**DOCUMENTATIE**

**TEMA 4**

NUME STUDENT: Mîrza Andrei Tudor

GRUPA: 30224

# CUPRINS

[1. Obiectivul temei 3](#_Toc95297885)

[2. Analiza problemei, modelare, scenarii, cazuri de utilizare 3](#_Toc95297886)

[3. Proiectare 7](#_Toc95297887)

[4. Implementare 8](#_Toc95297888)

[5. Rezultate 9](#_Toc95297889)

[6. Concluzii 9](#_Toc95297890)

[7. Bibliografie 10](#_Toc95297891)

**1.Obiectivul temei**

Obiectivul principal al acestei teme este cel de creare a unui sistem de management al livrarii de alimente pentru o companie de catering. Aplicatia va avea 3 moduri de functionare. Unul pentru client, care poate sa puna o comanda si sa caute produsele dupa titlu, rating, calorii, pret, grasimi, sodium si proteine. Unul pentru administrator care acesta la randul lui poate sa creeze un produs compus, sa stearga un produs de baza , sa adauge un produs de baza sau sa il modifice. De asemenea un administrator genereaza 4 rapoarte, unul in care se afiseaza cate comenzi au fost plasate intr-un anumit interval de timp, produsele care au fost comandate de mai multe ori, numarul de clienti care au comandat de mai multe ori o comanda a carei pret este mai mare decat o anumita suma introdusa de la tastatura si se mai genereaza produsele comandate pe zile.

Cat despre obiectivele secundare putem spune ca acestea reprezinta pasii care trebuie urmati pentru indeplinirea obiectivului principal si anume:

* Intelegerea cerintelor propuse, analiza problemei
* Proiectarea problemei, scrierea in cod a ideilor de rezolvare
* Testarea aplicatiei

**2.Analiza problemei, modelare, scenarii, cazuri si utilizare**

Analiza problemei – problema este una din viata reala care simuleaza un sistem de management al livrarii de alimente pentru o companie de catering.

Modelarea problemei – este una putin mai complexa intrucat analizand cerinta ne dam seama ca pentru fiecare client , administrator si angajat avem nevoie de cate un cont pentru ca fiecare dintre utilizatori are drepturi diferite.Clientul se logheaza cu un username si o parola, iar in cazul in care acesta nu are deja un cont isi poate crea un cont nou, unde poate sa vizualizeze liste cu produsele de baza , lista cu produsele compuse, iar in cazul in care acesta nu gaseste un anumit produs il poate cauta dupa rating, titlu, numarul de calorii, numarul de grasimi, sodium, proteine sau dupa pret , astfel poate selecta produsele dorite pentru a pune o noua comanda. In momentul in care se creeaza comanda se va afisa lista cu produsele dorite dar si pretul total al cesteia. Administratorul are drepturi diferite de utilizare fata de client si angajat, intrucat pe langa ca poate sa vada produsele de baza si produsele compuse, acesta poate sa modifice un produs compus, sa il stearga sau sa adauge unul nou. La fel si pentru produsele de baza, acesta poate crea unul nou , poate sterge un produs deja existent si poate sa il modifice in functie de preferinta. Pe langa acestea administratorul poate sa genereze patru rapoarte:

* Cate comenzi au fost plasate intre un intre un anumit interval de timp
* Produsele care au fost comandate de mai multe ori
* Clientii care au comandat de mai multe ori o anumita comanda, valoarea comenzii fiind mai mare decat o anumita valoare
* Comenzile plasate intr-o anumita zi si numarul acestora

Cazul in care se modifica un produs de baza in lista de produse:

Utilizare : Administrator

Cazurile de utilizare sunt:

* Administratorul selecteaza produsul care doreste sa fie modificat
* Administratorul introduce titlul, ratingul caloriile, proteinele, grasimile, sodiul si pretul
* Dupa apasa pe butonul “Update product” si produsul va fi updatat conform dorintelor

Cazul in care se adauga un produs nou in lista de produse:

Utilizare: Administrator

Cazurile de utilizare sunt:

* Administratorul introduce noile valori si noul titlu pentru produs in field-urile de titlu, rating, calorii, proteine, grasimi, sodiu si pret
* Administratorul apasa pe butonul “Add Product” si produsul nou va fi adaugat

Cazul in care se sterge un produs din lista de produse:

Utilizare: Administrator

Cazurile de utilizare sunt:

* Administratorul selecteaza din lista de produse care produs va fi sters
* Administratorul apasa pe butonul “Delete Product” si produsul va fi sters

Cazul in care se creeaza un produs compus:

Utilizare: Administrator

Cazurile de utilizare sunt:

* Administratorul selecteaza din lista de produse mai multe produse (cel putin doua) care vor forma un produs compus
* Apoi in field-ul “Compose Product Name” se adauga numele produsului compus
* Iar in momentul in care se apsa pe butonul “Add Composed Product” produsul va fi create si afisat in JList ul alaturat

Cazul in care se sterge un produs compus:

Utilizare: Administrator

Cazurile de utilizare sunt:

* Administratorul selecteaza din lista de produse compuse produsul
* Administratorul apasa pe butonul “Delete Composed Product” si produsul va fi sters

Interfata grafica pentru Administrator:

O imagine care conține text

Descriere generată automat

Cazul in care un client isi creeaza un cont nou:

Utilizator : Client

Cazurile de utilizare sunt:

* Clientul apasa pe butonul “New Account”
* In momentul in care butonul a fost apasat se deschide o noua interfata unde clientul trebuie sa introduca un username si o parola
* Clientul apasa pe butonul “Create” si va fi notificat ca a reusit/nu a reusit sa creeze un cont

O imagine care conține text

Descriere generată automat

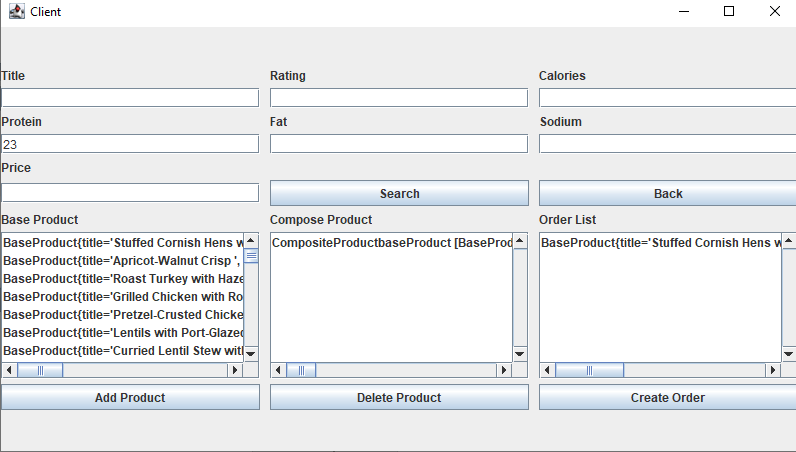
Cazul in care un client pune o comanda:

Utilizator: Client

Cazurile de utilizare sunt:

* Clientul cauta produsul de baza sau un produs compus pe care doreste sa il comande
* In momentul in care acesta gaseste produsul dorit il selecteaza si apasand pe butonul “Add product” produsul va fi adaugat in Jlist ul comenzii
* In cazul in care clientul apasa pe butonul “Delete Product” produsul va fi sters din comanda care urmeaza sa fie pusa
* Daca clientul nu gaseste produsele in lista le poate cauta dupa titlu, rating, calorii, proteine, grasimi, sodiu sau pret

Interfata grafica pentru client:



Cazul in care un angajat se conecteaza:

Utilizator: Angajat

Cazuri de utilizare:

* Angajatul este notificat de fiecare data cand se pune o noua comanda
* Angajatul vede fiecare comanda plasata
* Acesta are un buton de delete, iar in momentul in care butonul este apasat se va sterge comanda

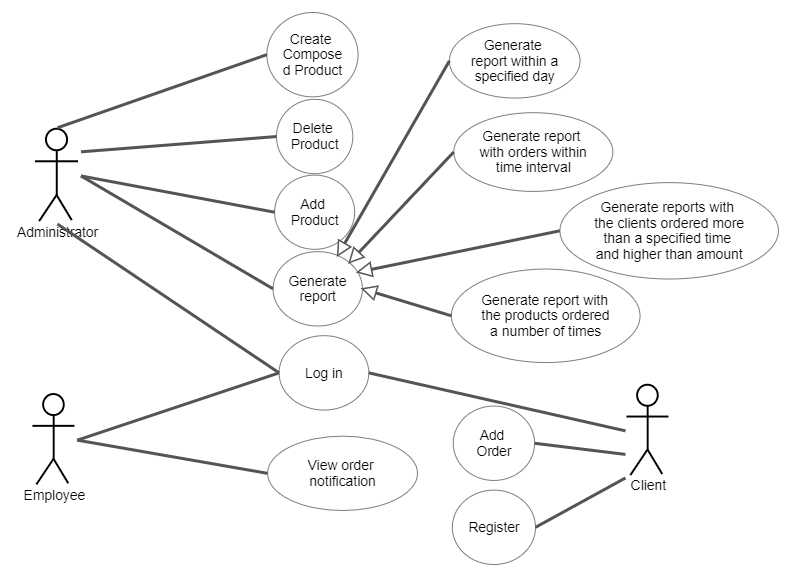
Interfata grafica pentru angajat:

O imagine care conține text

Descriere generată automat

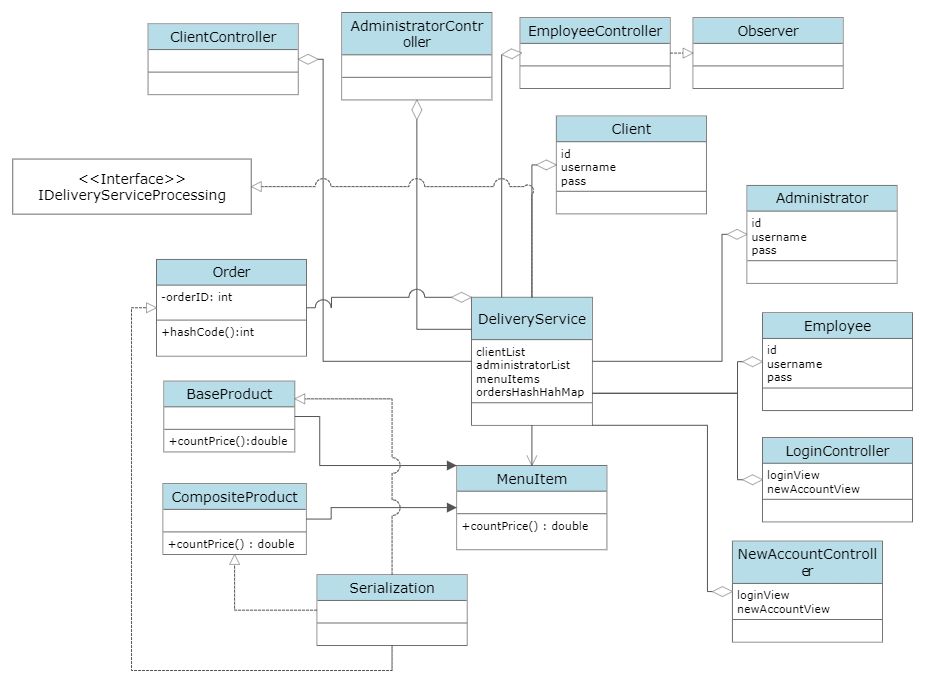
Bineinteles fiecare interfata are cate un buton de back, care in momentul cand este apasat se revine la pagina initiala, de login.

Use case diagram:



**3.Proiectare**

Diagrama claselor:



Aceasta este diagrama de clase si se observa o legatura de model-view-controller.

Fiecare dintre clasele de mai sus este structurata pe pachete si anume:

* BussinesLogic
* DataAccess
* Main
* Model
* Presentation

BusinessLogic contine clasele : BaseProduct, CompositeProduct, DeliveryService, IDeliveryService, MenuItem si order

DataAccess contine clasele : Deserialization si Serialization.

Main contine doar clasa MainClass.

Model contine clasele : Administrator, Client, Employee

Presentation contine clasele : AdministratorController, ClientController, EmployeeController, LoginController, NewAccountController, plus clasele de view.

Fiecare dintre aceste clase are anumite metode pe care am sa le prezint in cele ce urmeaza.

**4. Implementare**

Asa cum am zis mai sus fiecare dintre clasele enumerate are mai multe metode care contribuie la realizarea acestei teme.

1. Clasa BaseProduct:

* In aceasta clasa sunt implementate metodele de set si get pentru rating, title, sodiu, grasimi sip ret
* Pe langa acestea avem o metoda toString() pentru a afisa base products urille in JList

1. Clasa CompositeProduct

* In aceasta clasa sunt implementate metodele de set si get si addCompositeProd() care adauga produsele de tipul base product intr-un set
* Pe langa acestea mai avem o metoda toString() pentru a afisa produsele compuse in JList

1. Clasa DeliveryService:

* In aceasta clasa sunt implementate metodele care importa clientii din fisierul “Client.ser”, in care se face serializarea , la fel pentru administrator , listele de meniu, si comenzi.
* Pe langa aceste metode mai avem metoda de bill() in care se creaza un fisier txt in care se afiseasa ensileile
* Tot in DeliveryService avem metodele care genereaza raportul pentru intervalul de timp , numarul de clienti, numarul de comenzi dintr-o zi si numarul de client care au comandat un anumit numar de comenzi cu un anumit pret

1. Clasa Deserialization:

* In aceasta clasa se face deserializarea
* In metoda deserializationMemCsv() se face deserializarea din fisierul care are extensia .csv care adauga intr-un ArrayList produsul de baza deserializat
* In metoda deserializationMem() se face deserializarea din fisierele cu extensia .ser iar in metoda mapForDeserialization() se face deserializarea intr – un map

1. Clasa Serialization:

* In aceasta clasa se face serializarea
* In metoda mapForSerialization() avem intr-un map obiectele care vor fi serializate
* In metoda serializationMemCsv() avem intr – o lista produsele care urmeaza sa fie serializate din fisierul csv
* In metoda serializationMem() se serializeaza produsele dintr-o lista primita ca parametru

**5.Rezultate**

Rezultatele se pot observa din generarea celor 4 rapoarte puse in fisiere cu extensia .txt . Se genereaza cate un fisier nou pentru fiecare raport. Rapoartele sunt : intervalul de timp in care s-au generat rapoartele, produsele care au fost comandate de un anumit numar de ori, clientii care au comandat ne un anumit numar de ori si valoarea comenzii este mai mare de un anumit pret si produsele comandate intr – o anumita zi.

**6.Concluzii**

Principalele lucruri invatate in acest proiect  au fost creare unor usecase-uri mai putin mai complexe , a unor sequence diagram si o mai buna intelegere a legaturilor dintre obiecte si clase fiind obligati sa respectam anumite reguli. Ca o concluzie, acest proiect m-a ajutat sa imi consolidez cunostintele de programare orientata pe obiect dobandite in primul semestru si sa imi organizez munca in baza paradigmelor POO. Totodata,  felul in care am structurat proiectul si am creat clasele si metodele m-a ajutat sa eficientizez codul din punct de vedere al lungimii si al usururintei intelegerii acestuia, lucruri vitale in cazul programelor complexe care necesita mai multe clase sau care presupun un numar mai mare de programatori.

La capitoloul imbunatatiri si dezvoltari ulterioare intra foarte multe elemente . De exempul se poate imbunatati aspectul ferestrei , se pot adauga alte operatii cum ar fi deschiderea caselor si inchiderea caselor in timpul rularii programului , sau deschiderae respective inchiderea acestora in functie de numarul de client.

**7.Bibliografie**

1. <https://docs.oracle.com/javase/tutorial/uiswing/>

2.  www.stackoverflow.com

3. <https://netbeans.apache.org/kb/docs/java/gui-functionality.html>

4. <https://www.geeksforgeeks.org/serialization-in-java/>

5. https://docs.oracle.com/javase/tutorial/java/javaOO/lambdaexpressions.html