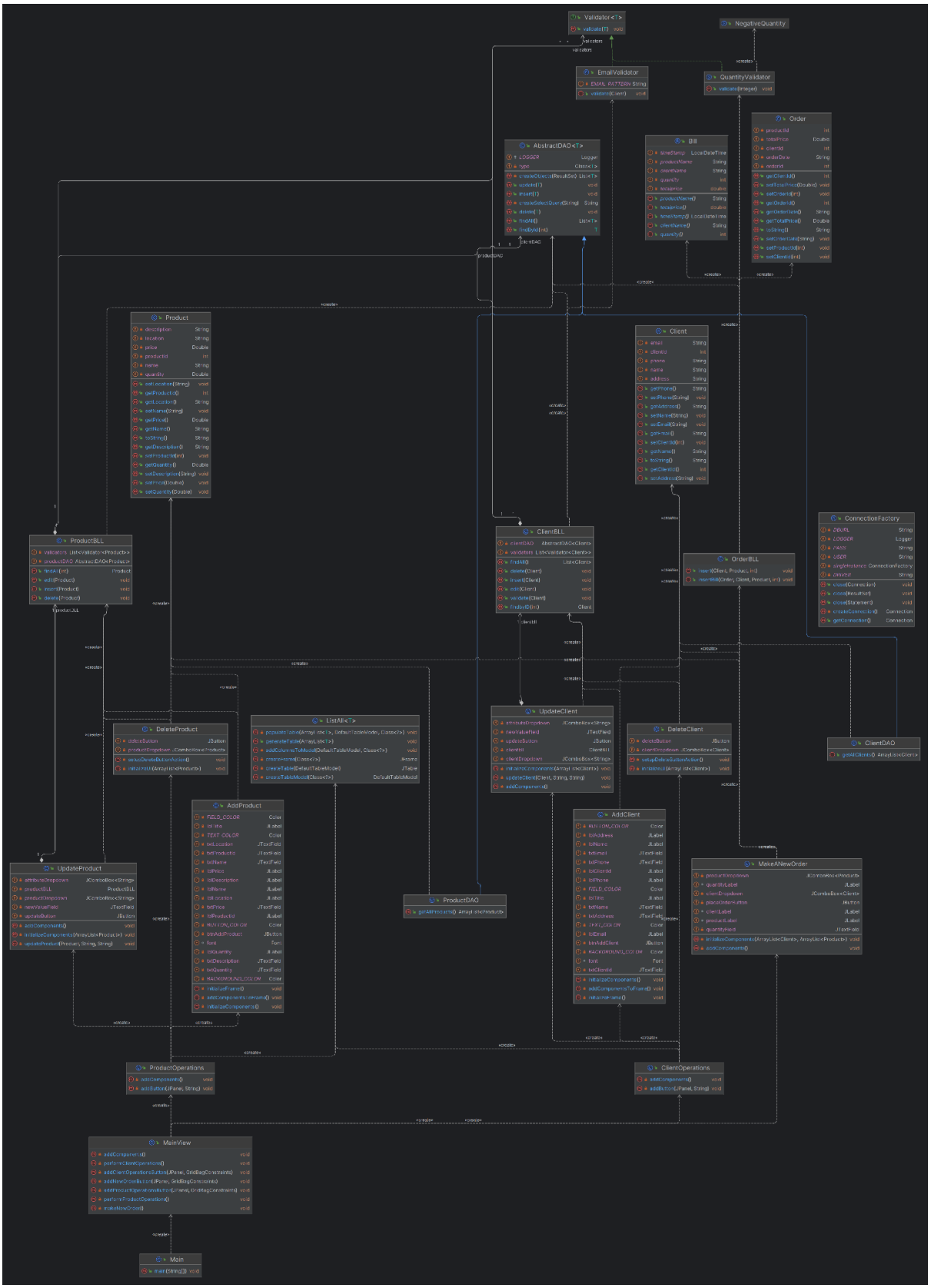
• Presentation- reprezintă interfața utilizatorului și interacțiunea cu acesta. Acestea se ocupă de afișarea informațiilor către utilizator și de colectarea datelor introduse de acesta.

• BussinessLogic- reprezintă nucleul aplicației, gestionând regulile și calculele specifice domeniului de activitate. Aici se efectuează validările datelor și se decide asupra acțiunilor viitoare în funcție de intrările utilizatorului. De asemenea, acest strat comunică cu baza de date pentru a accesa și manipula datele.

• DataAccess- furnizează metode pentru interogarea și manipularea datelor din baza de date. Implementează operațiile CRUD și asigură o conexiune cu baza de date.

• Model - reprezintă structurile de date și modelele de obiecte folosite în aplicație. Aici sunt definite clasele pentru entitățile principale, cum ar fi clienții, produsele și comenzile.

4.2 Diagrama claselor



4.3 Implementarea claselor

4.3.1 Model

- Client - stochează informații despre clienți, inclusiv ID-ul,, nume, adresă, telefon si email. Oferă metode pentru gestionarea acestor detalii și pentru a obține o reprezentare sub formă de șir a unui client.
- Product - reprezintă produsele din sistem, inclusiv detalii precum ID-ul unic, nume, preț, descriere, locatie și cantitate disponibilă. Aceasta permite accesul și modificarea informațiilor despre produse și oferă o modalitate de afișare a detaliilor sub formă de șir.
- Order - gestionează comenzile clienților, inclusiv detalii cum ar fi ID-ull comenzii, ID-ul clientului, ID-ul produsului și cantitatea. Ea oferă metode pentru accesarea și manipularea detaliilor comenzilor.
- Bill - reprezintă facturile generate pentru achizițiile efectuate. Ea stochează informații despre comandă, client, produs și sumă totală

4.3.2 Presentation

Toate clasele din pachetul presentation sunt interfete grafice ce ajuta utilizatorul sa interactioneze in mod usor si intuitiv cu aplicatia. El poate atat sa introduca date, sa vada toti clientii/produsele si sa dea o comanda.

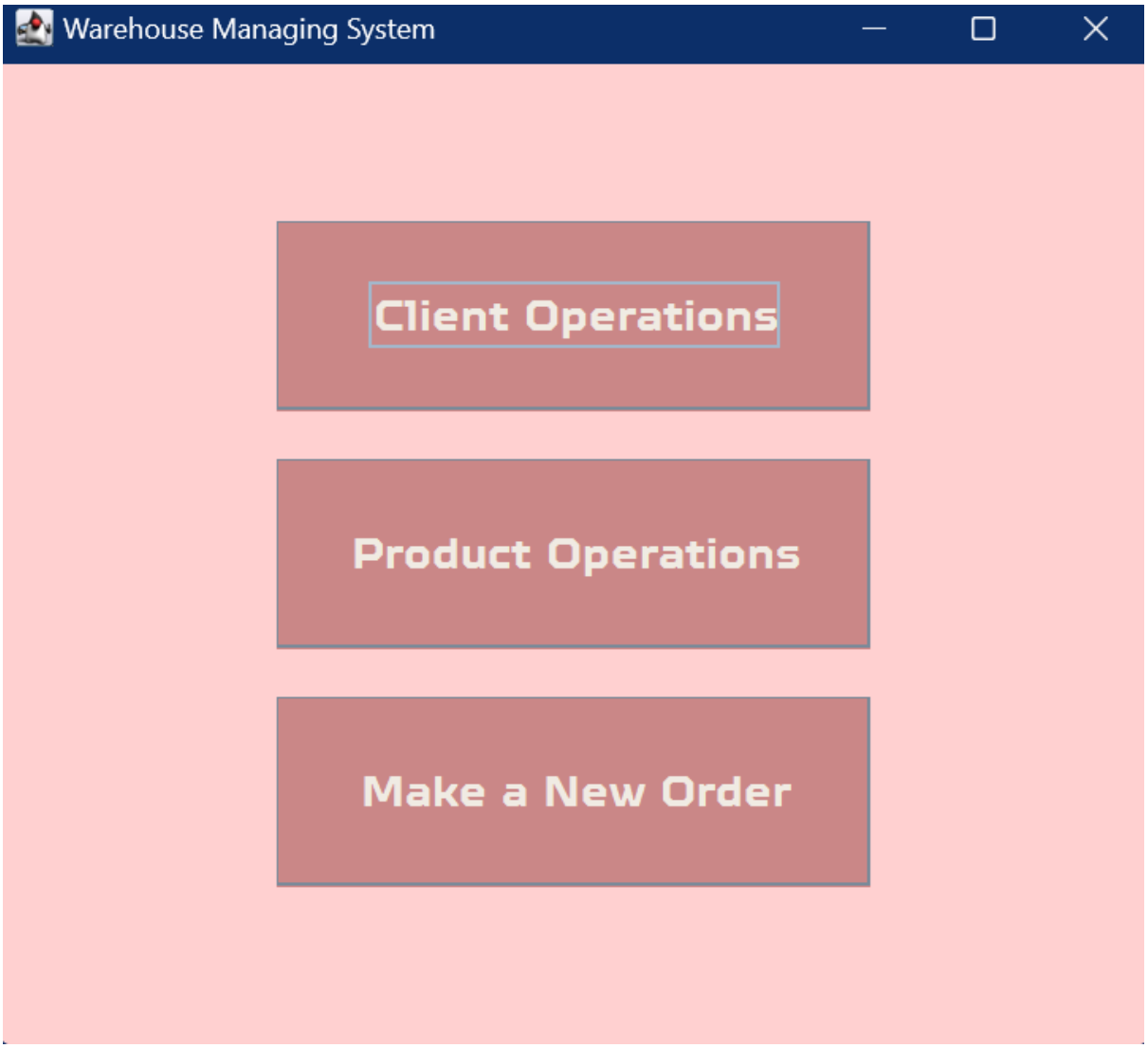
4.3.3 Bll

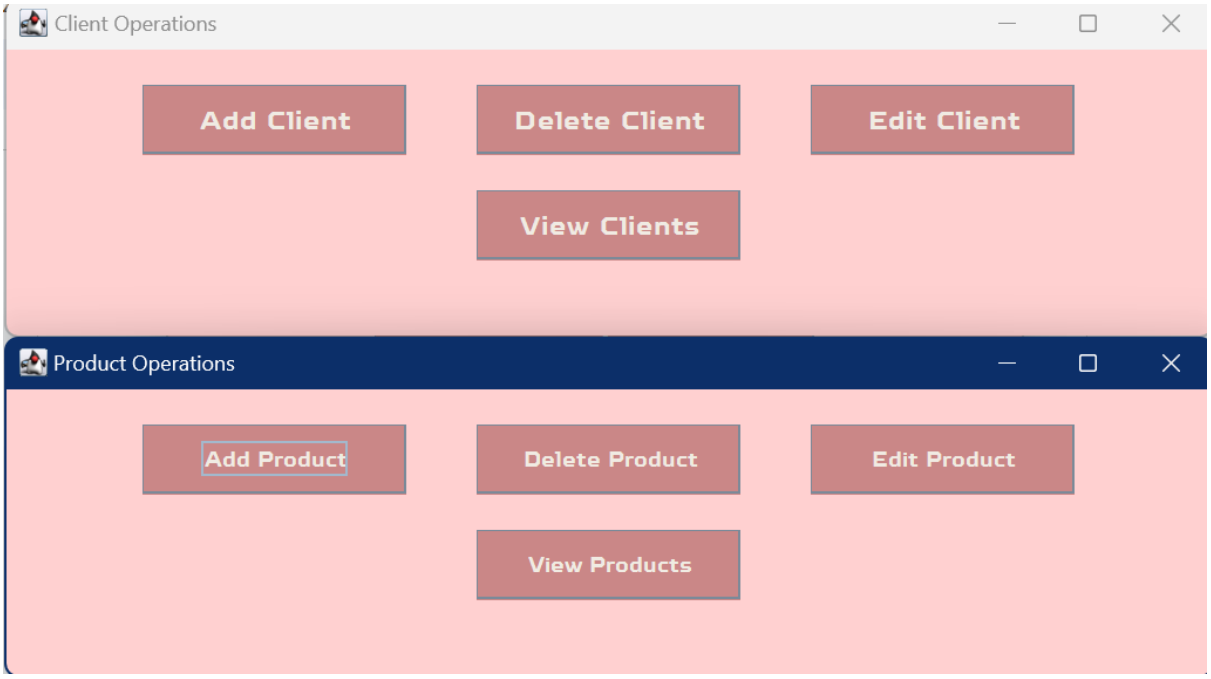
- ClientBLL - gestionează operațiile pe clienți în aplicație, intermediind între stratul de prezentare și cel de acces la date.
- OrderBLL - se ocupă de operațiile legate de comenzile clienților, inclusiv crearea de facturi și înregistrarea acestora în baza de date.
- ProductBLL - gestionează operațiile legate de produse, inclusiv inserarea, actualizarea și ștergerea acestora din baza de date.

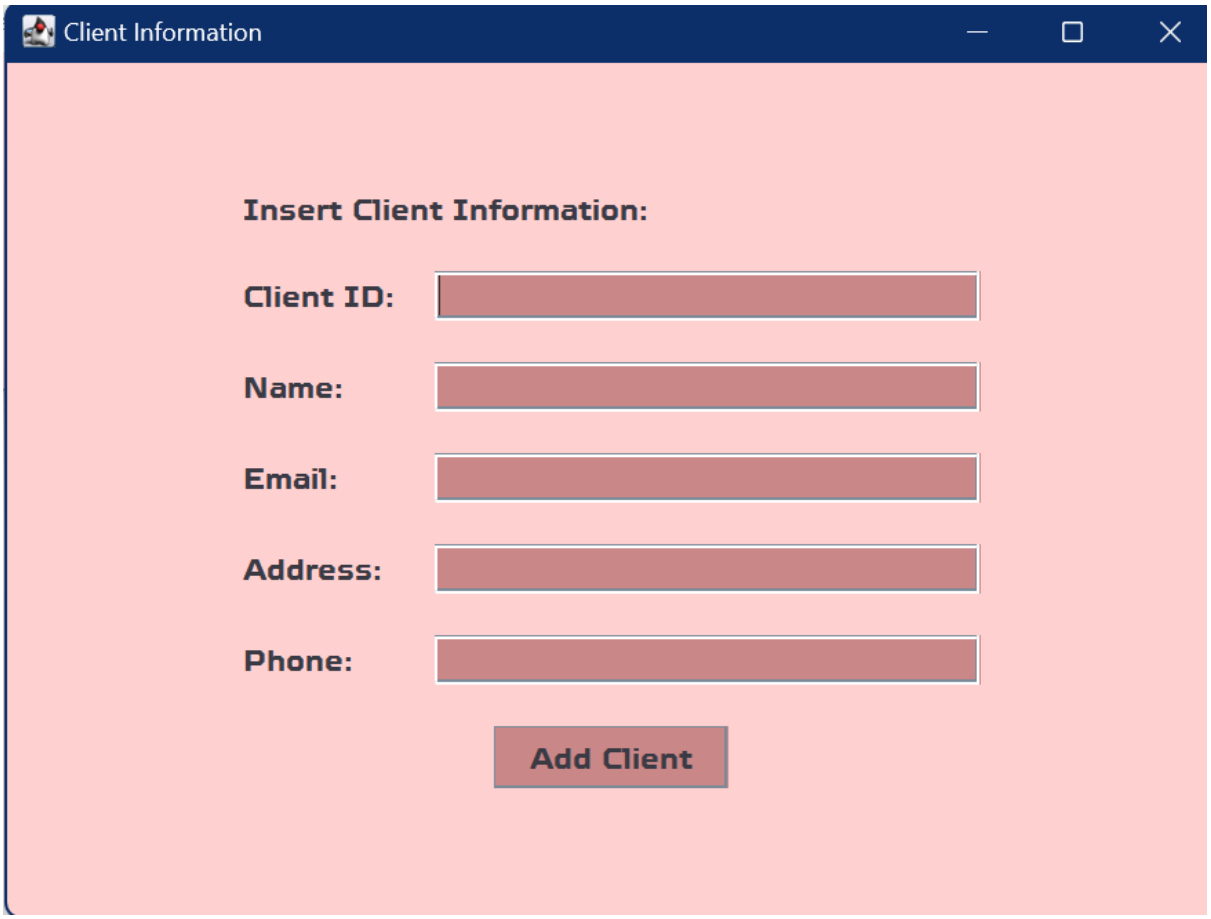
4.3.4 Dao

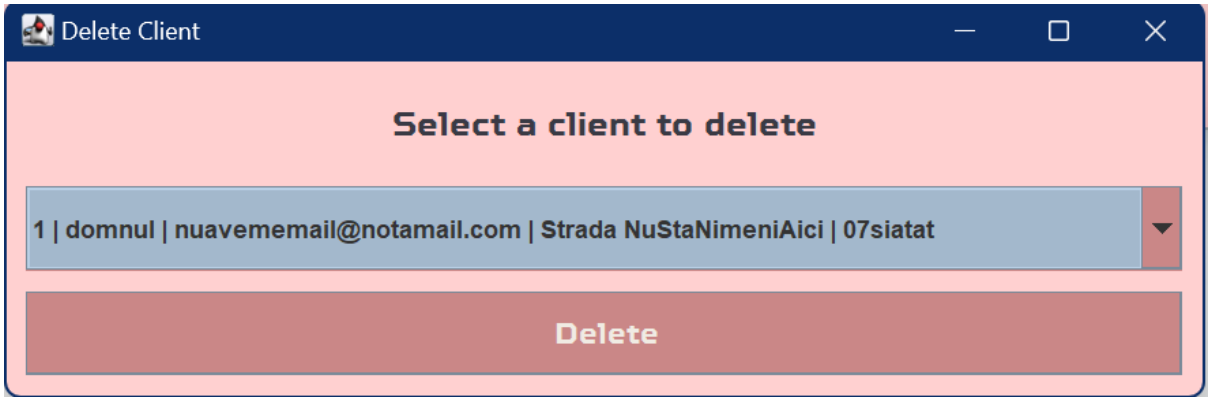
- ConnectionFactory - gestionează conexiunea la baza de date și oferă metode pentru executarea interogărilor SQL.
- AbstractDAO - implementează metode generice folosind tehnici de reflexie pentru a asigura funcționalitatea CRUD pentru orice clasă, aducând reutilizabilitate la cod și facilitând interacțiunea cu baza de date din partea pachetului controller.

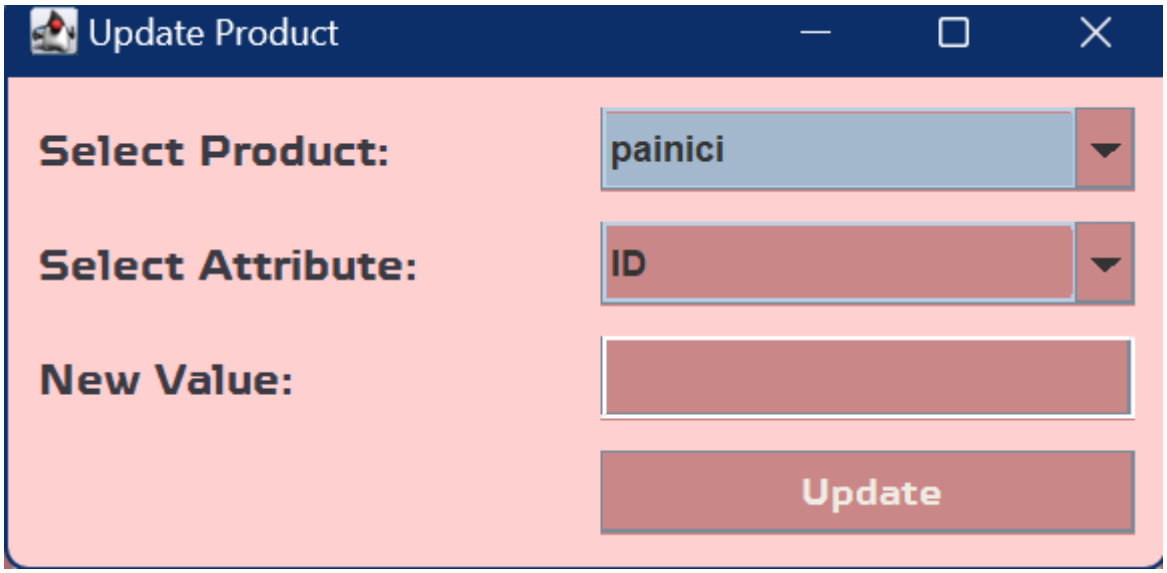
5. Prezentarea GUI




Client Operations
Add Client
Delete Client
Edit Client
View Clients
Product Operations
Add Product
Delete Product
Edit Product
View Products

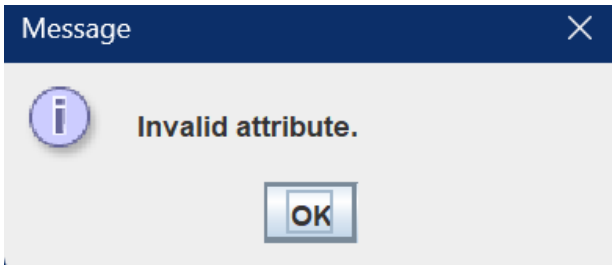

Client Information
Insert Client Information:
Client ID:
Name:
Email:
Address:
Phone:
Add Client

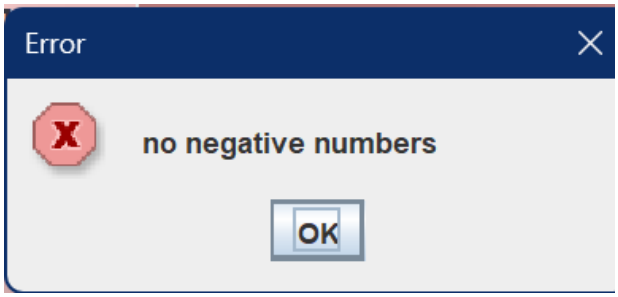


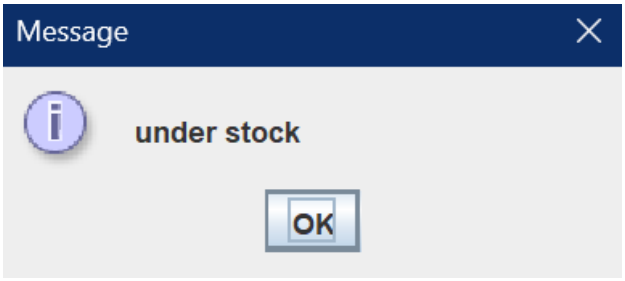


List All Product

productId	name	description	price	quantity	location
1	painici	bune	5.0	116.0	langa paine
2	paine	mai bune	6.0	82.0	langa painici
3	eclere	dulci	9.0	17.0	in vitrina
4	mere	acre	2.0	150.0	in cos







6. Concluzii

Acest proiect m-a ajutat să învăț noi tehnici în Java, precum reflexia și utilizarea genericelor, și m-a familiarizat cu arhitectura stratificată. De asemenea, am câștigat experiență în utilizarea MySQL Workbench.

7. Bibliografie

• Connect to MySql from a Java application
o https://www.baeldung.com/java-jdbc
o http://www.mkyong.com/jdbc/how-to-connect-to-mysql-with-jdbc-driver-java/

• Layered architectures
https://dzone.com/articles/layers-standard-enterprise

• Reflection in Java
http://tutorials.jenkov.com/java-reflection/index.html

• Creating PDF files in Java
https://www.baeldung.com/java-pdf-creation

• JAVADOC
https://www.baeldung.com/javadoc

• SQL dump file generation
https://dev.mysql.com/doc/workbench/en/wb-admin-export-import-management.html