



# DOCUMENTATIE ORDER MANAGEMENT

**BORZA DIANA-CRISTINA**  
GRUPA 30225 | AN 2 SEMESTRUL 2

# Cuprins

1. Obiectivul temei .....	3
1. 1. Obiectivul principal al temei .....	3
1. 2. Obiective secundare.....	3
2. Analiza problemei, modelare, scenarii, cazuri de utilizare .....	3
3. Proiectare .....	4
3. 1. Decizii de proiectare .....	4
3. 2. Diagrama UML.....	4
3. 3. Structuri de date .....	5
3. 4. Proiectare clase .....	5
3. 5. Interfete .....	5
3. 6. Pachete .....	5
3. 7. Interfata utilizator .....	6
4. Implementare .....	10
5. Concluzii.....	10
6. Bibliografie .....	10

# 1. Obiectivul temei

## 1. 1. Obiectivul principal al temei

Obiectivul principal al temei este acela de a crea o aplicatie care permite manipularea produselor dintr-un depozit. Vor exista produse, client si comenzi plasate. Aplicatia va permite operatii precum adaugare clienti sau produse, editare client sau produse, stergere clienti sau produse si plasare comenzi. Evidenta produselor, a comenzilor si a clientilor se vor pastra intr-o baza de date.

## 1. 2. Obiective secundare

Obiectivele secundare reprezinta pasii care trebuie urmati pentru a indeplini obiectivul principal, iar ele sunt definite in cele ce urmeaza.

- Analiza problemei si identificarea cerintelor

Programul va contine un meniu principal in care se va putea alege operatiile dorite. Vor exista trei interfete pentru fiecare din tipurile de obiecte Product , Client , Order unde se vor putea efectua operatii specifice fiecaruia.. Un client va putea fi editat , adaugat sau sters. De asemenea se vor putea vizualiza toti clientii existenti in baza de date. Un produs va putea fi adaugat , modificat sau sters din baza de date si se va putea vizualiza un tabel cu toate produsele existente in baza de date. In interfata pentru Order se vor putea alege un client , un produs si o cantitate dorita pentru a plasa o comanda. Daca exista stoc suficient comanda va fi plasata, altfel va fi afisat un mesaj pentru stoc insuficient. La plasarea unei comenzi stocul va fi decrementat si se va crea un bon. Acest obiectiv va fi prezentat pe larg in capitolul 2.

- Proiectarea sistemului de management

Intregul sistem va avea ca si intrari obiecte de tipul Order , Product si Client preluate din baza de date creata in MySQL. Obiectivul va fi detaliat in capitolul 3.

- Implementarea sistemului de management

Pentru implementarea aplicatiei se va tine cont de faptul ca exista obiecte de tipul Product , Order si Client care reprezinta tabelele specifice din baza de date in care se vor pastra datele. Obiectivul va fi explicat amanuntit in capitolul 4.

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

Cerinta functionala a proiectului este aceea de a implementa un sistem care proceseaza produse si comenzi intr-un depozit. Depozitul manipuleaza diferite produse si detine mai multi clienti. Un client poate plasa o comanda pentru un anumit produs. Aplicatia permite efectuarea unor operatii pe fiecare tip de obiect , dupa cum urmeaza :

→ Produs :

Un produs nou va putea fi adaugat in baza de date . Datele se vor introduce in interfata specifica , vor fi preluate si actualizate in baza de date . Un produs existent va putea fi modificat , noile sale attribute fiind introduse in interfata specifica , preluate si actualizate in baza de date . De asemenea, un produs existent va putea fi sters din baza de date . Toate produsele vor putea fi vazute intr-un tabel , impreuna cu toate attributele lor .

→ Client :

Un client nou va putea fi adaugat in baza de date . Datele se vor introduce in interfata specifica , vor fi preluate si actualizate in baza de date . Un client existent va putea fi modificat , noile sale attribute fiind introduse in interfata specifica , preluate si actualizate in baza de date . De asemenea, un client existent va putea fi sters din baza de date . Toti clientii vor putea fi vazuti intr-un tabel , impreuna cu toate attributele lor

→ Comanda :

O comanda va putea fi plasata in interfata specifica de unde se va alege un client , un produs si cantitatea dorita si se va apasa butonul pentru plasarea comenzii. Daca exista un stoc suficient atunci comanda va fi plasata, se va genera bonul si stocul va fi actualizat. Daca stocul este mai mic decat cantitatea dorita comanda nu va fi plasata.

Cazuri de utilizare :

- Produs : se doreste inserarea unui nou produs. Se va deschide interfata specifica si se vor introduce campurile specificate. In caz de success, toate campurile au fost completate corect iar produsul va fi adaugat cu success, afisandu-se un mesaj. In cazul in care un camp este gol sau nu este introdus corespunzator va fi afisat un mesaj de eroare iar produsul nu va fi adaugat.

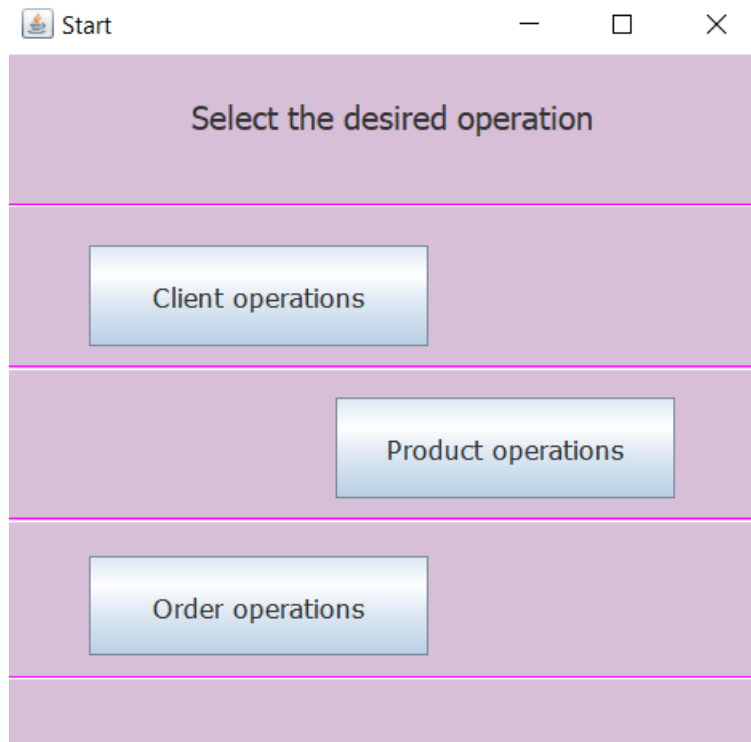




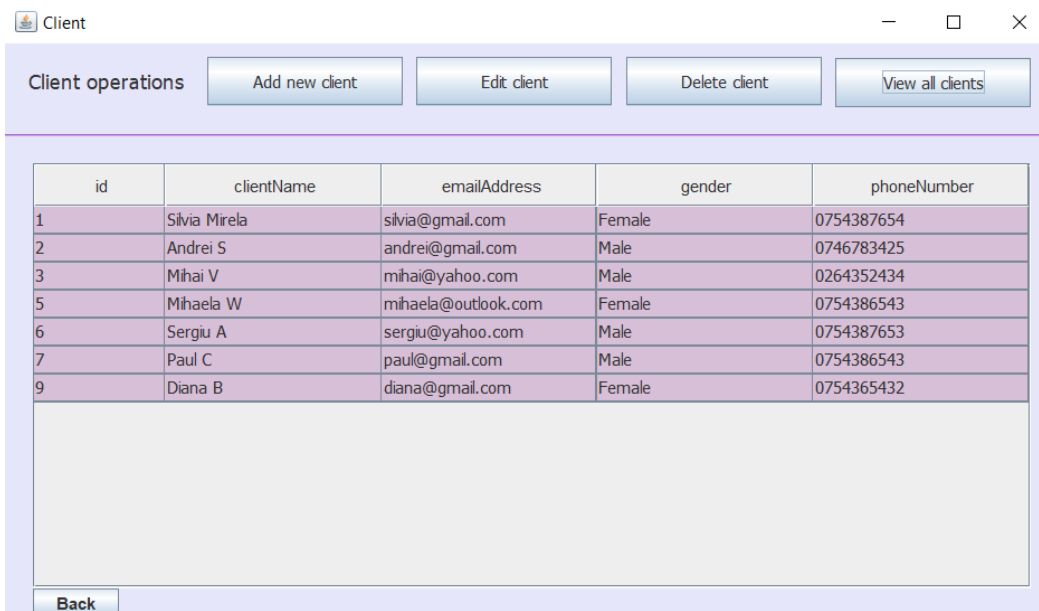
### 3. 7. Interfata utilizator

Aplicatia contine mai multe interfete pentru utilizator, toate sugestive si usor de folosit.

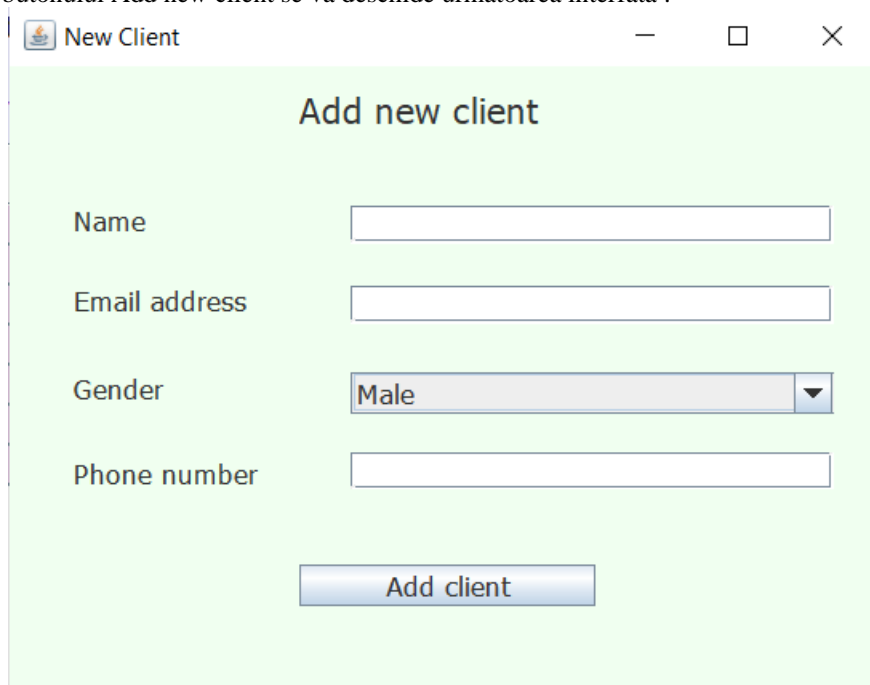
Interfata de pornire este urmatoarea si contine trei butoane de unde utilizatorul va putea alege operatia dorita



Interfata pentru client contine cinci butoane. Un buton de back pentru revenirea in pagina de start, un buton pentru vizualizarea clientilor existenti in baza de date, un buton pentru a adauga un client in baza de date, un buton pentru modificarea unui client existent si unul pentru stergerea unui client existent. Pentru stergerea unui client va trebui selectat un client din tabel si apasat butonul. Pentru a actualiza tabelul se va apasa din nou butonul pentru afisare.



La apasarea butonului Add new client se va deschide urmatoarea interfata :



New Client

### Add new client

Name

Email address

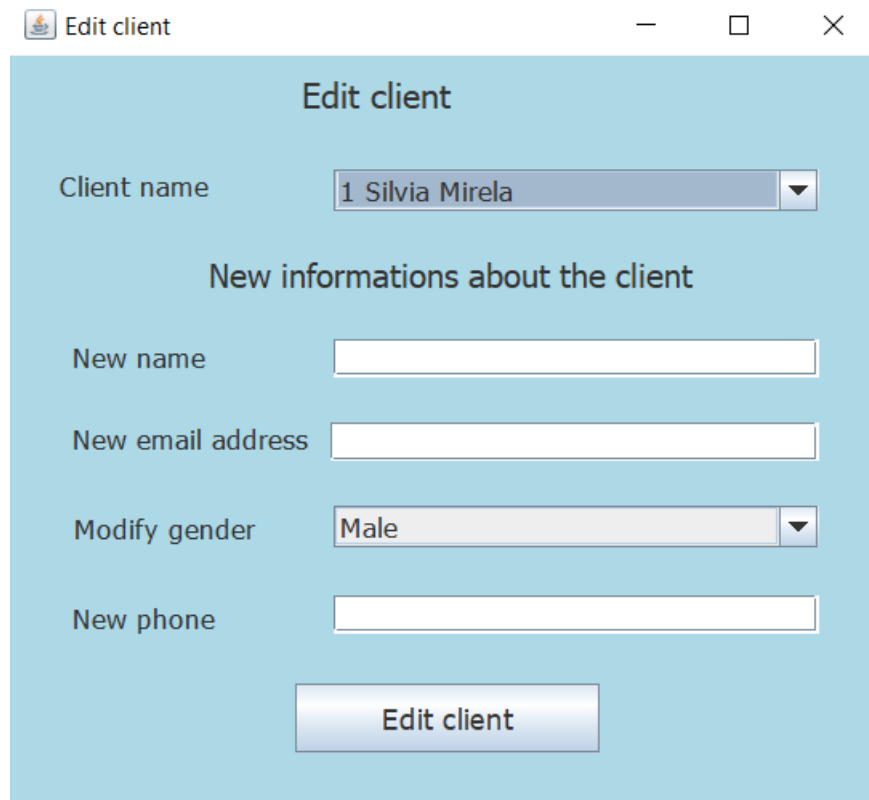
Gender

Phone number

Add client

Aici se vor introduce datele unui nou client si se va apasa butonul Add client.

La apasarea butonului Edit Client se va deschide o noua interfata unde se va putea selecta clientul pe care dorim sa il modificam si sa introducem noile date. Interfata :



Edit client

Client name

### New informations about the client

New name


New email address

Modify gender

New phone

Edit client

Interfata pentru produs contine cinci butoane. Un buton de back pentru revenirea in pagina de start, un buton pentru vizualizarea produselor existente in baza de date, un buton pentru a adauga un nou produs in baza de date, un buton pentru modificarea unui produs existent si unul pentru stergerea unui produs existent. Pentru stergerea unui produs existent va trebui selectat un produs din tabel si apasat butonul pentru stergere. Pentru a actualiza tabelul se va apasa din nou butonul pentru afisare.


 Product
 — □ ×

**Product operations**
Add new product
Edit product
Delete product
View all products

id	productName	price	stock	weight	returnable
1	Peach	20	6	0.7	false
2	Frozen Peas	15	1	0.4	false
3	Chips	12	4	0.7	true
4	Meat	22	2	1.2	true
5	Water bottle	2	3	0.5	true
6	Pepsi twist	3	1	0.5	false
7	Strawberries	30	5	1.7	false
8	Bread	10	4	0.8	false
9	Cheddar cheese	25	10	0.2	true
11	Tomato sauce	6	3	1.2	true
12	Pasta	20	7	0.8	false
13	Chicken wings	34	3	1.9	false
14	Water	4	4	0.5	true
15	Carrots	16	2	2.0	true

Back

La apasarea butonului Add new product se va deschide urmatoarea interfata. Aici se vor introduce datele unui nou produs si se va apasa butonul Add product.

 New Product
 — □ ×

### Add new product

Name

Price

Stock

Weight

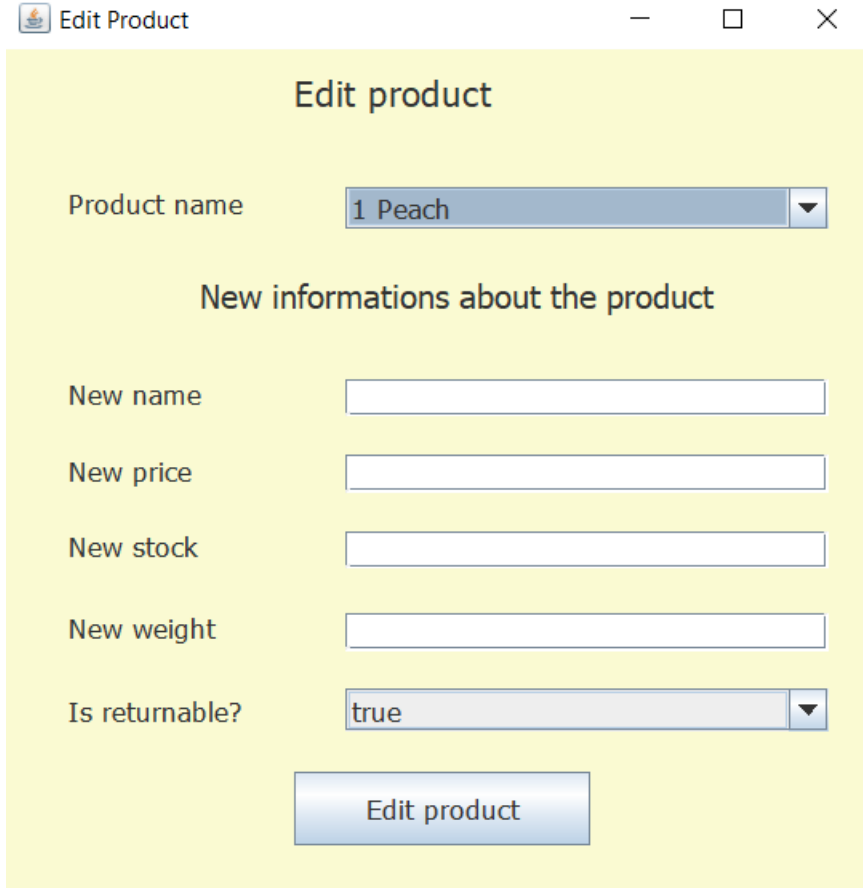
Is returnable?

true ▼

Add product



La apasarea butonului Edit product se va deschide o noua interfata unde se va putea selecta produsul pe care dorim sa il modificam si sa introducem noile date. Interfata :



Product name 1 Peach ▼

New informations about the product

New name

New price

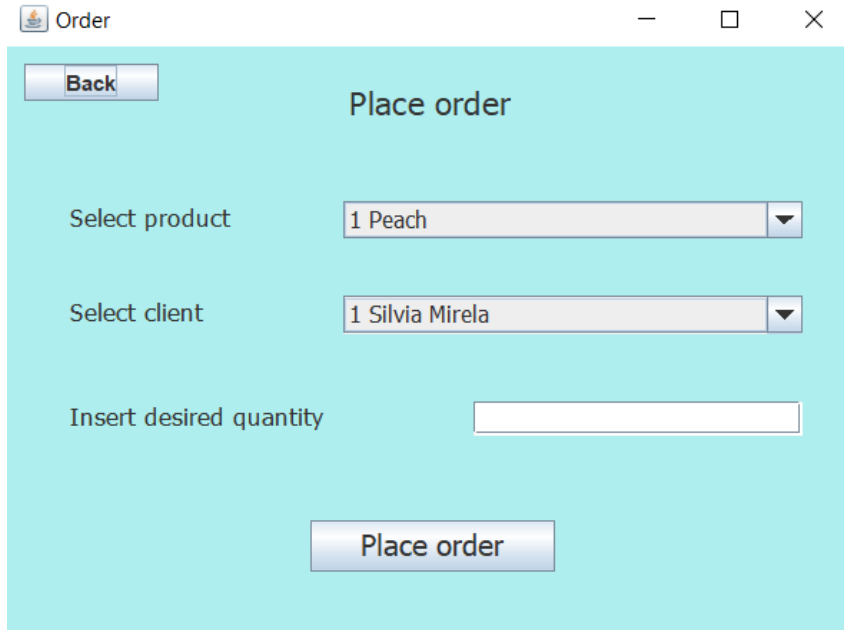
New stock

New weight

Is returnable? true ▼

Edit product

Interfata pentru plasarea unei comenzi contine doua butoane. Unul pentru revenirea in pagina de start si unul pentru plasarea comenzii.



Order

Back

Place order

Select product 1 Peach ▼

Select client 1 Silvia Mirela ▼

Insert desired quantity

Place order

## 4. Implementare

Aplicatia contine 24 de clase si o interfata structurate astfel :

Interfata **Validator** este interfata ce defineste un validator si este implementata de clasele **EmailValidator** , **PhoneNumberValidator** si **PriceValidator** . Ele contin metode pentru validarea anumitor atribute ale obiectelor.

Clasa **ClientBLL** este clasa care contine metodele specifice clientului: cautare client dupa id, stergere client cu id specificat, editare client cu id specificat, inserare client nou, returnare lista cu toti clientii din baza de date si returnare lista cu numele tuturor clientilor.

Clasa **ProductBLL** este clasa care contine metodele specifice unui produs: cautare produs dupa id, stergere produs cu id specificat, editare produs cu id specificat, inserare produs nou, returnare lista cu toate produsele din baza de date, returnare lista cu numele tuturor clientilor, determinare pret pentru un produs cu id specificat si determinare stoc pentru un produs cu id specificat.

Clasa **OrderBLL** este clasa care contine metodele specifice unei comenzi: creare comanda si obtinere lista de comenzi din baza de date.

Clasa **TableBLL** este clasa care contine metoda pentru obtinerea tabelului si a headerului acestuia folosind reflection technique.

Clasa **ConnectionFactory** este clasa care contine metode pentru conexiunea cu baza de date: creare conexiune, inchidere conexiune, inchidere result set si statement.

Clasa **AbstractDAO** este clasa abstracta care defineste metode pentru accesul la baza de date pentru obiecte de orice tip T ( Order , Product , Client ) : metoda ce returneaza o lista cu toate obiectele din baza de date de tipul T , metoda care returneaza un obiect de tipul T din baza de date cu un id specificat, metoda care creeaza lista de obiecte preluate din result set, metode de update, stergere si adaugare a unui obiect de tip T in baza de date.

Clasele **ClientDAO** , **OrderDAO** si **ProductDAO** extind clasa abstracta AbstractDAO si implementeaza metode specifice clientului sau a produsului care nu sunt mostenite din AbstractDAO.

Clasele **Client** , **Product** si **Order** sunt clasele care contin modelele de date prezente si in baza de date. Reprezinta obiectele modelate de catre aplicatie.

Clasele **ClientGUI**, **ModifyClientGUI**, **ModifyProductGUI**, **NewProductGUI**, **NewClientGUI**, **OrderGUI**, **ProductGUI**, **StartGUI** reprezinta clasele care implementeaza interfata grafica a aplicatiei.

Clasa **Start** reprezinta clasa care contine metoda main de invocare a aplicatiei. Aceasta clasa va fi supusa rularii initiale a aplicatiei.

## 5. Concluzii

Implementarea programului ilustreaza tehnicile de programare orientate obiect, modul de impartire al claselor, de stabilire a relatiilor, de implementare eficienta si concisa, de pastrare a datelor prin intermediul unei baze de date relationale. de generare a JavaDoc dar si de implementare a unei interfete accesibila oricarui utilizator. Prin aceasta tema mi-am aprofundat cunostintele in limbajul java si a tehnicilor de programare, cat si a modului de a conecta o aplicatie java la o baza de date si de a lucra concomitant cu aceasta.

Aplicatia ar putea fi imbunatatita ulterior prin crearea unei interfete mai interactive si prin adaugarea unor functionalitati noi precum logarea in functie de anumite tipuri de client sau angajati.

## 6. Bibliografie

<https://docs.oracle.com/javase/tutorial/jdbc/basics/processingsqlstatements.html>

<https://docs.oracle.com/en/java/>

<https://www.oracle.com/technical-resources/articles/java/javareflection.html>

<https://www.oreilly.com/library/view/software-architecture-patterns/9781491971437/ch01.html>