**DOCUMENTATIE TEMA 3**

**ORDER MANAGEMENT**

**Csillag Szabolcs Andras**

**Grupa 30227**

**Profesor: Dan Mitrea**

Cuprins:

1. Cerintele temei

2. Analiza problemei

3.Proiectare

4.Implementare

5.Testare

6.Dezvoltari ulterioare si Concluzii

7.Bibliografie

# Cerintele Temei

( cerintele problemei )

Considerati o aplicatie OrderManagement pentru a procesa comenzile clientilor . Sa se foloseasca o baza de date relationala pentru a salva produsurile,clientii si comenzile ( order-urile). Aplicatia ar trebui sa foloseasca un layered architecture prezentat in suportul material Assignment\_3\_Indications.pdf si ar trebui sa contina minim urmatoare clase:

* Model classes – modelul de date pentru aplicatie
* Business Logic classes – implementarea aplicatiei logic
* Presentation classes – implementarea user input / output
* Data Access classes – implementeaza accessul pentru o baza de date

Aplicatia ar trebui sa proceseze comenzi dintr-un text file primit ca un argument, performati operatiile cerute , salvati datele in baza de date , generati reporturi in pdf format. Se pot adauga clase si pachete noi pentru a implementa functionalitatea toala a aplicatiei. Sa implementam un “parser” pentru a citi comenzile in Presentation layer (in loc de standard grapical user interface) , si pentru a genera pdf file , cand comanda este report sau o anumita factura.

**Obiective:**

## a)Obiectiv Principal:

(Obiectivul principal pe care il urmarim in acest proiect)

Obiectivul principal unei comenzi este de a oferi clientului precizat un anumit produs cu o anumita cantitate, si pentru ca fiecare produs are un pret diferit sa se genereze un fisier pdf cu factura care precizeaza plata totala . De asemenea stockul pentru produsul respectiv sa se modifice (quantityold-quantitycerut) . Daca cantitatea ceruta este mai mare decat cea aflata in stock sa se pune in fisierul pdf doar un mesaj in care se precizeaza asta. Comenzile pot fi de mai multe feluri , le vom descrie in obiectivele secundare. Proiectul sa poata fi rulat cu un runnable jar file generat prin export->java ->runnable jar file: java –jar PT2020\_30227\_Csillag\_Szabolcs\_Assignment\_3.jar

## b)Obective Secundare:

( comenzille care pot aparea in fisierul de intrare commands.txt )

|  |  |  |
| --- | --- | --- |
| **Numele Comenzii** | **Sintaxa comenzii** | **Descrierea** |
| Adaugarea unui client in baza de date | Insert client: Ion Popescu, Bucuresti | Insereaza in baza de date un client cu numele Ion Popescu si cu adresa Bucuresti |
| Stergerea unui client din baza de date | Delete client: Ion Popescu, Bucuresti | Sterge din baza de date clientul cu numele Ion Popescu si cu adresa Bucuresti |
| Adaugarea unui produs in baza de date | Insert product: apple, 20, 1 | Inserarea unui produs in baza de date cu numele apple , cantitatea 20 si pretul 1 .  In cazul numele acestui produs exista , nu il inseram din nou ,ci adaugam la cantitatea veche cea actuala |
| Stergerea unui produs din baza de date | Delete product: apple | Stergerea din baza de date a produsului cu numele : apple |
| Crearea unei comenzi pentru un client | Order: Ion Popescu, apple, 5 | Crearea unei comenzi pentru clientul : Ion Popescu pentru produsul cu numele apple,cerand cantitatea 5. Astfel vom scadea stockul produsului apple cu 5, deci vom avea 15 cantitati. Generarea unui bill(facture) in pdf format cu total price=5(price\*cantitatea ceruta) |
| Generarea reporturilor | Report client  Report order  Report product | Generarea unor pdf reports cu clientii/comenzile/produsele puse intr-o forma de table . Repoartele ar trebui sa contine informatiile corespunzatoare pentru tobelul ales client / orderi / product . Tabelurile trebuie accesate din baza de date sub forma SELECT \* FROM …. , puse intr-un table macar visual in PDF file |

# Analiza Problemei

( use case-uri / scenarii care au o probabilitate mare sa apara )

**Use case-uri / Scenarii**

Utilizarea acestei aplicatii se face cu introducerea diferitor comenzi de forma tabelului anterior.Se scrie in argumentul programului numele fisierului in care punem comenzile(de ex: commands.txt).

Putem verifica in baza de date cele 4 tabele , cum se modifica ele si cum arata , accesand localhost/phpmyadmin:

* Client(id,name,address)
* Product(id,name,quantity,price)
* Orderi(id,nameclient,nameproduct,quantity)
* Bill(id,nameclient,nameproduct,quantity,totalprice)

Factura(bill) se va genera si sub forma de pdf file cu numele “Order” urmat de un numar care reprezinta numarul comenzii. Rapoartele de client , orderi , product se vor genera sub forma unor fisiere pdf cu numele “ReportClient” , ”ReportOrder” , “ReportProduct” si fiecare urmat de un numar care specifica al catelea report de acel fel s-a generat.

# Proiectare

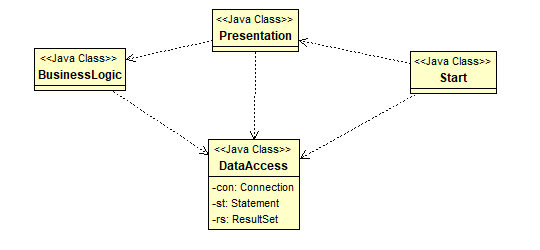
Aici voi descrie etapele de proiectare folosite pentru realizarea acestui proiect ,dar mai ales gandirea constructiei ,si metodele alese.

## a)Structuri de date

(Ce structuri de date am folosit ?) Ca structura de date nu am folosit nimic superficical , am citit din fisier linie cu linie si pentru fiecare am executat ce era necesar in baza de date. Astfel am simplificat mult aceasta problema de aceea nici nu inteleg de ce ar necesita de o documentatie asa lunga .

## b)Diagrama de clase

Diagrama de clase,adica schema UML a proiectului am generat-o in ecplise si apareau prea multe detalii. Dar ca sa incapa frumos pe pagina ,am decupat un pic de jos. Am lasat doar ceea ce este esential, pentru ca metodele in sine le voi discuta mai jos in detaliu pe fiecare.



## Algoritmi

* In acest proiect nu am folosit un algoritm anume ,cum ar fi Greed sau Dijkstra , am folosit doar logica matematica folosind baza de date si accesul acestuia .
* Se fac inserari, stergeri , modificari si anumite cautari in tabelele implementate.

# Implementare

Fiecare din cele 4 clase are rolul ei in program , Presentation se ocupa cu citirea fisierului si generarea fisierelor pdf, DataAccess are rolul de a conecta programul cu baza de date si de a face operatiile necesare , adica stergerea,inserarea,modificarea si selecturile . BusinessLogic este pentru modelarea informatiei ,ea face legatura dintre cele doua . Start da drumul programului doar.

**4.1 Clasa DataAccess :**

-contine 3 variabile instante: con de tip Connection , st de tip Statement, rs de tip ResultSet

-avem constructorul care conecteaza programul nostrum cu baza de date,o face o singura data , memoram conexiunea asta in con , si in st salvam statementul acestuia

-avem mai multe metode ,fiecare facand ceva in baza de date:

* Order , care primeste 2 argumente de tip string : numele si product si un int: quantity , vom insera in tabelul orderi aceste date , dupa care facem un query ,care sa ne scoata cantitatea si pretul produsului respective, daca cantitatea in stock e mai mica cu ceea a cantitatii primit ca parametru ,atunci returnam 0. Altfel scadem din stock cantitatea comandata si returnam pretul unui produs.
* insereazaClient : va insera in baza de date clientul cu numele si adresa primite ca parametru
* insereazaProduct: va insera in bd produsul cu numele ,cantitatea si pretul primit in parametru, dar daca exista numele produsului in baza de date , atunci nu o va insera din nou ,doar va adauga la cantitatea veche , cea primita ca parametru
* insereazaBill: va insera in bd facture cu numeclient , numeproduct , quantity , totalprice primit ca parametru
* deleteClient : sterge clientul cu numele si adresa primita ca parametrii
* deleteProduct: sterge produsul cu numele primit ca parametru
* reportClient: va returna un string cu toti clientii existenti in baza de date
* reportProduct: va reurna un string cu toate produsele aflati in baza noastra de date
* reportOrder: va returna un string cu toate comenzile aflata in tabela orderi din bd

**4.2 Clasa Start**

-contine metoda main , care apeleaza constructorul din DataAccess si da drumul proiectului prin metoda start din clasa Presentation

**4.3 Clasa Presentation**

-lucreaza cu citirea datelor din fisier folosind metoda parser si genereaza rapoartele pdf necesare

-are 3 variabile instante statice de tip int:nr\_order , nr\_client , nr\_product si 2 metode:

-a doua este aia mai usoara care primeste ca parametrii stringul nume si stringul continut, acesta va genera un fisier pdf cu numele :nume concatenate cu un extensie .pdf ,iar continutul il va memora sub forma unui paragraph nou

-prima clasa este cea care citeste linie cu linie din fisierul text(primit ca parametru) ,si executa comanda cu comanda ,folosindu-se de conexiunea cu baza de date(primit si el ca parametru) , daca comanda e unul din cele 3 rapoarte atunci doar apelam a doua metoda cu numele de tipul raportului si continutul returnat de metoda din DataAccess corespunzatoare ,in rest apeleaza metoda verific din BusinessLogic

**4.4 Clasa BusinessLogic**

-are o variabila static de tip int i , si un string de stringuri :valori

-are o singura metoda : verific , care primeste conexiuunea cu baza de date , sirul1 pana la ‘:’ si sirul2 dupa acesta in comanda actual. Folosind mai multe if-uri verificam in ce caz ne aflam si in functie de asta apelam in clasa DataAccess ceea ce este necesar pentru a fi executat in baza de date, si prelucram corespunzator datele.

**4.5 Pachetul Model**

-cele 4 clase din inainte erau in diferite pachete

-acest pachet doar arata cum sunt memorate tabelele in baza de date , nu neaparat este folosit in programul nostru , contin variabilele instante exact asa cu tipul lor cum sunt create ele in baza de date inclusiv tipul lor , probabil pentru a folosi reflection trebuia sa folosesc clasa asta dar sincer nu prea am inteles aceea varianta

# Testare

Pentru verificarea acestui program/assignment am folosit fisierul care apare in cerinta temei (linkul fiind la sfarsitul documentatiei) . Fisierul contine mai multe comenzi de tipul mentionat in tabelul din obiective secundare. Parcurgand fiecare comanda , in cazul inserarii sau stergerii modificam simplu baza de date , daca primim un report doar generam un pdf file cu acel raport , partea mai complicata as zice ca apare cand avem comanda „Order: „ Pentru ca urmeaza sa verificam daca comanda respectiva poate fi executata, adica avem in stock cantitatea ceruta , daca da vom genera un pdf file cu factura ,vom modifica cantitatea produsului respectiv, si inseram acel order in baza de date , dar daca nu se paote efectua comanda, doar inseram comanda in baza de date si generam un pdf file in care zicem ca nu se poate efectua comanda ,pentru ca nu avem cantitatea necesara pentru produsul respectiv.

# Concluzii si Dezvoltari Ulterioare

Acest program este foarte util in viata reala , de exemplu cand facem comenzi online de obicei completam un formular cu datele noastre , astfel incat managerul site-ului respective de unde comandam sa aiba acces la datele noastre ,sa ne stie numele si adresa unde trebuie efectuata livrarea. Desigur fiecare site are un stock prin care primeste produse si se tine cont de adaugarea cantitatii primite, sau chiar stergerea ei totala . Pentru cei care livreaza este bines a aiba si un table cu comenzile primite pentru a stii cui sad ea produsul, ce produs sa ii ofere si cate , desigur si pretul total al acestora. Cleintul de obicei primeste factura care ii dovedeste pretul necesar pe care trebuie s ail achite .

Observatie : Nu am inteles exact cum se lucreaza cu reflection si cu layer architecture , am implementat problema dupa cum am inteles eu , si pentru comenzile primite programul merge fara probleme , nu inteleg de ce as avea nevoie de modelul fiecarui tabel creat in baza de date , eu in acest program am mers pe ideea ca primind o anumita comanda o execut rapid , fara prea multe complicatii.

Am preferat o maniera mai personala , chiar daca probabil nu este cea mai buna abordare , este destul de precisa si din punctual meu de vedere usor de inteles.

Pentru ca nu prea am inteles cum e faza cu mysql , eu am lucrat semestrul trecut la material baza de date folosinf phpmyadmin si am l-am folosit si in acest program. Am instalat xampp si acolo aveam o varianta si pentru mysql pe care o activez , si tabelele le creez accesand localhos / phpmyadmin . Am vorbit si cu colegii care aveau aceeasi proiect pentru a ma lamuri cu cateva problem , dar uitandu-ma de discord am cam inteles, chiar daca sunt constient ca programul nu e de 10 pentru ca nu am respectat foarte mult modelul , dar daca problema functioneaza ,asta conteaza nu?

# Bibliografie

<http://www.coned.utcluj.ro/~salomie/PT_Lic/4_Lab/Assignment_3/Assignment_3_Indications.pdf>