**ORDER MANAGEMENT**

**TEMA 3**

Nume: Adam Cristina-Ioana

Grupa:30224

Facultatea de Automatica si Calculatoare

Specializarea: Calculatoare

Cuprins:

[1.Obiectivul temei 2](#_Toc513154088)

[2. Analiza problemei, asumptitii, scenarii, cazuri de utilizare, erori 3](#_Toc513154089)

[3. Proiectare(decizii de proiectare, diagram UML, structure de date, proiectare clase, interfete, relatii, packages, algoritmi, interfata utilizator, modul de tratare a erorilor) 8](#_Toc513154090)

[4. Implementare 10](#_Toc513154091)

[5. Testare 11](#_Toc513154092)

[6. Rezultate 11](#_Toc513154093)

[7. Cocluzii 12](#_Toc513154094)

[8.Bibliografie 12](#_Toc513154095)

# 1.Obiectivul temei

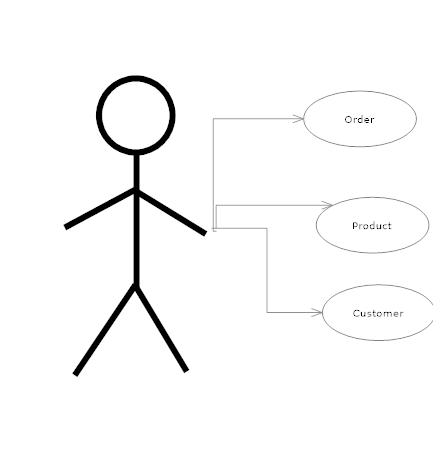
In acest proiect se dezvolta o aplicatie pentru gestiunea unor cozi in functie de timul de asteptare folosind thread-uri. Cerinta este urmatoarea:” Consider an application OrderManagement for processing customer orders for a warehouse. Relational databases are used to store the products, the clients and the orders. Furthermore, the application uses (minimally) the following classes: Model classes - represent the data models of the application (for example Order, Customer, Product), Business Logic classes - contain the application logic (for example OrderProcessing, WarehouseAdmin, ClientAdmin), Presentation classes – classes that contain the graphical user interface, Data access classes - classes that contain the access to the database.

In aceasta tema se v-a invata conceptual de Reflection si la ce ne foloseste acesta. Aplicatia permite ca utilizatorul sa poata introduca atat client cat si produse.In functie de ce avem noi in baza noastra de date putem selecta un client si un anumit produs si cantitatea acelui produs pentru a putea realiza comanda. In aceasta aplicatie folosim atat java , cat si database. Mai pe larg termenul de reflection, database, etc. v-a fi discutat la urmatoarele subpuncte.

# 2. Analiza problemei, asumptitii, scenarii, cazuri de utilizare, erori

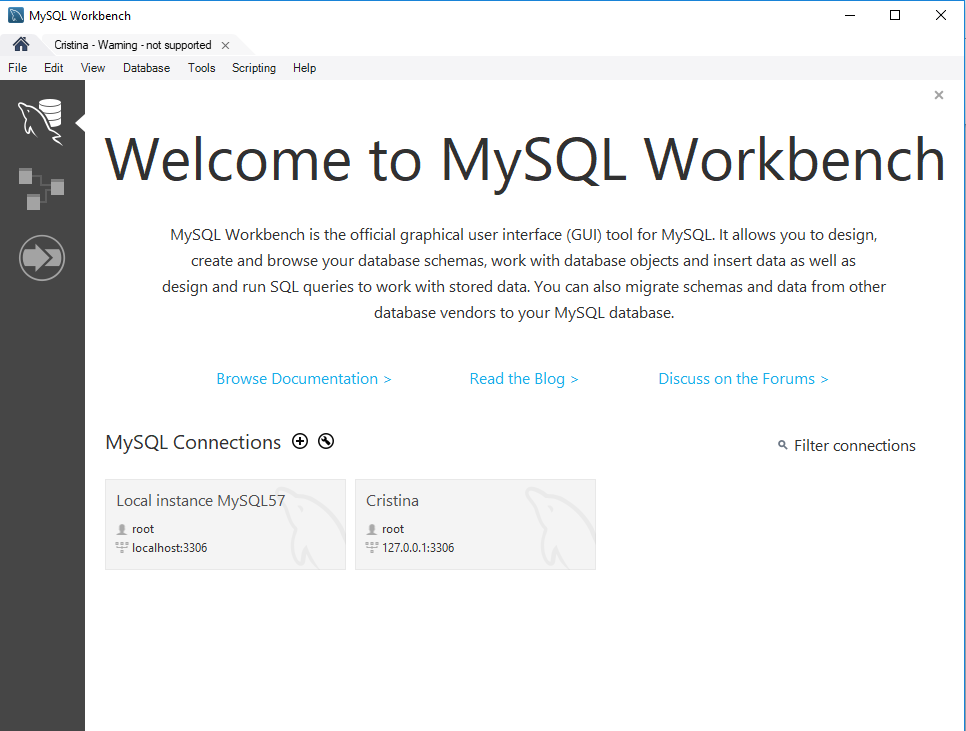
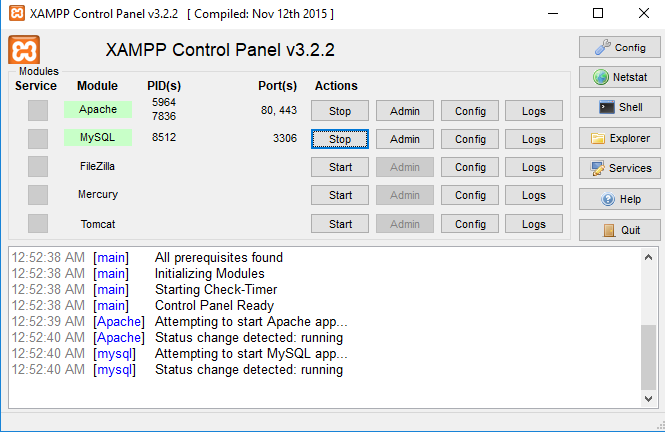
## 2.1 Analiza problemei

Diagrama use case ne prezinta toate operatiile pe care le poate efectua un utilizator. Utilizarea diagramelor use case nu este absolute necesara pentru a scrie o specificatie cu use case-uri, aceasta este utila pentru a crea o imagine generala asupra sistemului.

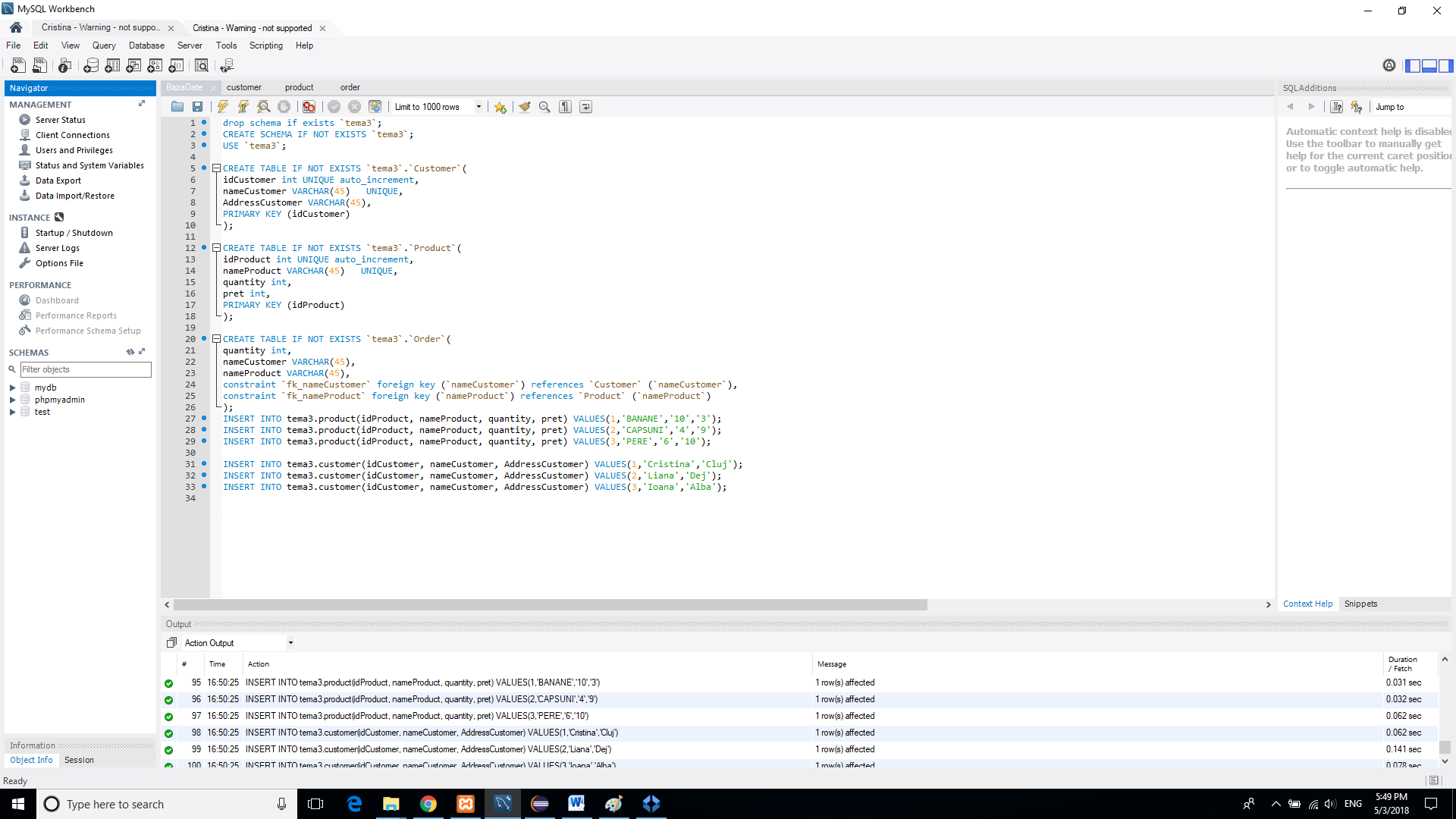


## 2.2 Asumptitii

In momentul in care utilizatorul v-a introduce un id care deja exista in baza de date, sau un nume indiferent ca e la produs sau la customer, care de asemenea exista in baza de date ii v-a genera un mesaj de eroare utilizatorului. Daca utilizatorul vrea sa stearga din baza de date un produs sau un client, acesta se v-a sterge in functie de id. Daca se scrie in JTextField un id care nu exista in baza de dat si se apasa pe butonul de deleta , v-a aparea un mesaj de eroare. De asemnea in momentul in care se face conexiunea la baza de date se v-a genera un mesaj de eroare, daca coexiunea s-a putut realiza sau nu. Pentru a functiona aplicatia inainte de a o porni trebuie deschis Xampp si la Apache trebuie apasat butonul de start si bineinteles si la MySql trebuie apasat pe butonul de start. Dup ace s-a efectuat acest pas putem cu usurinta sa ne deschidem baza noastra de date. In baza mea de date exista 4 tabele: unul pentru customer, unul pentru product,unul pentru order si unul pentru Order\_product.Vom prezenta mai jos cateva etape de pornire a aplicatiei MySql si vom prezenta de asemenea un mesaj de eroare.



Baza noastra de date este create in Cristina. Mai jos o sa afisez o poza cu baza de date.

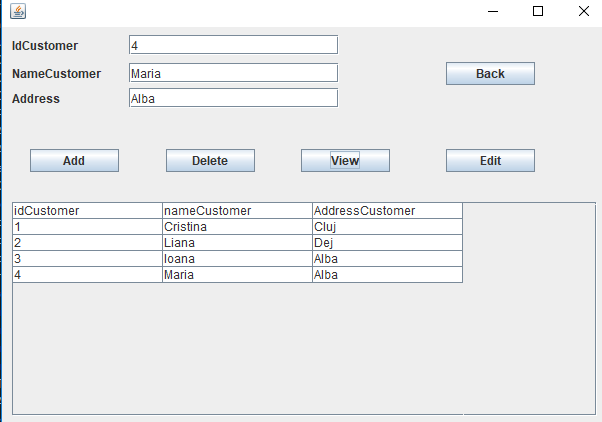


## 2.3. Scenarii

In aceasta sectiune vom prezenta cateva secvente din ceea ce face interfata mea grafica. Vom adauga un client, vom adauga un produs dupa care vom face comanda. Putem da o multime de exemple pentru a ne evidentia toate posibilitatile de testare a acestei aplicatii.

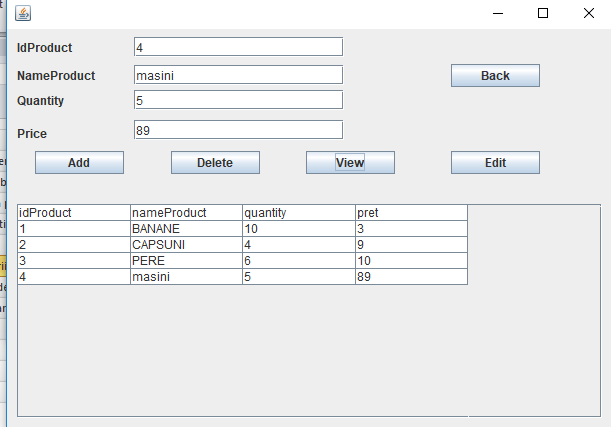
Pentru a putea observa cu usurinta modificarile care apar pe parcurs, trebuie sa dam un refresh la table in care realizez modificarile necesare. Acest lucru il poate observa utilizatorul cu usurinta.

1.Adaugare client nou in baza de date:



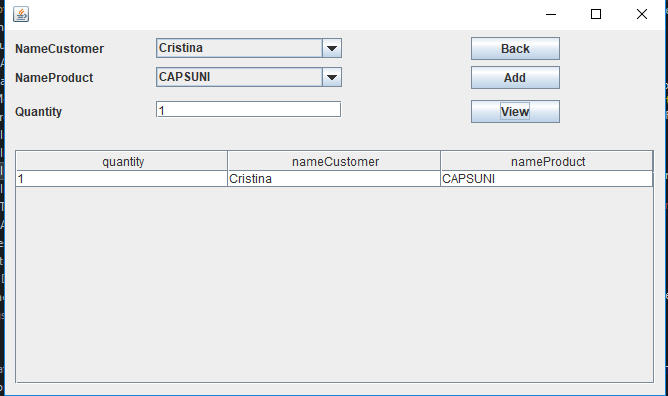
Daca o sa apasam pe butonul de Delte si vom scrie in JTextFieldul de la idCustomer se v-a sterge clientul care are id-ul respective. Butonul de View este pentru a vizualiza in JTable ceea ce avem noi in baza de date. Pentru crearea de JTable am folosit reflectia.

2. Adaugare produs nou:



In momentul in care introducem un idProduct egal cu un idProduct deja existent in baza noastra de date se va genera un mesaj de eroare.Pentru a putea vedea cate produse avem in baza de date trebuie sa apasam butonul de View. La fel ca si la customer avem un buton de Adaugare si anume Add , buton de stergere , buton de View si buton de Edit. Butonul de Edit ne permite sa modificam un anumit produs in functie de Id-ul pe care il introduct eu de la tastatura.

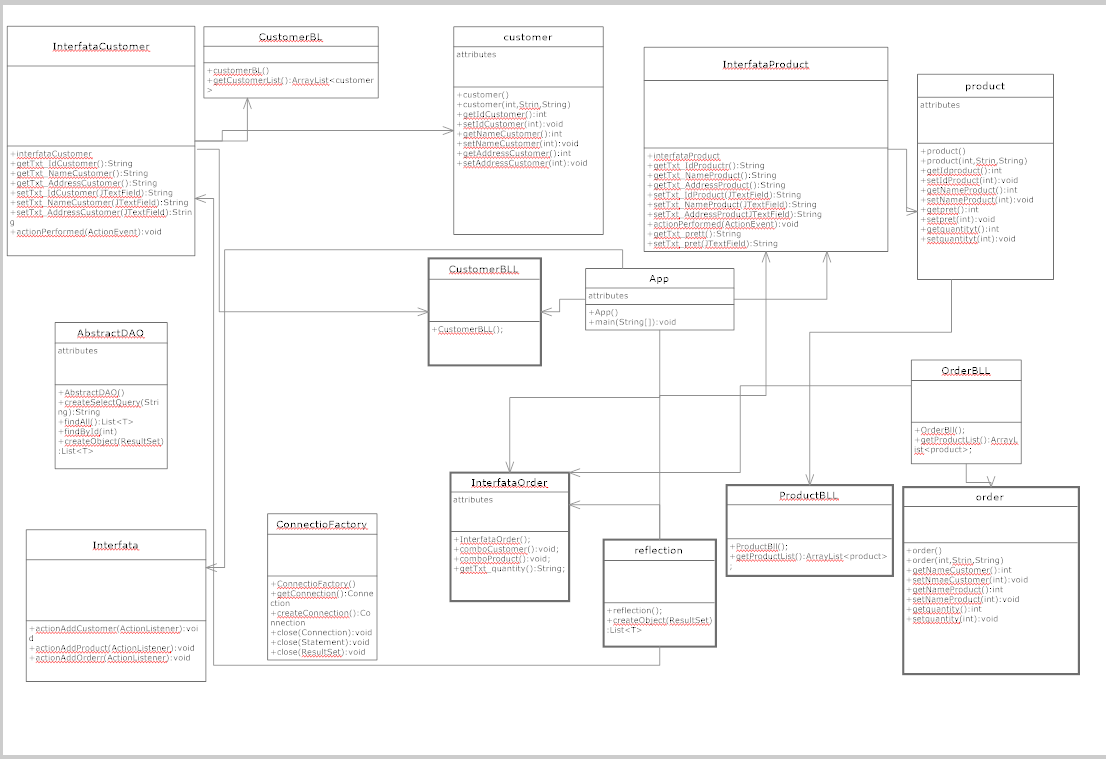
3.Adaugare comanda nou:



La campul NameCustomer avem numele tututot clientiilor din baza de date, la NameProduct avem numele tututot produselor din baza de date, iar Quantity reprezinta cantitatea pe care doresc sa o cumpar eu.Daca in baza de date in tabelul product nu am suficienta cantitate ne va genera un mesaj cum ca nu exista suficienta cantitate. In momentul in care apas pe View imi apare comanda pe care am facut-o , iar butonul de add imi adauga in baza de date comanda facuta de mine.

# 3. Proiectare(decizii de proiectare, diagram UML, structure de date, proiectare clase, interfete, relatii, packages, algoritmi, interfata utilizator, modul de tratare a erorilor)

O analiza mai amanuntita asupra aplicatie este prezentata de diagramele UML. UML(Unified Modelling Language) dupa cum ii spune si numele este un limbaj. Acestea sunt folosite pentru reprezentarea desfasurarii unei activitati in rularea programului. Acestea au in componenta urmatoarele elemente: numele clasei, atribute si metode. Diagrama UML pentru acest proiect este urmatoarea:



O diagramă conceptuală este o diagramă de structură statică care reprezintă concepte din lumea reală și relațiile dintre ele. Acesta se concentrează pe relațiile și atribute, mai degrabă decât metode și vă ajută să înțelegeți terminologia în zona de domeniu pentru care sunt dezvoltarea unui sistem.

In acest proiect avem 5 pachete:BusinessLayer, DAO, DataAccessLayer, Model, Presentation si PT2018.Tema3. In pachetul BusinessLayer avem clasele CustomerBL, CustomerBll, OrderBll, ProductBll. In clasa CustomerBll se face legatura intre interfete, voi vorbi despre acele interfete cand o sa ajung la apchetul presentation. In pachetul DAO avem clasa AbstractDAO care este folosita pentru reflective. In DataAccessLayer avem clasa ConnectionFactory, care ne realizeaza conectarea la baza de date. In Model avem clasele, order,product si customer, care au ca si attribute ceea ce am in baza mea de date cu exact acelasi nume ca si in baza mea de date, iar pe lanaga aceasta mai am si clasa reflection pentru reflective la JTable.

In pachetul Presentation avem 4 clase pentru interfete, Interfata, InterfataCustomer, InterfataProduct si InterfataOrder. In momentul in care utilizatorul ruleaza aceasta aplicatie v-a aparea un JFrame care contine 3 butoane, un buton pentru Order, unul pentru Customer si unul pentru Product. In momentul in care utilizatorul v-a apasa de exemplu pe butonul de Customer se v-a deschide un nou JFrame care ne permite sa introducem in baza noastra de date un client nou, ne permite sa vedem tot ce avem in baza nostra de date, sa stergem un client din baza de data sis a editam un client din baza noastra de data. In interfata de la product putem adauga un nou produs in baza de date, putem edita acel produs, putem sterge un produs si putem sa vedem tot ce avem in baza de date. In frame-ul de la order se face legatura intre tabelul customer si tabelul product, pentru a putea face un client o comada. Pentru a face comanda acesta are nevoie sa stie daca el se afla in baza de date si daca se afla sa poata vedea si ceea ce se afla in baza de date. In momentul in care apasa pe butonul de Add I se inregistreaza comanda si totodata se creaza un bon in care apare numele clientului care a facut comanda, producul pe care l-a cumparat , cantitatea in care a comanda , dupa care se face un total pentru a vedea cati bani a cheltuit, se inmulteste cantitatea introdusa de client cu pretul produsului care se afla in baza de date.

Mecanismul de reflective ne este oferit de catre pachetul java.lang.reflect. Reflectia inseamna autoexaminare, adica permite determinarea structurii clasei. In limitele managerului de securitate putem afla metodele clasei, constructorii clasei si restul membrilor clasei. Cateodata putem modifica starea obiectului prin apelul metodelor specific sau putem construe obiecte noi. Mecanismul de reflective este utilizat de componentele Java(Java bens)pentru determinarea capabilitatilor obiectelor pe timpul executiei. Membrii clasei sunt atributele si metodele. Metodele se impart la randul lor in doua categorii, constructori si metode obisnuite. Principalele clase definite in acest pachet sunt: java.lang.reflect.Field, java.lang.reflect.Method, java.lang.reflect.Constructor.

Anumite principii ghidează procesul de proiectare a bazei de date. Primul principiu este că informațiile dublate (denumite și date redundante) nu sunt bune, întrucât consumă spațiu și cresc riscul producerii de erori și inconsecvențe. Al doilea principiu este faptul că sunt importante corectitudinea și gradul de completitudine al informațiilor. Dacă baza dvs. de date conține informații incorecte, toate rapoartele care extrag informațiile din baza de date vor conține, de asemenea, informații incorecte. În consecință, toate deciziile pe care le luați și care sunt bazate pe aceste rapoarte vor fi incorecte.

# 4. Implementare

Dupa cum am specificat mai sus exista 5 pachete, fiecare avant clasele proprii. Impartirea pe clase ne ajuta pentru o organizare mult mai buna a codului. Un pachet este o colectie de clase si interfete inrudite. Sunt folosite pentru gasirea si utilizarea mai usoara a claselor, pentru a evita conflictele de nume si pentru a controla accesul la anumite clase. In alte limbaje de programare pachetele se numesc librarii.

Pentru a evita generarea de erori a aplicatie se recomanda sa se urmareasca urmatorii pasi:

1. In momentul in care rulezi aplicatia si apesi pe unul dintre butoane este recomandat ca in momentul in care doresti sa adaugi, sa stergi sau sa editez un produs sau un client sa apesi primadata pe butonul de View pentru a pentru a putea stii ce ai in baza de date. Se recomanda ca toate campurile sa fie completate.

2. Dupa ce ai efectuat pasul de mai sus, inseamna cat u stii cantitatea de produse care se afla in baza de date, deci se recomanda sa nu adaugi o cantitate mult mai mare, deoarece se v-a genera un mesaj prin care te atentioneaza ca nu ai suficienta cantitate.In interfata pentru order poti cu usurinta sa vezi ce client si ce produse ai in vaza de date. Puteti observa cu usurinta comenzile care s-au efectuat pana in momentul in care efectuezi sit u o comanda, pentru a vedea ca comanda s-a realizat cu success apasti pe butonul de View si ultimele linie din tabela reprezinta comanda dumneavoastra. Pretul pentru produs este realizat intr-un fisier de tipul txt.Acesta contine numele clientului, numele produsului, catitatea si pretul total.

# 5. Testare

Pentru aceasta aplicatie nu s-a folosit JUnitTest. S-au introdus cate multe exeple pentru a testa aplicatia. De remarcat faptul ca nu putem avea un client care sa aiba acelasi nume cu un alt client din baza mea de date. Se presupune ca clientul este unic , iar de asemenea acelasi lucru se aplica si pentru produs. Un produs este unic, daca se doreste modificarea unui produs sau a unui client se itroduc noile valori in JTextField si obligatoriu id ul produslui pe care doresc sa il modific eu trebuie sa fie egal cu id-ul produsului respective din baza de date. Aflarea id ului se face fie din baza de date, fie prin apasarea butonului de View.

# 6. Rezultate

La sectiunea 2.3 am introdus screenshot-uri in care avem efectuata fiecare operatie. De remarcat cateva observatii:

In momentul in care rulezi aplicatia si apesi pe unul dintre butoane este recomandat ca in momentul in care doresti sa adaugi, sa stergi sau sa editez un produs sau un client sa apesi primadata pe butonul de View pentru a pentru a putea stii ce ai in baza de date. Se recomanda ca toate campurile sa fie completate.

# 7. Cocluzii

Dupa realizarea acestui proiect am invatat conceptual de reflection si totodata mi-am reamintit cum se scrie cod in baze date si cum se realizeaza conectiunea la baze de date.Am invatat sa utilizez JTable in java sis a lucrez cu toate aceste lucruri cu usurinta.

# 8.Bibliografie

<https://cloud.smartdraw.com/editor.aspx?templateId=95dfa214-2a45-46dd-857e-b2dc7286f844#depoId=9120997&credID=-18736549>

<https://github.com/caseyscarborough/jtables-crud/blob/master/GUI.java>

<https://www.w3schools.com/sql/sql_insert.asp>

<https://docs.oracle.com/javase/tutorial/uiswing/components/table.html>