

# **Tehnici de programare in Java**

## **Tema 4**

**Automatica si Calculatoare**

**Calculatoare si tehnologia informatiei**

**Tincu Diana**

**An 2**

**Grupa 30221**

## **1. Obiectivul temei**

**Proiectati si implementati o aplicatie de tip food delivery management system pentru o companie de catering. Clientii pot comanda produse din meniul oferit de companie. Sistemul trebuie sa aiba trei tipuri de utilizatori: administrator, angajat normal si client.**

**Administratorul poate sa faca urmatoarele operatii:**

- **Sa importe un set initial de produse dintr- un fisier .csv care vor popula meniul firmei.**
- **Sa managerieze produsele din meniu adica: sa aduge produse noi in meniu, sa stearga un anumit produs, sa modifice/ editeze un produs din meniu, dar va putea si crea produse compuse noi, formate din alte produse din meniu.**
- **Sa genereze rapoarte despre plasarea comenzilor urmarind criteriile:**
  - **Comenzile plasate intr-un anumit interval de timp: unde se dau ora de start si ora de final.**
  - **Produsele comandate mai mult de un anumit numar de ori specificat**
  - **Clientii care comanda de mai mult de un numar de ori si a caror comenzi depasesc o anumita valoare data**
  - **Produsele comandate intr-o anumita zi specificata de un anumit numar de ori.**

**Clientul poate sa faca urmatoarele operatii:**

- **Sa isi creeze un cont nou, adica sa se inregistreze si apoi sa foloseasca username-ul si parola pentru a se conecta la system cu contul creat.**
- **Sa vada lista de produse din meniu**
- **Sa caute un produs dupa unul sau mai multe criterii: pret, nume, numar de calorii, numar de grasimi, cantitatea de proteine, sodium, sau dupa rating.**

- Sa selecteze unul sau mai multe produse dorite si sa plaseze comanda-  
pentru fiecare comanda se va retine data cand a fost plasata si se va  
genera cate o chitanta care contine lista de produse comandate si pretul  
total al comenzii.

Angajatul primeste cate o notificare la fiecare comanda plasata de catre un client pentru a putea pregati comanda pentru livrare.

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

Pentru implementare m-am folosit de urmatoarea diagrama:

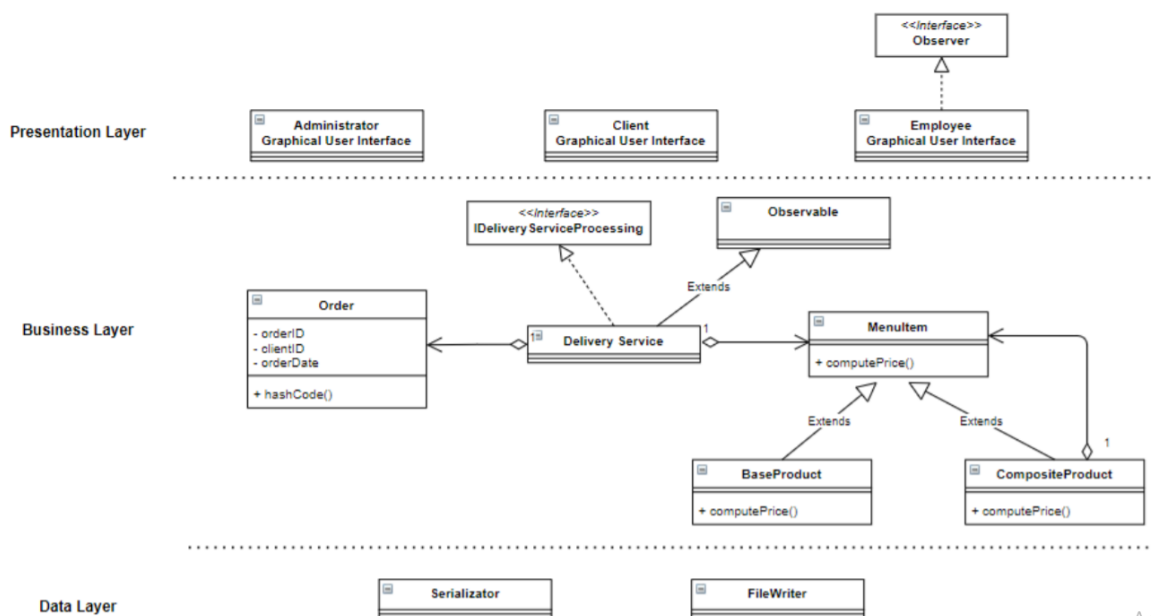


Figure 1: Class diagram to be considered as starting design of the system.

Activ:  
Go to S

Am considerat urmatoarele indicatii:

- 1) Am definit interfata `IDeliveryService` in care am declarat metodele ce urmeaza sa le implementeze clasa `DeliveryService`. Aceste metode sunt metode care executa operatiile care pot fi executate de catre administrator sau de catre client precum: Administratorul poate sa importe produsele din fisierul `.csv` in meniu, sa adauge , sa modifice

sau sa stearga produse din meniu si totodata sa genereze rapoarte. Clientul poate crea noi comenzi ceea ce inseamna ca va trebui sa calculam si pretul total al fiecarei comenzi noi plasate.

- 2) Am definit si am implementat clasele din diagrama de mai sus: Administrator, Order, DeliveryService, BaseProduct, CompositeProduct, MenuItem, User, Client, Employee, Serializator, FileWriter.

Am folosit Composite Design Pattern pentru a define clasele : MenuItem, BaseProduct si CompositeProduct.

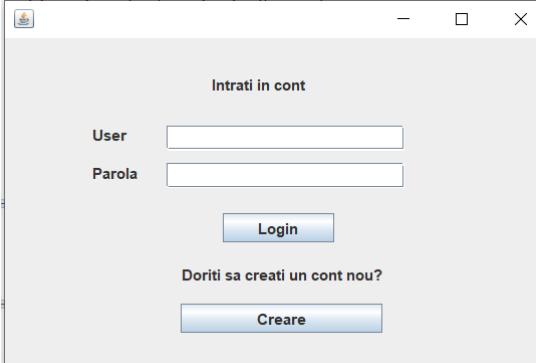
Am folosit Observer Design Pattern pentru a notifica angajatul Employee de fiecare data cand este plasata o comanda noua pentru a putea pregati comanda clientului.

4) Produsele de baza folosite initial pentru popularea obiectelor din DeliveryService pot fi incarcate din fisierul .csv disponibil folosind lambda expressions si stream processing. Nota: Administratorul poate adauga produse manual.

5) Produsele din meniu, comenzile plasate si informatiile utilizatorilor vor fi pastrate folosind serializarea si vor fi extrase folosind deserializarea.

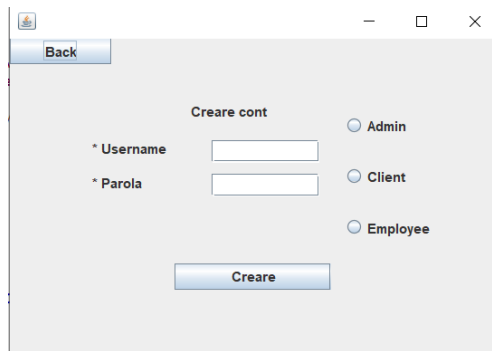
**Mod de utilizare al interfetei:**

Se intra in clasa Login si se da Run. In continuare se va deschide o fereastra de logare in care ne putem conecta la un cont deja existent sau putem crea un alt cont.



The screenshot shows a Java Swing window titled "Intrati in cont". It contains two text input fields labeled "User" and "Parola". Below these fields is a "Login" button. At the bottom of the window, there is a text label "Doriti sa creati un cont nou?" followed by a "Create" button.

Daca se doreste crearea unui nou cont se va selecta butonul Creare si se va deschide o noua fereastra:

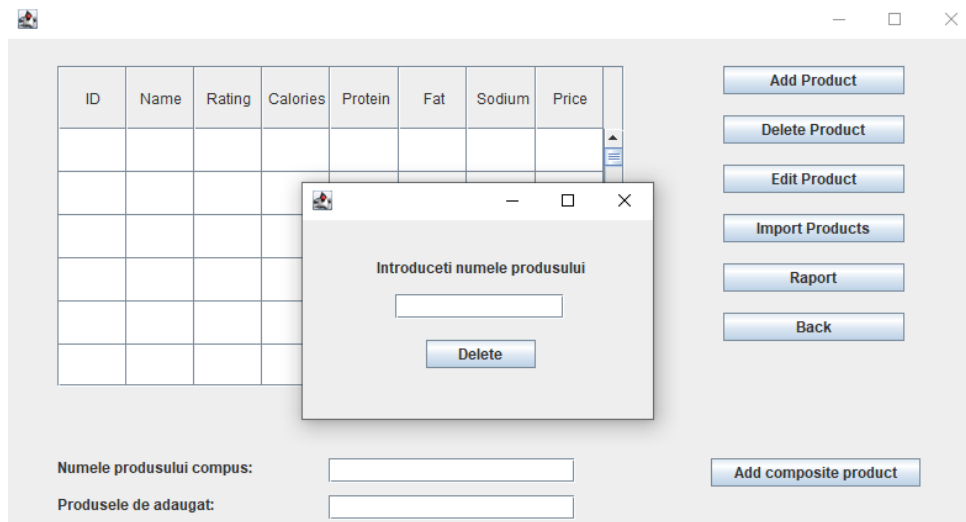


Introducem username-ul si parola dorite in casutele libere si neaparat selectam unul din butoanele alaturate pentru a preciza tipul contului pe care vrem s ail cream. Cand am terminat apasam pe butonul de mai jos pe care scrie Creare si

Adaugam sau importam produse din contul unui administrator folosind butoanele pentru Add Product sau Import Products.

Pentru a sterge produse din meniu apasam butonul Delete Product si introducem in casuta data numele produsului pe care dorim sa il stergem.

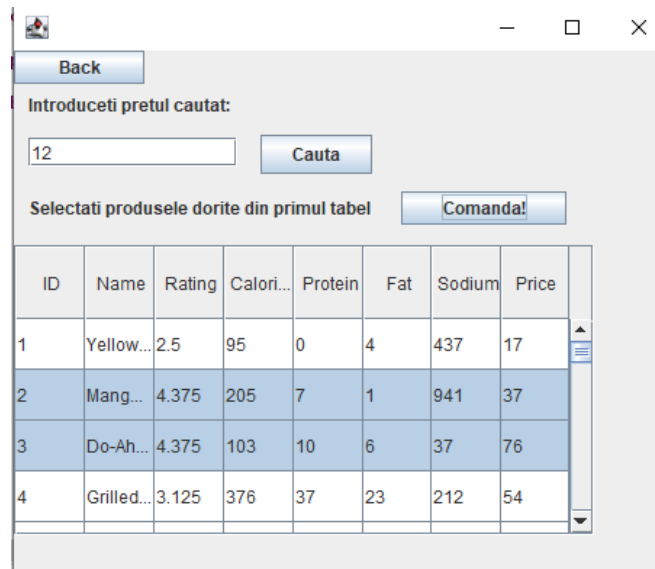
Pentru generarea rapoartelor apasam pe butonul Raport, selectam tipul de



raport dorit si completam casutele unde este cazul.

Pentru a plasa o comanda ne logam cu contul unui client si selectam din tabel produsele dorite dupa care apasam pe butonul Comanda.

Pentru a cauta un produs dupa pret introducem in casuta data pretul dorit si apasam butonul Cauta. Rezultatele cautarii se vor afisa in fisierul SearchResult.txt .



ID	Name	Rating	Calori...	Protein	Fat	Sodium	Price
1	Yellow...	2.5	95	0	4	437	17
2	Mang...	4.375	205	7	1	941	37
3	Do-Ah...	4.375	103	10	6	37	76
4	Grilled...	3.125	376	37	23	212	54

### 3. Proiectare ( decizii de proiectare, diagrame UML, structuri de date, proiectare clase, interfete, relatii, packages, algoritmi, interfata utilizator )

Structuri de date folosite:

Am implementat clasa `DeliveryService` folosind colectii predefinite in Java precum structura de date `HashTable<>`. Cheia folosita pentru hashtable va fi generate pe baza clasei `Order`, si va putea avea mai multe iteme din meniul oferit.

- Am definit o structura de tipul `Map<Order, Collection<MenuItem>>` pentru a stoca informatiile legate de o comanda in clasa `DeliveryService`. Cheia pentru

map va fi formata din obiecte de tipul Order si pentru fiecare dintre aceste comenzi vom mapa la cheia respective Lista de produse comandate in acea comanda.

- Am definit o structura de date de tipul Collection<MenuItem> in care am salvat produsele din meniul oferit de firma de catering. In tema mea am ales structura de date HashSet<> in continuare, pentru a avea produse unice in lista c u produsele din meniu. HashSet este cea mai bună abordare pentru operațiile de căutare.

- Am definit in clasa DeliveryService si o metoda wellFormed

- Am implementat clasa DeliveryService folosind metoda Design by Contract (incluzand pre, post conditii , invariants, si assert- uri).

#### Clasele create :

- BaseProduct ;
- Clientt ;
- CompositeProduct ;
- DeliveryService ;
- MenuItem ;
- Order ;
- User ;
- Users ;
- Serializator ;
- WriteFile ;
- AdaugareProdus ;
- Administrator ;
- Client ;
- CreareCont ;
- DeleteProduct ;
- EditProduct ;
- Employee ;
- GenerareRapoarte ;
- LogIn ;

[illegible]

BaseProduct :

*Clientt :*

Aceasta clasa extinde clasa User si are ca si attribute urmatoarele campuri : nrOrders si comenzi. Unde nrOrders este de tipul int si este initializat cu 0. Atributul nrOrders reprezinta numarul de comenzi plasate de clientul respectiv, acesta se va incrementa cu unu de fiecare data cand clientul va plasa o noua comanda. Atributul comenzi reprezinta o lista a carei structura de date este



urmatoarea: `List<Order> comenzi = new ArrayList<Order>()` . In aceasta lista adaugam fiecare comanda pe care clientul curent o plaseaza.

Aceste attribute sunt declarate ca fiind private deci pentru a putea fi accesate si din alte clase am generat setteri si getteri pentru fiecare dintre acestea.

Am creat si un constructor fara parametrii care apeleaza `super()` pentru clasa parinte `User`.

### CompositeProduct :

Aceasta clasa extinde clasa `MenuItem` si va contine ca si attribute campurile `compPr`, ce reprezinta o lista de produse din meniu care formeaza produsul compus final si este declarat astfel:

```
List<BaseProduct> compPr= new ArrayList<BaseProduct>();
```

Tot in aceasta clasa avem si atributul `name`, care va reprezenta numele produsului compus creat. Acesta este de tipul `String`.

Ambele attribute sunt declarate ca fiind private deci vom genera getteri si setteri pentru a putea fi accesate si din alte clase.

Tot in aceasta clasa am creat si metoda `computePrice()` care calculeaza valoarea finala a unui produs compus prin insumarea preturilor tuturor produselor din lista de produse care il alcatuie. Aceasta metoda returneaza pretul deci asadar o valoare de tip `int`.

### DeliveryService :

Aceasta clasa implementeaza interfata `IDeliveryService` si metodele acesteia. In aceasta metoda am implementat urmatoarele metode:

- `importProducts()` : aceasta metoda importa produsele din fisierul `.csv` in structura de date `baseProducts` de tip `HashSet<BaseProduct>`
- `addProduct(BaseProduct baseProduct)` care adauga in structura `baseProducts` un nou produs.

- deleteProduct(String titlu) care sterge din structura baseProducts care este de tip HashSet<BaseProduct> un produs in cazul in care este gasit, daca nu il gaseste atunci nu se va face nimic, adica baseProducts va ramane la fel. Cautarea si stergerea produsului am facut-o dupa nume.
- modifyProduct() aceasta metoda va cauta tot dupa nume un produs in baseProducts si in cazul in care il va gasi va edita attributele produsului inlocuindu-le cu noile valori transmise ca parametrii metodei.
- Report1(int start, int end) aceasta metoda primeste ca parametrii un interval de timp [ start , end ] si scrie in fisierul Raport1.txt toate comenzile care au fost plasate in acest interval de timp. Start si end reprezinta orele de inceput respective de final.
- Report2() scrie in fisierul Raport2.txt toate produsele care au fost comandate cel putin de un numar specificat de ori. Acest numar este transmis ca parametru metodei.
- Report3() scrie in fisierul Raport3.txt toti clientii care au comandat de un anumit numar specificat de ori si a caror comenzi valoreaza cel putin o anumita suma. Acest numar de comenzi si aceasta suma sunt transmise ca si parametrii metodei.
- Report4() scrie in fisierul Raport4.txt toate comenzile care au fost plasate intr-o anumita zi.
- findProductsByPrice() in aceasta metoda cautam toate produsele din baseProducts care au un anumit pret specificat. Aceasta metoda returneaza un HashSet<BaseProduct> in care se afla toate produsele care au pretul specificat ca si parametru.
- SearchProductByName() cauta un produs dupa nume in baseProducts si daca il gaseste il returneaza. Aceasta metoda o folosesc pentru a verifica daca produsele pe care vreau sa le adaug in CompositeProduct se afla in meniu.
- extractFromCompositeProduct() ia dintr-un String dat ca si parametru toate produsele separate printr-o virgula si le adauga intr-un sir ca mai apoi sa verifice daca exista in meniu, daca exista toate cream un produs compus cu acestea si il returnam.
- addCompProduct() adauga un produs compus in meniu.

- addItem(int id) adauga un produs in lista de produse comandate
- newOrder() are ca si parametru clientul care executa comanda. Aceasta metoda adauga o noua comanda si calculeaza valoarea totala a acesteia. Ea returneaza un string care contine datele despre comanda precum data, si ora comenzii, numele si pretul fiecarui produs comandat dar si valoarea totala a comenzii.

#### MenuItem :

Aceasta clasa contine attributele name, price, id si nrCom si getteri si setteri pentru acestea. Am creat si cativa constructori care sa ne fie utili atunci cand cream obiecte de tipul MenuItem. Aceasta clasa implementeaza interfata Comparable si metoda acesteia compareTo() utila pentru a compara item-urile din meniu.

#### Order :

Aceasta clasa contine datele unei comenzi precum: id-ul comenzii, data comenzii, clientul care a plasat comanda si valoarea totala a comenzii. Pentru fiecare atribut am generat setteri si getteri.

#### User :

Aceasta clasa contine datele unui utilizator precum username-ul, parola, tipul si id-ul.

#### Serializator :

Aceasta clasa implementeaza metode general valabile pentru serializarea si deserializarea obiectelor.

#### WriteFile :

Aceasta clasa implementeaza metode pentru scrierea in fisiere.

#### AdaugareProdus :

In aceasta clasa cream interfata prin intermediul careia administratorul va putea edita un produs in meniu adaugand numele acestuia si restul caracteristicilor. Este obligatorie completarea fiecarui textField!

#### Administrator :

In aceasta clasa se creaza interfata unui administrator prin intermediul careia un administrator poate importa produse din fisierul .csv, poate adauga/edita/sterge produse din meniu, poate genera rapoarte si poate crea produse compuse.

#### Client :

In aceasta clasa se creeaza interfata unui client prin intermediul careia un client poate sa comande sau sa caute anumite produse.

#### CreareCont :

In aceasta clasa cream interfata prin intermediul careia vom putea crea un utilizator nou si ii vom putea allege tipul acestuia, adica daca dorim sa fie client, administrator sau angajat.

#### DeleteProduct :

In aceasta clasa cream interfata prin intermediul careia administratorul va putea sterge din meniu un produs pe care il va identifica prin numele acestuia.

#### EditProduct :

In aceasta clasa cream interfata prin intermediul careia administratorul va putea edita un produs din meniu pe care il va identifica prin numele acestuia.

#### Employee :

In aceasta clasa se creaza interfata pentru angajat.

#### GenerareRapoarte :

Se creaza interfata din care administratorul va putea genera rapoartele dorite.

#### LogIn :

Se creaza interfata pentru LogIn a user-ului.

## **5. Rezultate**

Rezultatele vor fi afisate in fisiere:

- pentru chitanta in fisierul bill.txt,
- pentru rezultatele cautarii unui produs dupa nume in fisierul SearchResult.txt
- pentru raportul 1 in fisierul Raport1.txt
- pentru raportul 2 in fisierul Raport2.txt
- pentru raportul 3 in fisierul Raport3.txt
- pentru raportul 4 in fisierul Raport4.txt

Iar pentru notificarea angajatului am afisat in consola de fiecare data cand angajatul este notificat.

## **6. Concluzii:**

In concluzie am reusit sa implementez o aplicatie de delivery food in care clientii pot comanda produse.

## 7. Bibliografie

<https://stackoverflow.com/questions/8902331/what-is-a-class-invariant-in-java>

<https://www.javatpoint.com/java-hashset>

**Serializare Java:**

[https://www.tutorialspoint.com/java/java\\_serialization.htm](https://www.tutorialspoint.com/java/java_serialization.htm)

**Adding custom tags to javadoc:**

<https://docs.oracle.com/javase/7/docs/technotes/tools/windows/javadoc.html#tag>

**Java assert:**

<https://intellij-support.jetbrains.com/hc/en-us/community/posts/207014815-How-to-enable-assert>

<https://stackoverflow.com/questions/11415160/how-to-enable-the-java-keyword-assert-in-eclipse-program-wise>

<https://javarevisited.blogspot.com/2012/01/what-is-assertion-in-java-java.html#axzz6wYav4UgA>

**Lambda Expressions:**

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