**DOCUMENTATIE**

**Tema 3**

**Nume student: Faur Sergiu**

**Grupa: 30224**

**Cuprins**

1.Obiectivul lucrarii . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

2.Analiza problemei, modelare, scenarii, cazuri de utilizare . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

3.Proiectare . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

**4.Implementare . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .8**

[5.Rezultate](#_Toc95297889) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .11

6.Concluzii . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .13

7.Bibliografie . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .13

**1.OBIECTIVUL TEMEI**

**Obiectivul Principal**

**Obiectivul principal al acestui proiect este de a face o aplicatie pentru a menegeria comenzile unor clienti pentru un deposit.**

**Obiective secundare**

**Analiza problemei si identificarea solutiei – la aceasta se doreste a se intelege problema si cee ace trebuie implementat – se va discuta mai detaliat in capitolul 2**

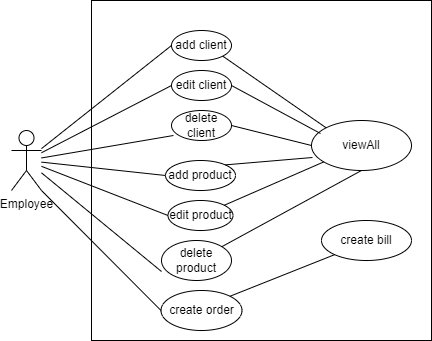
**Sa se faca designul aplicatiei – se doreste a se gandi cum va arata aceasta aplicatie pentru a fi cat mai usor pentru utilizator de a lucra cu ea – se va discuta in capitolul 3**

**Implementarea aplicatiei - pentru aceasta se va discuta structura claselor si implementarea interfetei grafice – se va discuta in capitolul 4**

**Testarea aplicatiei – la aceasta parte trebuie sa vedem functionalitatea aplicatiei – se va discuta in capitolul 5**

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

**DIAGRAMA USE-CASE**



Use case : adauga produs / adauga client

Actor principal : employee

Scenariul de success:

Angajatul introduce datele pentru client/produs

Angajatul apasa pe butonul de adaugare

Angajatul apasa pe butonul de viewAll si vede clientul/produsul inserat

Scenariul alternativ: angajatul introduce date cu avand tipul de date gresit caz in care va da eroare

Use case: edit client/product

Actor principal:angajat

Scenariul de success

Angajatul introduce datele ce se modifica

Angajatul apasa pe butonul de update

Angajatul apasa pe butonul de viewAll si vede clientul/produsul modificat

Scenariul alternativ: angajatul introduce date cu avand tipul de date gresit caz in care va da eroare

Use case : delete client/product

Actor principal:angajat

Scenariul de success

Angajatul introduce id-ul clientului/produsului ce se sterge

Angajatul apasa pe butonul de delete

Angajatul apasa pe butonul de viewAll si vede ca clientul/produsul nu mai este

Scenariul aleternativ: angajatul introduce un id ce nu exista ,caz in care va da eroare

Use case: create order

Actor principal:angajat

Scenariul de success

Angajatul introduce id-ul clientului , al produsului ,cantitatea si se genereaza pretul

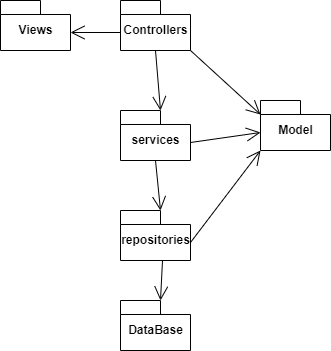
Apasa pe butonul de create order

Se genereaza un bill pentru comanda

Scenariul alternativ: nu mai sunt destule produse pe stock si astfel se afiseaza mesajul FULL STOCK

# 3.Proiectare

DIAGRAMA DE PACHETE



Pachetul Views contine interfetele grafice pentru client product si order cat si cea princpala.

Pachetul Controllers contine clasa Controller care controleaza interfetele grafice

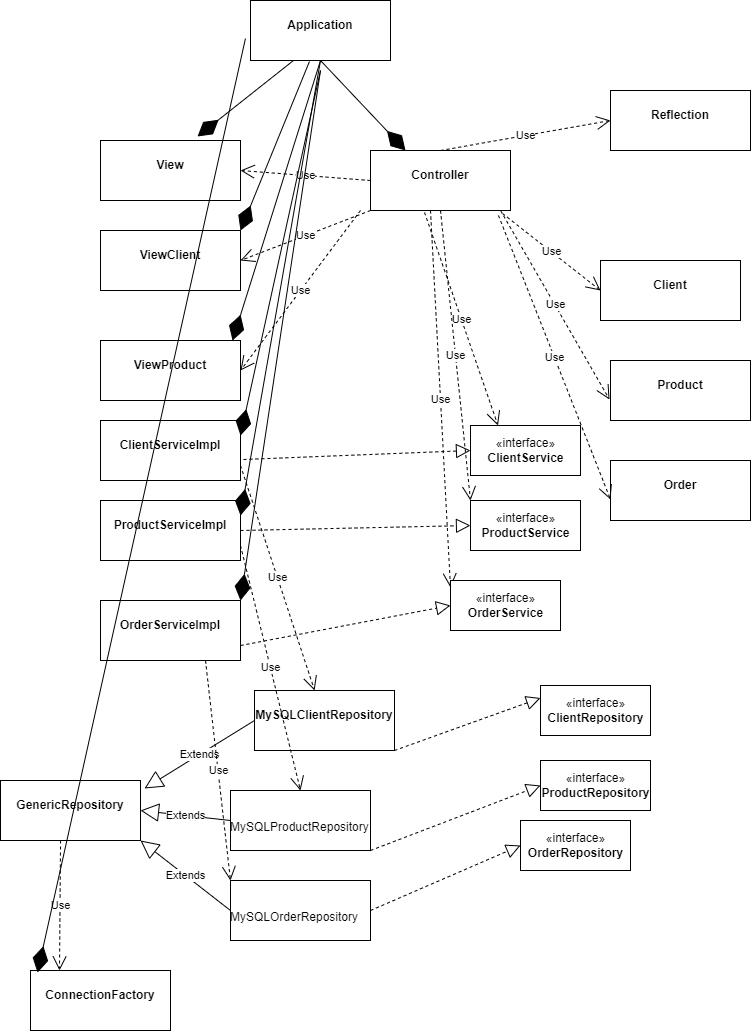
Pachetul services contine clase ClientServiceImpl, ProductServiceImpl , OrderServiceImpl cat si interfetele pe acare acestea le implementeaza.

Pachetul repositories contine clasa Genericrepository , clasele care extend aceasta clasa MySQLClientRepository , MySQLProductRepository, MySQLOrderRepository , cat si interfetele pe care acestea le implementeaza.

Pachetul DataBase contine clasa ConnectionFactory ce face legatura cu baza de date.

Pachetul Model contine clasele Client , Product si Order.

Diagrama de clase:



VIEWS

Clasele din acest pachet (View , ViewClient, ViewProduct si viewOrder) reprezinta ferestrele pentru aplicatia de gestiune a comenzilor.

CONTROLLERS

Clasa Controller din acest pachet are rolul de a gestiona view-urile si modul in care acestea apar .

MODEL

Clasele din acest pachet descriu modelul aplicatiei (Client Order Product) .

SERVICES

Clasele ClientServiceImpl , ProductServiceImp , OrderServiceImpl , cat si interfetele pe care acestea le implementeaza contin metodele folosite pentru a da functionalitate metodelor pentru CRUD , acestea reprezinta parctic business-logic .

REPOSITORIES

Clasele din acest pachet(MySQLClientRepository, MySQLProductRepository , MySQLOrderRepository, GenericRepository) cat si interfetele pe care acestea le implementeaza contin implementarea concreta a metodelor pentru CRUD (aceasta este in calsa generic aceste metode apelandu-se pentru celelalte clase in functie de clasa ce o apeleaza ) .

DATABASE

In aceast pachet este prezenta clasa de ConnectionFactory ce realizeaza conxiunea cu baza de date

# 4.Implementare

Fiind un proiect OOP , se bazeaza pec lase sip e relatiile dintre acestea , in capitolul acesta vom descrie fiecare clasa cu functionalitatea sa si cee ace reprezinta .

Clasele din pachetul Model :

CLIENT

In aceasta clasa se retin atributele pentru un client si de asemenea se construiesc settere si gettere pentru a putea accesa informatiile despre aceste attribute si de alte clase.

PRODUCT

In aceasta clasa se retin atributele pentru un produs si de asemenea se construiesc settere si gettere pentru a putea accesa informatiile despre aceste attribute si de alte clase.

ORDER

Ca si cele de mai sus ,se retin atributele unei comenzi si se construiesc gettere si settere pentru accesarea si modificarea lor din alte clase.

Clasele din pachetul Repositories :

GenericRepository

In aceasta clasa se implementeaza metodele pentru CRUD intr-un mod generic , putand fi utilizate pentru toate cele 3 calse din pachetul model. Astfel am creat metodele de update , findAll , findById,insert, delete . Pentru ca lucram cu o baza de date in SQL , a trebuit sa creez metode pentru creeare de interogari . Astfel am facut metodele de createInsertStatement , createSelectQuery , createSelectAllQuery , createUpdateQuery si createDeleteQuery , acestea avand rolul de a genera o interogare SQL corespunzatoare pentru operatiile de create update read si delete.

De asemenea in aceasta clasa exista metoda de createObjects care are rolul sa creeze obiecte in clasa de generic de tiul altor clase.

De asemenea in aceasta clasa se afla atributul de ConnectionFactory .

MYSQLCLIENTREPOSITORY

In aceasta clasa ce extinde clasa de generic am apelat metodele de CRUD folosind clasa parinte , astfel acestea executandu-se ca si cum ar fi fos un Client.

MYSQLPRODUCTREPOSITORY

In aceasta clasa ce extinde clasa de generic am apelat metodele de CRUD folosind clasa parinte , astfel acestea executandu-se ca si cum ar fi fos un Produs .

MYSQLORDERREPOSITORY

De asemenea , ca in clasele anterioare , in aceasta clasa se apeleleaza metodele CRUD folosind clasa parinte care este GenericRepository , acestea executandu-se ca si cum ar fi o comanda .

CLIENTREPOSITORY

Aceasta este o interfata ce contine metodele de CRUD specific pentru un client.

PRODUCTREPOSITORY

Aceasta este o interfata ce contine metodele de CRUD specific pentru un produs .

ORDERREPOSITORY

De asemenea este o interfata ce contine metodele de CRUD pentru o comanda.

Clasele din pachetul SERVICES :

CLIENTSERVICEIMPL

In aceasta clasa se afla metodele de CRUD pentru un Client , acestea apelandu-se cu ajutorului atributului repository. Pe langa apelarea acestora in aceasta clasa se si testeaza corectitudinea parametrilor de intrare .

PRODUCTSERVICEIMPL

In aceasta clasa se afla metodele de CRUD pentru un Produs, acestea apelandu-se cu ajutorului atributului repository. Pe langa apelarea acestora in aceasta clasa se si testeaza corectitudinea parametrilor de intrare .

ORDERSERVICEIMPL

In aceasta clasa se afla metodele de CRUD pentru o comanda , acestea apelandu-se cu ajutorului atributului repository. Pe langa apelarea acestora in aceasta clasa se si testeaza corectitudinea parametrilor de intrare .

CLIENTSERVICE

Aceasta este o interfata ce contine metodele de CRUD ce se folosesc in ClientServiceImpl .

PRODUCTSERVICE

Aceasta este o interfata ce contine metodele de CRUD ce se folosesc in productServiceImpl .

ORDERSERVICE

Aceasta este o interfata ce contine metodele de CRUD ce se folosesc in OrderServiceImpl .

Clasa din pachetul DATABASE :

Clasa ConnectionFactory are rolul de a ne conecta la baza de date din MySql . Aceasta clasa este utila mai ale pentru ca daca am vrea sa schimbam de exemplu din MySql cu altceva ar trebui doare sa modificam aceasta clasa .

Clasele din Pachetul VIEWS :

VIEW

Aceasta este fereastra principala , prima care apare la rularea aplicatiei si contine butoanele pentru order, client si product .

VIEWCLIENT

Aceasta fereastra apare in moemntul in care apasam pe butonul de client , continand butoane pentru a crea un client , a face update la un client , a sterge un client si a lista toti clientii din baza de date .

VIEWPRODUCT

Aceasta fea=reastra apare in momentul cand apasam pe butonul product , continand la fel butoanele de insert update delete si viewALL

VIEWORDER

Analog ca la cele doua de mai sus , dar aceasta apare cand apasam pe Orders si are doar doua butoane de create si viewAll.

Pachetul CONTROLLERS :  
CONTROLLER

In aceasta clasa se menegeriaza ferestrele si cee ace trebuie sa se intample la apasarea butoanelor din interfata grafica , adica se implementeaza metodele de ActionPerformed pentru ActionLitenerii din cele 4 View-uri.

De asemenea aceasta clasa foloseste clasa Reflection pentru a afla proprietatile unui obiect .

Ce ar mai fi de specificat este faptul ca pentru creearea unei coemnzi , in metoda de actionPerformed corespunzatoare butonului , pretul este calculate automat , iar totodata se face si un update la stockul produsului respectiv , iar in cazul in care cantitatea este mai mare decat stockul , se afiseaza mesajul FULL STOCK.

Clasa REFLECTION :

Aceasta clasa contine doar metoda static de RetrieveProprieties ce are rolul de a afla si proprietatile unui obiect si a le afisa in tabelul din interfata grafica.

Clasa APPLICATION

Aceasta clasa contine main-ul , acesta fiind pentru rularea programului.

# 5.Rezultate

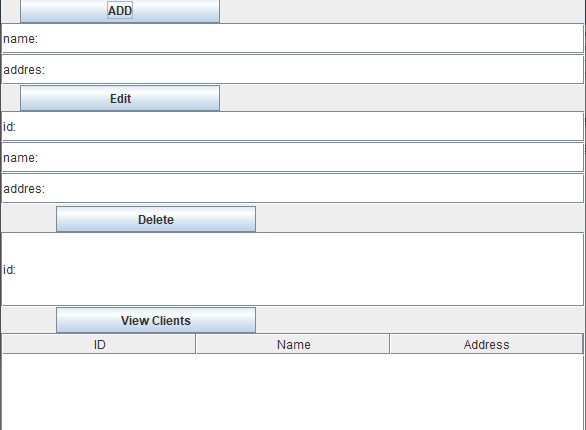
Rezultatele au fost testate in permanenta in timpul crearii codului prin testarea pe rand a fiecarei metode de CRUD si verificarea eventualelor erori de functionalitate.

In final aplicatia arata in felul urmator:

Pagina principala:

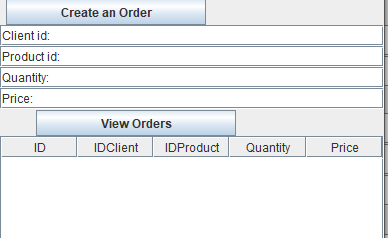


Pagina pentru Client:



Pagina pentru produs este realizata in mod analog dar cu campurile schimbate dupa atributele produsului.

Pagina pentru Order :



**6.Concluzii**

Acest proiect a fost benefic pentru intelegerea si experimentarea lucrului cu o baza de date si este foarte bun pentru definirea conceptelor de baza pentru lucrul cu o baza de date

Niste dezvolari ulterioare ar fi:

* Modificarea aplicatie astfel incat , Clientul sa se poata loga sis a plaseze comanda , aceasta fiind vazuta de un angajat.
* De asemenea modificarea aplicatie astfel incat sa se imparata fereastra de login pentru administrator , angajat , client .

# 7.Bibliografie

<https://app.diagrams.net>

<https://dsrl.eu/courses/pt/materials/A1_Support_Presentation.pdf>

<https://www.geeksforgeeks.org/>