UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA: Automatica si Calculatoare
SPECIALIZAREA: Calculatoare si Tehnologia Informatiei
DISCIPLINA: Tehnici de programare
PROIECT: Sistem de gestiune a comenzilor
ASSIGNMENT NO: 4

Indrumator laborator:
Moldovan Dorin-Vasile

Realizator:
Mihalache Rares

Contents

1. Obiective

Obiectivul principal al acestei teme este de a realiza o aplicatie care sa gestioneze sis a faciliteze comenzile venite de la clienti. In acest exemplu, particular, vorbim despre un restaurant care are un meniu cu mai multe produse. In aceasta aplicatie vom vorbi despre 3 actori: administratorul, clientul si angajatul. Fiecare dintre acestia are anumite caracteristici si operatii pe care le pot desfasura in cadrul aplicatiei.

Obiectivele secundare urmarite in dezvoltarea acestui proiect sunt urmatoarele:

- Lucrul cu fisiere .csv pentru importarea unor informatii care constituie detele de pornire a aplicatiei.
- Folosirea mai multor tipuri de design patterns cu ajutorul carora am reusit sa structurez clasele intr-un mod inteligibil, printre care:
    - Composite Design Pattern
    - Observer Design Pattern
    - Design by Contract
- Folosirea expresiilor lambda pentru a realiza anumite operatii de generare (in cazul administratorului) si de cautare (in cazul clientului).
- Folosirea streamurilor, atat pentru expresii lamda, cat si pentru serializarea datelor in fisiere si deserializarea din fisiere.
- Folosirea preconditiilor si postconditiilor cu ajutorul instructiunii assert pentru testarea codului.
- Realizarea a 3 ferestre pentru cei 3 actori: o fereastra pentru administrator, o fereastra pentru client si o fereastra pentru angajat. Bineinteles, va exista si o fereastra principal, de logare/inregistrare care va fi afisata la pornirea aplicatiei si la care se va putea reveni ulterior.
- Folosirea unei structure de tip Map pentru stocarea unor informatii folositoare clasei “DeliveryService”. In aceastra structura cheia va fi Order-ul si va trebui sa suparscriu metodele “hashCode()” si “equals()”.

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

In momentul in care utilizatorul porneste aplicatia este intampinat de un Frame principal format din doua parti: o parte de Log In si o parte de Register. Pentru cazul in care utilizatorul doreste sa se logheze si introduce niste credentiale gresite, acesta este avertizat sa reintroduca datele (username-ul si parola). Pentru logare exista 3 scenarii diferite: utilizatorul poate fi administrator, client sau angajat, la logarea in sistem fiecare avand 3 ferestre diferite cu operatii utile pentru acestia.

In cazul in care un utilizator nou intra in sistem (un potential client), acesta se poate inregistra, fiind suficient un username si o parola. Utilizatorul care doreste sa se inregistreze in sistem va avea asignat doar permisiunile de client.

Sa presupunem ca se logheaza un administrator in sistem. In acest caz, o fereastra asemanatoare celei de mai jos se va deschide. In aceasta fereastra, administratorul poate alege din mai multe operatii: poate importa produse dintr-un fisier csv, poate sa gestioneze produsele care se afla in restaurant (de exemplu: poate adauga produse, poate sterge produse, poate edita produse sau poate compune produse din alte produse deja existente in sistem). Ultima operatie pe care acesta o poate alege, este cea de generare a raporturilor.

Daca se va apasa pe butonul de “Generate reports” o noua fereastra se va deschide care va contine 4 butoane, cu un text asignat fiecarui buton. Aceste label-uri descriu functionalitatea fiecarui buton (practic, fiecare corespunde unui anumit tip de raport). De exemplu, primul este destinat pentru generarea de produse care au fost comandate intr-un interval orar, indiferent de data. Al doilea genereaza toate produsele care s-au comandat de mai multe ori decat “un numar dat” pe care il introduce administratorul. (acest raport este foarte util pentru cazul in care se doreste sa se vada care sunt cele mai populare si cele mai cerute produse). Al treilea raport numele de utilizator al clientilor care au cumparat cel mai des, si care au avut comenzi in valoare mai mare decat o suma introdusa de administrator. (raport care genereaza cei mai fideli client). Ultimul buton este creat pentru a evidentia produsele vandute in decursul unei zile care este introdusa din sistem.

In cazul in care utilizatorul care s-a logat este un client, se deschide o fereastra asemanatoare celei de mai jos. (Fig. 1) Ca si in cazul administratorului, si clientul poate alege sa efectueze mai multe operatii. Aceste operatii sunt: vizualizarea meniului, cautarea unor produse din meniu pe baza a mai multor cirterii si realizarea unei comenzi. Daca acesta doreste sa iasa din cont, poate apasa oricand pe butonul “Back” pentru a ajunge inapoi la fereastra de pornire.

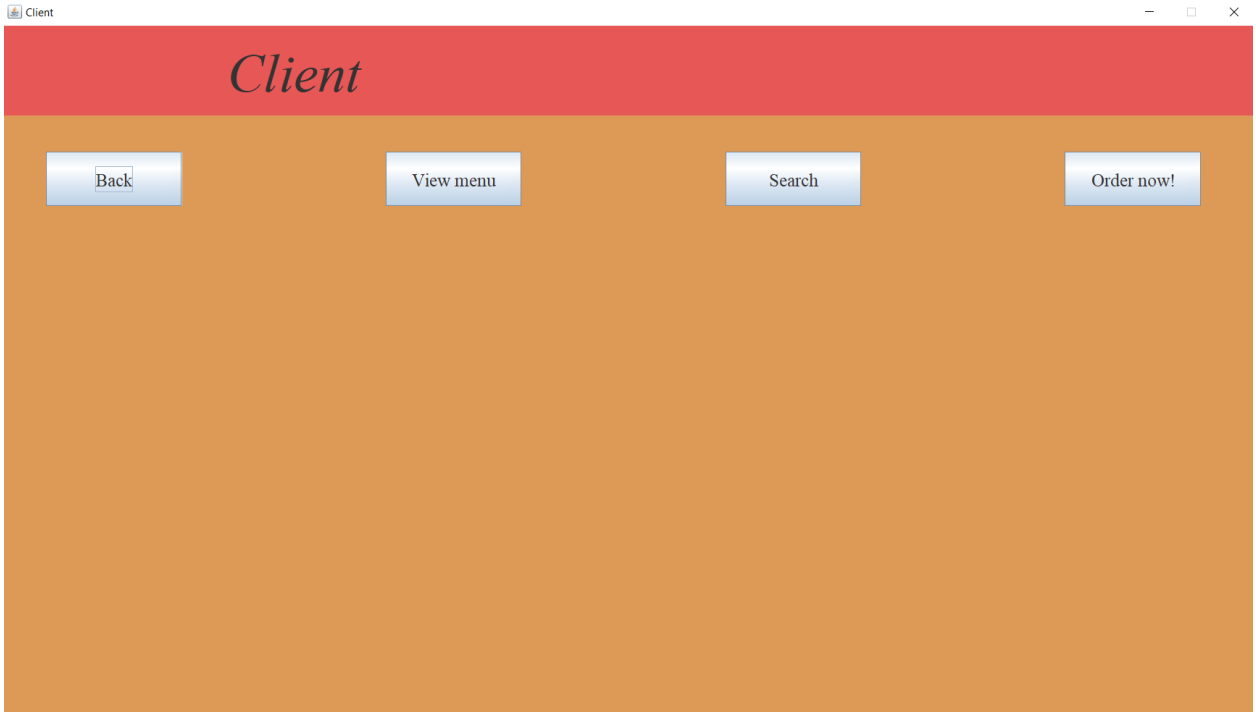


Fig. 1

Cand clientul apasa pe butonul “View menu” se deschide o noua fereastra cu un tabel cu toate produsele din meniu. Fiecare produs este compus din mai multe campuri: titlu (numele produsului), rating, numarul de calorii, numarul de proteine, numarul de grasimi, cantitatea de sodium si pretul. In acest tabel pot exista atat produse simple, cat si produse compuse. Aceste produse compuse se pot observa prin intermediul caracterului “&” din numele produselor.

Daca se apasa pe butonul “Search”, se deschide o fereastra (Fig. 2.1.) cu multe butoane din care se poate alege cate un criteriu, pe rand, pentru gasirea anumitor produse din meniu. De exemplu, pentru apasarea butonului “Search on keyword”, se creeaza un nou Panel cu un buton de “Back”, in cazul in care se doreste sa se revina si sa se selecteze un alt criteriu, fara a inchide aplicatia, un TextField unde introducem cuvantul sau secventa de caracte pe care dorim sa o cautam, un buton de “Search” care se apasa dupa ce se introduce textul in casuta mentionata anterior si un tabel unde se vor afisa datele care corespund filtrarii dorite. Dupa realizarea unei astfel de operatii ar trebui sa rezulte ceva de genul acesta: (Fig. 2.2.)

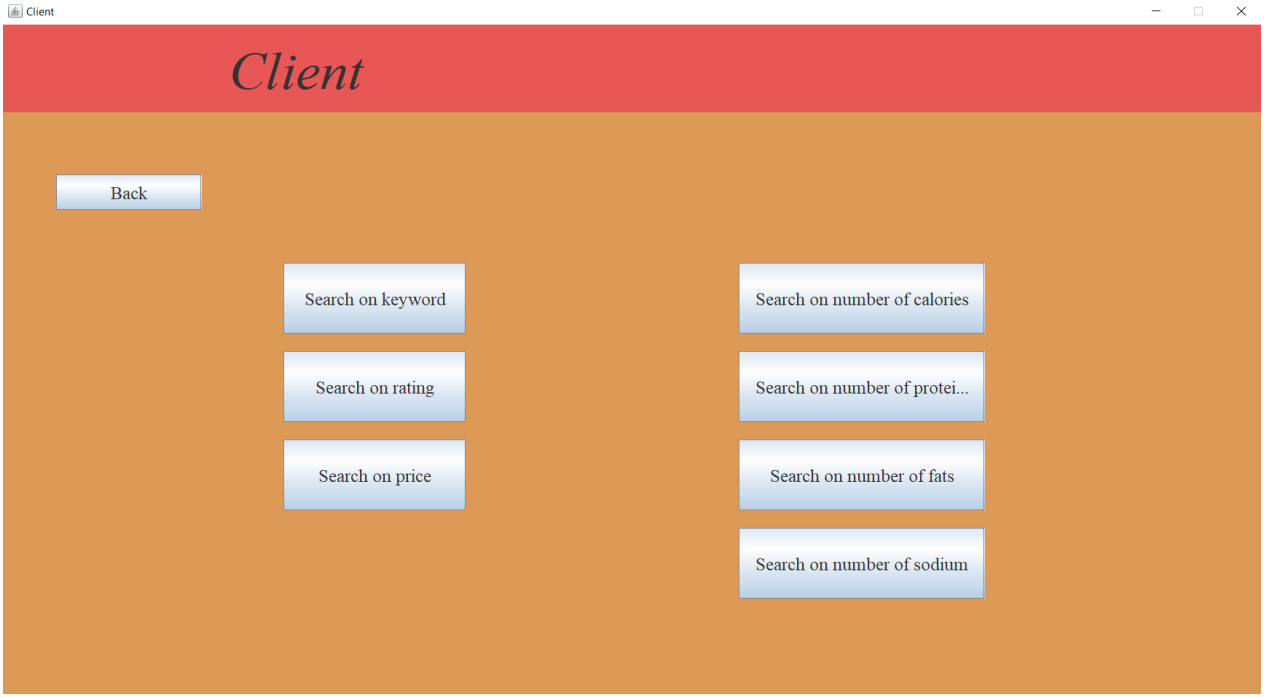


Fig. 2.1.

title	rating	calories	protein	fat	sodium	price
Polenta with Salt Codfish	3.75	198	29	6	805	31
Steamed Catfish in Banana Leaves	3.75	203	14	11	480	83
Catfish in Spicy Tomato Sauce	4.375	217	23	11	777	46
Orange-Scented Bluefish	3.125	227	29	9	85	61
Shellfish Chowder with Fresh Thyme	3.125	235	17	10	422	77
Roasted Monkfish with Fennel-Saffron Compote	4.375	247	32	12	40	45
Swordfish Souvlaki	3.125	300	29	10	285	88
Catfish with Green Olives	3.75	307	26	21	635	47
Roasted Monkfish with Saffron Tomato Sauce and Celeriac Mash	4.375	309	36	13	421	61
Swordfish Tacos	4.375	324	24	21	104	88
Baked Whitefish with Tomatoes and Garlic	3.75	356	31	24	219	67
Baked Swordfish with Olive Relish	4.375	366	34	24	393	86
Grilled Swordfish with Cucumber Lime Salsa	4.375	371	35	17	730	61
Swordfish with Lime-Ginger Glaze	4.375	381	38	12	1031	21
Broiled Bluefish Fillets with Fennel Mayonnaise	4.375	389	46	21	803	54
Grilled Swordfish Verde	4.375	408	45	22	193	19
Spicy Catfish with Bell Peppers and Onions	4.375	421	28	28	173	55
Sauteed Catfish Fillets with Pecan Butter Sauce	4.375	423	35	30	215	64
Broiled Bluefish with Tomato and Herbs	4.375	431	46	25	543	100
Swordfish with Balsamic Brown Butter Sauce	4.375	440	34	31	237	94
Platter of Roasted Shellfish with Trio of Sauces	5.0	449	57	20	2157	83
Catfish with Herbed Lemon Cream	3.75	473	28	31	765	45
Grilled Swordfish with Coriander Lime Butter	4.375	484	41	29	170	33
Shellfish Mixed Grill	5.0	502	85	14	2900	93
Shellfish Chowder	4.375	508	37	21	1885	78
Swordfish Kebabs	4.375	515	34	29	361	26
Whitefish Salad and Roasted Beet Coleslaw	3.75	527	20	18	751	22
Sauteed Swordfish with Niçoise Vinaigrette	4.375	530	4	31	73	75
Pan-Fried Catfish with Arugula-Orange Salad	3.75	540	13	27	23	84
Grilled Swordfish with Rosemary-White Wine Sauce	4.375	541	5	37	64	33
Swordfish Stemperata	4.375	545	27	44	1601	41

Fig. 2.2.

In cazul in care clientul apasa pe butonul “Order Now!” se va astepta introducerea numelor a 3 produse pe care acesta doreste sa le comande. Daca cel putin un produs care a fost introdus nu exista sau are numele specificat gresit, se va afisa un mesaj care va indica acest lucru. Altfel, operatia are loc cu success.

Pentru angajat, se deschide asemanator o fereastra noua, cu un buton de “Back” pentru delogare si cu un textArea unde se vor scrie comenzile, in momentul in care se face una. In acest fel, angajatorul va fi notificat la fiecare noua comanda.

3. Proeictare

3.1. Proiectarea claselor

Clasele au fost proiectate dupa modelul din specificatia temei. Astfel am respectat modelul arhitectural “Layered architecture”, avand 4 pachete:

- businessLayer -> contine functionalitatea de baza a aplicatiei. Exista atat interfete cat si clase abstracte si normale. De exemplu in interfata IDeliveryServiceProcessing, sunt definite operatiile de baza care se realizeaza de catre administrator si de catre client, iar clasa DeliveryService implemeneteaza aceasta interfata, deci implicit si metodele din aceasta. Exista 3 clase (una abstracta – MenuItem si doua normale – BaseProduct si CompositeProduct). Aceste clase nu ofera neaparat foarte multe functionalitati, sunt folosite mai mult pentru getters si setters, dar sunt foarte importante pentru comunicarea cu celelalte clase. Aceste 3 clase sunt implementate dupa modelul “Composite Design Pattern”, in sensul ca “MenuItem” (clasa abstracta) ofera functionalitatile de baza pe care celelalte doua clase care extind “MenuItem” le vor implementa. Clasa “BaseProduct” reprezinta produsul, asa cum apare el in meniul .csv. Dar putem sa realizam produse compuse, acest lucru facandu-se prin intermediul clasei “CompositeProduct” care are drept variabila instanta definitorie o lista de “MenuItem” (care pot fi la randul lor “BaseProduct” sau “CompositeProduct”).
- dataLayer -> are doua clase: “FileWriter” si “Serializator”. Clasa “FileWriter” este folosita (instantiata) in momentul in care clientul doreste sa faca o comanda. In acest caz, cand se apasa pe butonul de “Order Now!”, se creeaza un fisier nou “Bill.txt”, care va contine numele produselor introduse de client si pretul aferent acestora, iar la sfarsit pretul total. Clasa “Serializator” este deosebit de importanta pentru momentul in care dorim sa citim/ sa scriem date din fisierele serializate. Aici exista doua operatii de serializare si de deserializare: “serialize” simplu si “serialize” shared. Varianta “shared” este folosita pentru cazurile in care modificam o anumita resursa care a fost scrisa anterior in fisierul serializabil. Daca nu facem acest lucru si lucram cu structure care sunt imutabile, putem folosi varianta care nu este “shared”. Pentru deserializare, exista la fel, doua variante, aici lucrurile fiind putin diferite.

Una dintre metode returneaza tot fisierul serializat, in timp ce cealalta nu face acest lucru si nu returneaza nimic.

- mainPackage -> contine clasa main prin care se intantiaza fereastra de LogIn. De la LogIn se realizeaza instantierea celorlalte ferestre in functie de ce butoane sunt apasate.
- presentationLayer -> contine tot ce este legat de interfata grafica si de interactiunea pe butoane si faciliteaza transmiterea informatiilor introduce in GUI de utilizator catre celelalte tipuri de “layere inferioare”.

Mai jos este diagrama de la care am plecat: (este prezentata si in specificatia temei): (Fig. 3.)

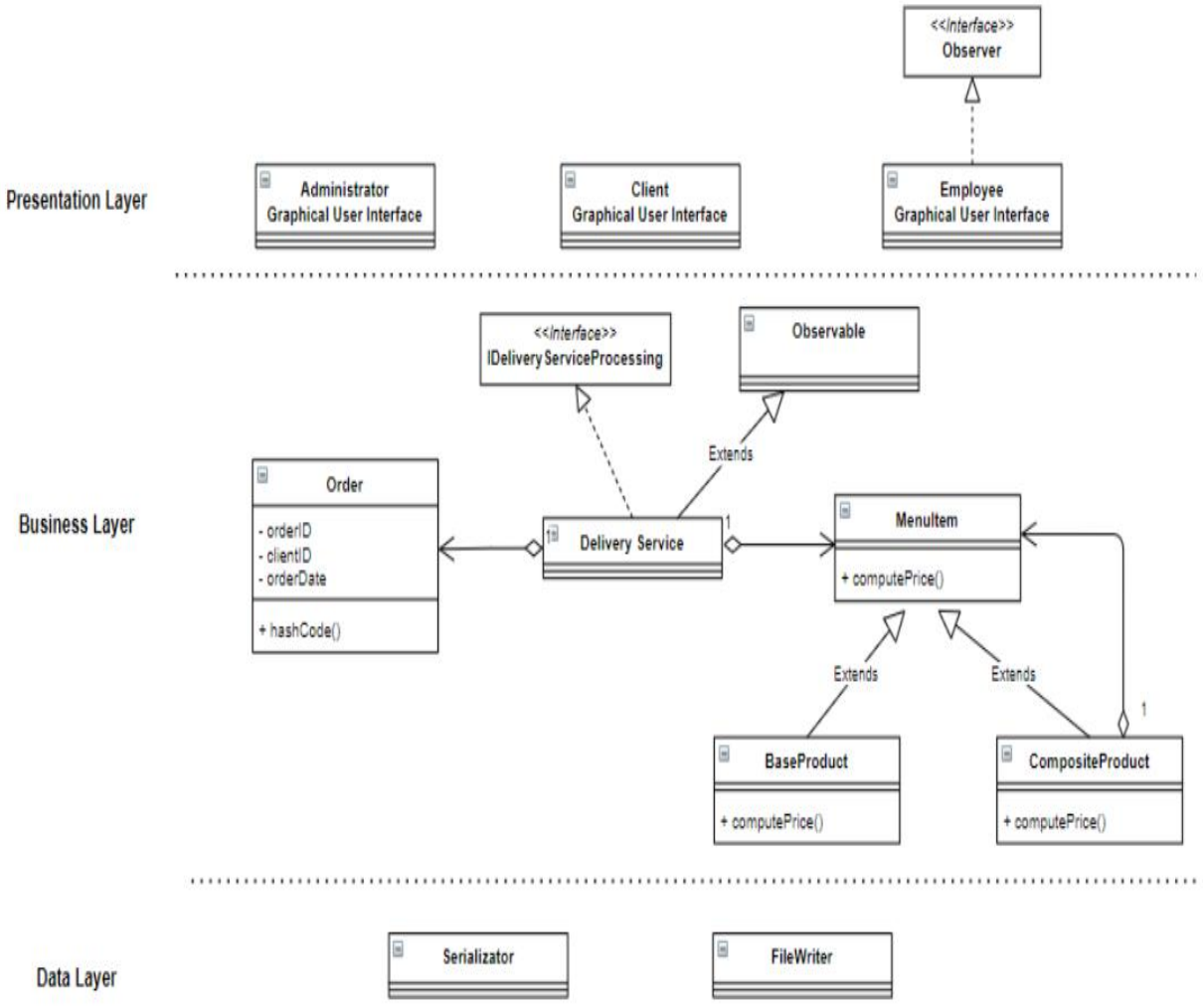


Fig. 3

4. Implementare

In cadrul acestei teme am folosit concepte noi invatate cum ar fi lambda expressions sau procesarea cu streamuri. Aceste notiuni au fost folosite pentru implementarea rapoartelor (in cazul amdinistratorului) si pentru cautarea de produse (in cazul clientului).

Un exemplu de astfel de caz este prezentat in urmatoarea imagine: (Fig. 4)

```
public ArrayList<MenuItem> searchProductByInterval(int min, int max, int mode) throws Exception{
    ArrayList<MenuItem> menuItems = getItemsFromMenu();
    List<MenuItem> toReturn;
    if(mode == 1) {
        toReturn = menuItems.stream()
                .filter(p -> p.getRating() >= min && p.getRating() <= max)
                .collect(Collectors.toList());
```

Fig. 4

Aceasta expresie este folosita pentru toate criteriile de cautare care au o componenta de minim si maxim (toate in afara de criteriul de cautare pe baza cuvantului cheie – keyword). Este o expresie foarte simpla, care filtreaza toate produsele din meniu in functie de rating. Daca rating-ul se afla intr-un anumit interval dat de [min; max], atunci acest produs va contribui la un nou ArrayList<MenuItem> care va fi format doar din acele produse care respecta conditia impusa.

O varianta putin mai complicata (pentru generarea raportului cu numar 2) este urmatoarea: (Fig. 5)

```
public void generateReport2(int amount, JTextArea txtArea) throws Exception {
    List<MenuItem> myOrders = orderedItems(getOrders());
    List<MenuItem> toReturn = myOrders.stream()
            .filter(p -> countFrequencyInList(p, myOrders) > amount)
            .filter(distinctByKey(p -> p.getTitle()))
            .collect(Collectors.toList());

    txtArea.setText("");
    for(int i=0; i< toReturn.size(); i++) {
        txtArea.append("Product #" + i + ": " + toReturn.get(i).getTitle() + "\n");
    }
}
```

Fig. 5

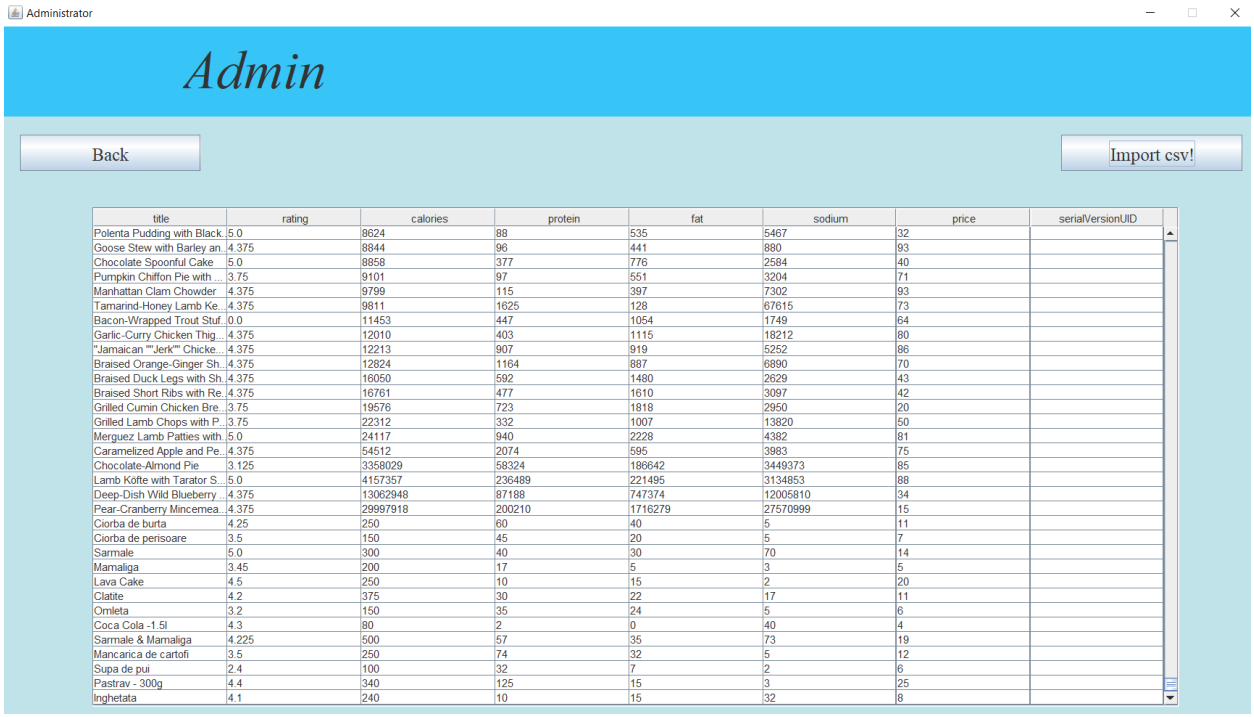
Aici tot ce se intampla este sa se selecteze acele produse care au fost comandate de un numar mai mare de ori decat “un numar dat” de catre utilizator. Acest lucru se face prin utilizarea a doua filtre de cautare. Al doilea filtru cauta sa selecteze un anumit produs doar o singura data, chiar daca se gaseste de mai multe ori in lista de “MenuItem”. Acest filtru foloseste un predicat pe care l-am construit mai jos: (Fig. 6)

```java
public static <T> Predicate<T> distinctByKey(Function <? super T, ?> keyExtractor){

    Map<Object, Boolean> seen = new ConcurrentHashMap<>();
    return t -> seen.putIfAbsent(keyExtractor.apply(t), Boolean.TRUE) == null;
}
```

5. Rezultate

Dupa cele spuse mai sus, aplicatia poate fi utlizata de catrei administrator, client si angajat. Mai jos se prezinta cateva rezultate pentru unele operatii ale administratorilor si clientilor.

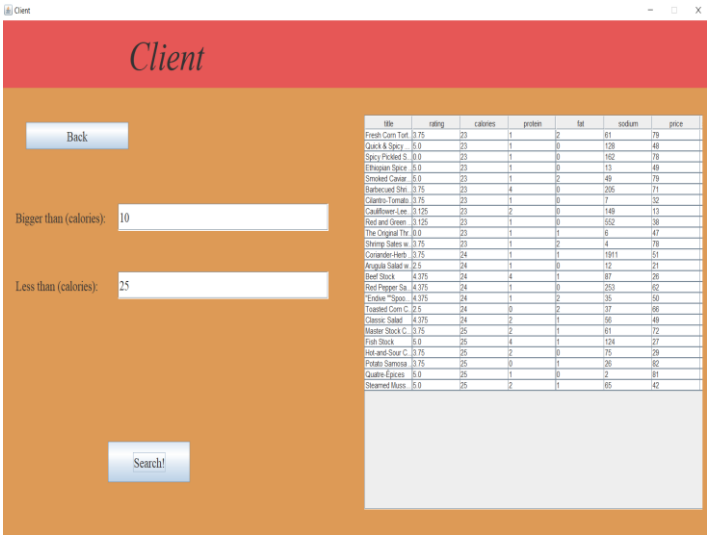


title	rating	calories	protein	fat	sodium	price	serialVersionUID
Polenta Pudding with Black...	5.0	8624	88	535	5467	32	
Goose Stew with Barley an...	4.375	8844	96	441	880	93	
Chocolate Spoonful Cake	5.0	8858	377	776	2584	40	
Pumpkin Chiffon Pie with ...	3.75	9101	97	551	3204	71	
Manhattan Clam Chowder	4.375	9799	115	397	7302	93	
Tamarind-Honey Lamb Ke...	4.375	9811	1625	128	67615	73	
Bacon-Wrapped Trout Stuf...	0.0	11453	447	1054	1749	64	
Garlic-Curry Chicken Thig...	4.375	12010	403	1115	18212	80	
"Jamaican ""Jerk"" Chicke...	4.375	12213	907	919	5252	86	
Braised Orange-Ginger Sh...	4.375	12824	1164	887	6890	70	
Braised Duck Legs with Sh...	4.375	16050	592	1480	2629	43	
Braised Short Ribs with Re...	4.375	16761	477	1610	3097	42	
Grilled Cumin Chicken Bre...	3.75	19576	723	1818	2950	20	
Grilled Lamb Chops with P...	3.75	22312	332	1007	13820	50	
Merguez Lamb Patties with...	5.0	24117	940	2228	4382	81	
Caramelized Apple and Pe...	4.375	54512	2074	595	3983	75	
Chocolate-Almond Pie	3.125	3358029	58324	186642	3449373	85	
Lamb Köfte with Tarator S...	5.0	4157357	236489	221495	3134853	88	
Deep-Dish Wild Blueberry ...	4.375	13062948	87188	747374	12005810	34	
Pear-Cranberry Mincemea...	4.375	29997918	200210	1716279	27570999	15	
Ciorba de burta	4.25	250	60	40	5	11	
Ciorba de perisoare	3.5	150	45	20	5	7	
Sarmale	5.0	300	40	30	70	14	
Mamaliga	3.45	200	17	5	3	5	
Lava Cake	4.5	250	10	15	2	20	
Clatite	4.2	375	30	22	17	11	
Omleta	3.2	150	35	24	5	6	
Coca Cola -1.5l	4.3	80	2	0	40	4	
Sarmale & Mamaliga	4.225	500	57	35	73	19	
Mancarica de cartofi	3.5	250	74	32	5	12	
Supa de pui	2.4	100	32	7	2	6	
Pastrav - 300g	4.4	340	125	15	3	25	
Inghetata	4.1	240	10	15	32	8	

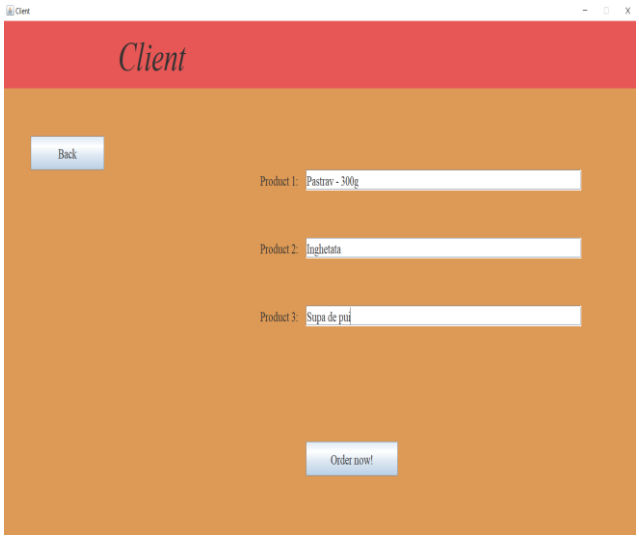
Importarea datelor din fereastra administratorului

Adaugarea unui produs nou (simplu)

Adaugarea unui produs nou (compus)



Afisarea produselor dupa calorii



Adaugarea unei comenzi noi

6. Concluzii

Aceasta tema de laborator m-a invatat multe lucruri pe care nu le stiam inainte despre serializarea datelor in fisier, despre lambda expressions si interfete functionale. Am invatat concept noi si “design patterns” de care nu stiam inainte, cat si utilizarea preconditiilor si a postconditiilor.

Desigur, multe lucruri puteau sa fie facute mai bine si codul putea fi scris mai usor. Ca si dezvoltari ulterioare, cred ca ar fi foarte folositor sa implementez cautarea produselor astfel incat sa se poata selecta mai multe filtre simultan. Momentan se poate selecta doar un filtru si este destul de limitative, mai ales daca nu iti mai aduci aminte exact ce caracteristici avea un produs.

Un alt lucru care poate fi schimbat ar fi interfata grafica si modul in care sunt prezentate rapoartele administratorului. Nu este chiar cea mai user-friendly modalitate modalitate, cea cu textArea, si in plus nu am pus ScrollPane, astfel ca daca sunt foarte multe date intr-un raport, administratorul nu va putea sa le vada pe toate. La asta se mai poate lucra ulterior.

Cu toate acestea, sunt de parere ca aplicatia, pana in acest stadiu este destul de buna si foloseste conceptele prezentate in specificatia temei. Proiectul este functional si pana acum nu am gasit nicio eroare grava care sa imi ofere rezultate neasteptate.

7. Bibliografie

- http://tutorials.jenkov.com/java/lambda-expressions.html [Java lamda expressions]
- https://docs.oracle.com/javase/tutorial/jndi/objects/serial.html [Serializable Interface]
- https://docs.oracle.com/cd/E19683-01/806-7930/assert-13/index.html [Precondtions and Postconditions]
- https://docs.oracle.com/javase/tutorial/collections/interfaces/map.html [Java Map]
- https://en.wikipedia.org/wiki/Observer_pattern#:~:text=The%20observer%20pattern%20is%20a,calling%20one%20of%20their%20methods. [Observer pattern]
- https://www.geeksforgeeks.org/composite-design-pattern/ [Composite pattern]
- https://docs.oracle.com/javase/8/docs/technotes/guides/language/assert.html [Java assert]
- https://www.oracle.com/technical-resources/articles/java/ma14-java-se-8-streams.html [Java stream processing]
- https://docs.oracle.com/javase/7/docs/technotes/tools/windows/javadoc.html#tag [Java -Adding custom tags to Javadoc]