

FACULTATEA: Automatică și Calculatoare

SPECIALIZAREA: Calculatoare și Tehnologia Informației

DISCIPLINA: Tehnici de programare fundamentală

PROIECT: Sistem managerial pentru un restaurant

**Buțiu Cristian**

**An universitar:2020-2021**

**Grupa: 30227**

**Profesor laborator: Dan Mitrea**

**Cuprins:**

1. **Obiectivul temei**
2. **Analiza problemei, modelare, scenarii, cazuri de utilizare**
3. **Proiectare**
4. **Implementare**
5. **Rezultate**
6. **Concluzii**
7. **Bibliografie**
8. **Obiectivul temei**

Obiectivul acestei teme a fost implementarea unui sistem managerial pentru un restaurant. Aceasta tema a avut drept ajutor o diagrama de clase care sa ne indrume in rezolvarea acestei probleme. Rezolvarea acestei teme a necesitat lucrul cu interfete grafice pentru toate operatiile facute:atat de admin - adaugare, editare, stergere de produse,crearea produselor compuse(formate din mai multe produse simple, de exemplu: daily menu1 este format dintr-o friptura, o ciorba de cartofi, garnitura de orez si desert tiramisu), importarea de date din csv, generarea raporturilor,cat si de client – vizualizarea meniului, cautarea dupa 1 sau mai multe criterii a unui produs, plasarea unei comenzi.

* Folosirea standardului Java de numire a pachetelor, claselor, metodelor, atributelor.
* Folosirea conceptelor de bază a Programării Orientate pe Obiect(moștenire, încapsulare, etc.).
* Folosirea expresiilor lambda si procesarea cu streamuri pentru generarea raporturilor specifice administratorului si pentru cautarea produselor facuta de client.
* Folosirea stream - urilor pentru citirea datelor din fisierul products.csv

1. **Analiza problemei, modelare, scenarii, cazuri de utilizare** Analiza unei probleme reprezintă cel mai important pas către rezolvarea acesteia. Acest pas presupune o examinare amănunțiă și o identificare a unei soluții optime pentru problema dată. Ideiile la nivel conceptual trebuie transformate într-o formă finală dorită, adica codul Java.

Modelarea datelor reprezinta si ea un aspect important pe care nu trebuie sa il omitem. Deoarece tot ce ne furnizeaza interfata grafica este de tip String vom face cast, dupa caz, la int, double, time sau date.

In implementarea proiectului au fost folosite mai multe straturi(layers), fiecare avand o responsabilitate bine definita, astfel incat cele 2 pachete sunt:

-Presentation Layer, acesta contine clasa ViewandController unde avem interfata grafica pentru administrator si pentru client dar si controlul actiunilor acestora.

-Bussines Layer, acest strat contine clasele de baza, model, MenuItem, BaseProduct, CompositeProduct, DeliveryService cu interfata IDDeliveryServiceProcessing, Order.

Scenarii

Se ruleaza programul, se va deschide o fereastra pentru log in unde utilizatorul trebuie sa furnizeze username-ul si parola, si dupa caz va fi logat in fereastra cu operatiile pentru admin sau fereastra cu operatiile pentru client. Pentru o functionare corecta prima data admin-ul trebuie sa insereze produsele in meniu, iar apoi se pot folosi si celelalte functionalitati.

1. **Proiectare**

Proiectarea claselor este asemanatoare cu cea a modelului dat,exceptand mici exceptii la structa straturilor.

Pachetul Business Layer:

1.Clasa BaseProduct extinde clasa MenuItem si reprezinta un produs simplu cu atributele String title, float rating, int calories, int protein, int fat, int sodium, int price. Ca metode avem gettere si settere pentru toate atributele si metoda computePrice().

2.Clasa CompositeProduct extinde clasa MenuItem si reprezinta un produs compus format din mai multe produse simple(BaseProduct). Are ca atribute un ArrayList<MenuItem> si cele mentionate in clasa BaseProduct. Ca metode avem constructorul, metoda getLista care returneaza List<MenuItem> si metoda suprascrisa computePrice().

3.Clasa abstracta MenuItem reprezinta un produs din meniul restaurantului. Produsul poate fi simplu sau compus.

4.Interfata IDDeliveryServiceProcessing – aceasta interfata este implementata de catre clasa DeliveryService si are metodele care vor fi implementate in DeliveryService.

5.Clasa DeliveryService implementeaza interfata IDDeliveryService si are ca attribute List<MenuItem> unde se tin produsele meniului, HashTable<Order, List<MenuItem>> unde se tin informatiile legate de comenzi. Ca metode avem importProducts() , addProduct(MenuItem product), deleteProduct(MenuItem product), modifyProduct(String title, MenuItem product) , compoundProduct(String newTitle, ArrayList<Menuitem> listaProduse), createNewOrder(Order order, ArrayList<MenuITem> listaOrder), productOrdered(int nrOfTimes), timeInterval(LocalTime startHour, LocalTime endHour), searchingForProducts(String keyword, double rating, int calories, int protein, int fat, int sodium, int price), getListaMenu().

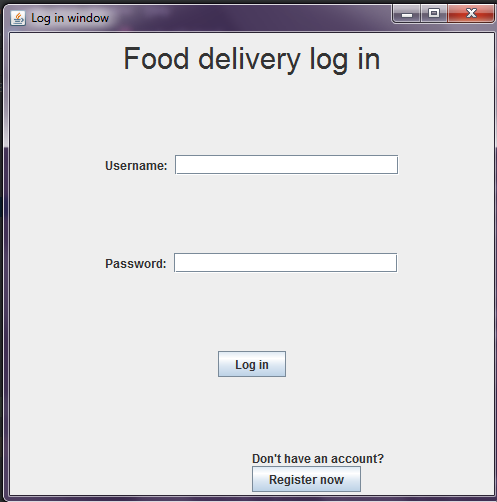
6.Clasa Order care simuleaza o comanda. Aceasta clasa are ca attribute int OrderID, String clientID, LocalDateTime dataAndHour. Ca metode avem gettere si settere pentur atribute si un override la metoda hashCode();

Pachetul PresentationLayer:

Aici se afla o singura clasa ViewandController care tine toata interfata programului si tot ce tine de controlul programului. Aici avem doar un constructor.

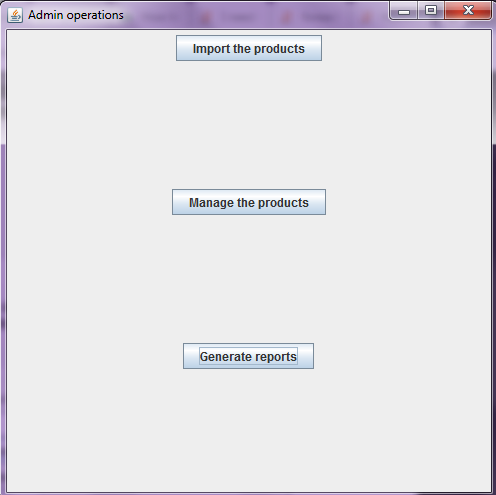
**Interfata grafica**

Aceasta este formata dintr-o fereastra principala de log in care poate sa deschida alte 2 ferestre(cate 1 pe rand) pentru tipul de utilizator ce foloseste aplicatia.

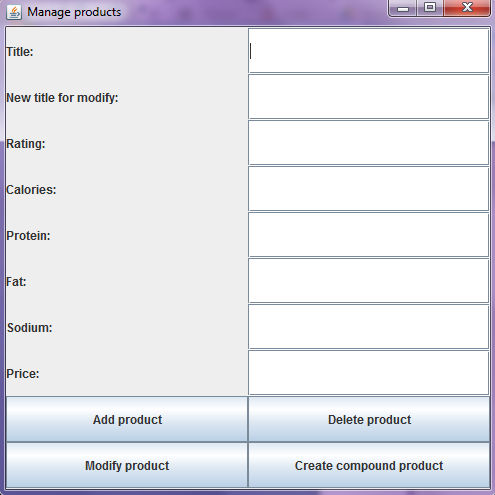


Fereastra principala

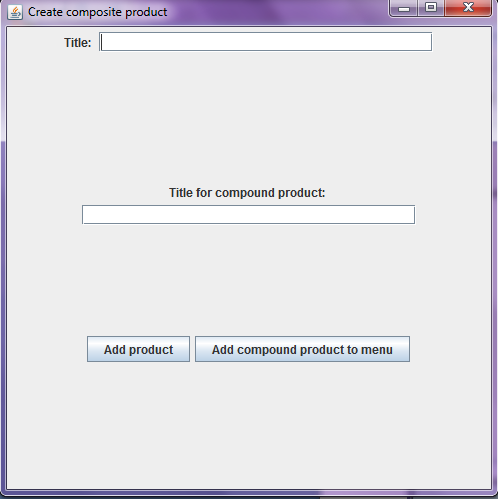
Aici se va loga uitilizatorul si va fi redirectionat catre una din ferestrele de admin sau de client in functie de rolul acestuia.

Fereastra admin

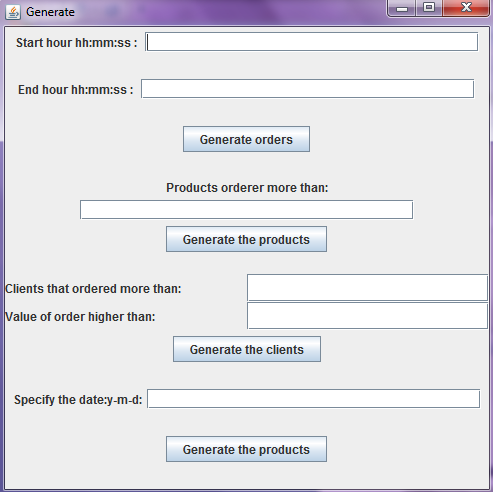
Pentru a adauga elemente din products.csv in meniu se va apasa butonul „Import the products”. Pentru a genera raporturile aferente se va apasa „Generate reports”, iar pentru a manageria produsele se va apasa butonul „Manage the products” care va deschide o alta fereastra.

Fereastra manage the products

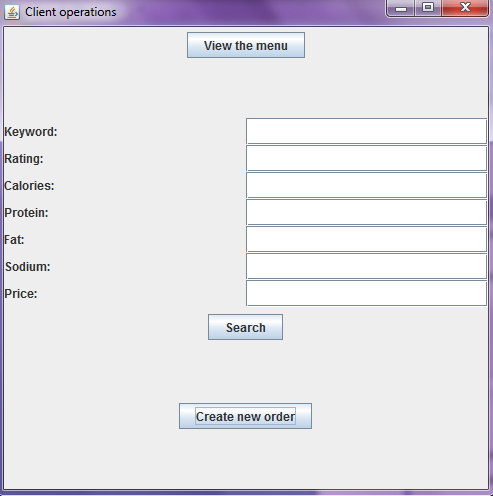
De aici se adauga un nou produs la meniu apasand butonul „Add product”. Daca se doreste stergerea unui produs se va completa campul Title si se va apasa „Delete product”. Daca adminul doreste sa modifice un produs va introduce titlu produsului, noul titlu in caz ca vrea sa il schimbe si restul campurilor pe care doreste sa le schimbe si va apasa „Modify product”.Apasarea butonului „Create compound product” va deschide o noua fereastra.

Fereastra compositeproduct

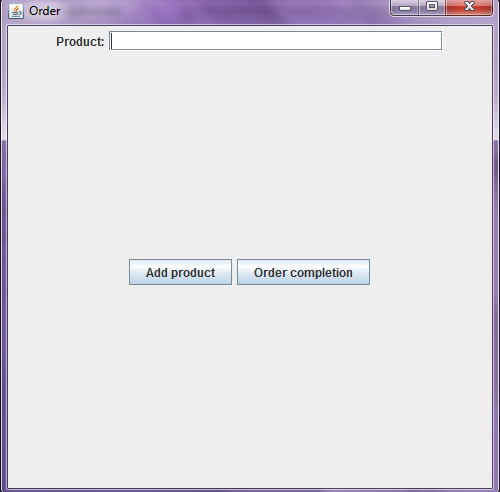
Aici admin-ul va introduce titlul produsului pe care doreste sa il adauge la produsul compus si va apasa „Add product”. Dupa ce a adaugat produsele dorite admin-ul seteaza un titlu pentru produsul compus si va apasa „Add compound product to menu” care va adauga produsul compus la meniu.

Fereastra generate reports

In fereastra de generate reports se fac generarile aferente fiecarei cautari.Avem 4 generari de facut fiecare generare va afisa lista obtinuta.

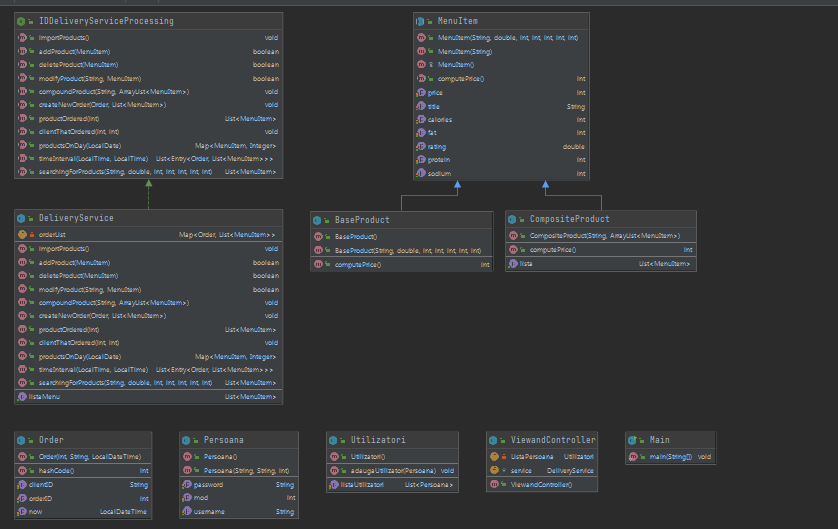
Fereastra de clienti

Daca clientul va apasa pe butonul „View the menu” va vedea produsele din meniu. Daca acesta doreste sa caute un produs il poate cauta dupa un keyword, rating, calories, etc(1 sau mai multe criterii). Daca clientul doreste sa efectueze o comanda va apasa „Create new order” si se va deschide o noua fereastra. Metoda de search va genera toate produsele care corespund cu valorile date de catre utilizator.

Fereastra comanda

Aici clientul introduce numele produsului dorit si apasa „Add product”. Dupa ce a terminat de introdus produsele dorite clientul va apasa „Order completion”, iar chitanta ii va fi afisata in fisierul Orders.txt.

Diagrama UML a proiectului se regaseste mai jos



1. **Implementare**

* Clasa BaseProduct

Aceasta clasa extinde clasa MenuItem si are 7 atribute, String title, double rating, int calories, int protein, int fat, int sodium, int price. Avem un constructor gol si un constructor care seteaza toate aceste atribute.Sunt implementate atat metodele de get si de set cat si metoda suprascrisa compoundPrice(), aceasta metoda returneaza pretul produsului.

* Clasa CompositeProduct

Aceasta clasa extinde clasa MenuItem si implementeaza metoda computePrice()care returneaza pretul pentru produsul compus. Aceasta clasa are si un constructor care seteaza numele produsului compus si lista de produse simple din care acesta este format.

* Clasa Order implementeaza metodele de get si set pentru atributele sale OrderID, ClientID si dataAndHour dar si metoda suprascrisa hashCode()
* Clasa abstracta Menuitem are un constructor gol un constructor care seteaza toate atributele sale si un constructor care seteaza titlul. Aceasta clasa implementeaza si metodele de get si set pentru atributele sale si mai are si metoda abstracta computePrice() care returneaza un int.
* Interfata IDDeliveryServiceProcessing contine metodele ce urmeaza a fi implementate in clasa ce implementeaza interfata.
* Clasa DeliveryService are 2 atribute unul de tip List<MenuItem> unde se tin produsele si unul de tip HashTable<Order, List<MenuItem>> unde se tin informatiile legate de comenzile create de clienti. Metoda importProducts() nu are parametrii si adauga lista de produse in primul atribut al clasei folosind stream – uri. Parcurgem fisierul products.csv linie cu linie si extragem intr-o lista de tipul List<List<String>> valorile primite. Trebuie sa avem grija sa sarim peste prima linie unde se afla numele coloanelor. Metoda addProduct primeste ca parametru un obiect de tipul MenuItem si il adauga la lista de produse deja existente. Atentie! Pentru a nu avea duplicate trebuie sa verificam la fiecare inserare daca nu cumva produsul pe care incercam sa il adaugam nu este in meniu. Daca este nu il vom insera si vom instiinta utilizatorul ca produsul pe care incearca sa il introduca se afla deja in meniu. Metoda deleteProduct primeste ca parametru tot un obiect de tipul MenuItem pe care incearca sa il stearga. Parcurgem cu un for lista de produse si verificam daca gasim produsul cautat, daca acesta se afla in meniu il vom sterge, daca nu vom afisa utilizatorului un mesaj ca produsul nu se afla in meniu.

Metoda modifyProduct() primeste ca parametrii un String title si un MenuItem product. Deoarece cautarea se face dupa nume trimitem un String suplimentar in caz ca admin - ul vrea sa ii schimbe numele. Luam pe rand produsele din lista iar daca numele cautat este gasit vom seta unde este cazul noi valori pentru atribute. Daca utilizatorul nu furnizeaza in una sau mai multe casute o valoare noua de setat, atunci product.acelAtribut va lua valoarea -1. Asa vom stii ca acel atribut nu trebuie modificat. Metoda compoundProduct primeste ca parametrii un String newTitle reprezentand titlul pentru produsul compus si un ArrayList<MenuItem> reprezentand lista de produse simple care compune acel produs. Vom crea un obiect de tipul CompositeProduct si il vom adauga la lista de produse. Metoda createNewOrder primeste ca parametrii Order order si ArrayList<MenuItem> si adauga in lista cu informatii despre comenzi acesti parametrii. Metoda timeInterval primeste 2 parametrii de tip LocalTime si genereaza folosind streamuri lista de comenzi care a fost plasata in intervalul orar startHour -> endHour.

Metoda SearchingForProducts() primeste 7 parametrii – String keyword, double rating, int calories, int protein, int fat, int sodium, int rating. Cautarea se poate face dupa 1 sau mai multe criterii(de exemplu dupa un cuvant cheie si calorii).Lista filtrata va fi afisata intr-o noua fereastra.Aceasta metoda se foloseste de streamuri pentru a cauta produsele care corespund cu cerintele specificate.Daca utilizatorul nu specifica un textField campul acestuia va lua valoarea -1 pentru a nu cauta dupa acel criteriu. Metoda getListaMenu() nu are niciun parametru si returneaza lista de produse din meniu.Metoda productOrdered primeste ca parametru un int nrOfTimes reprezentand numarul minim de aparatii al produsului in lista ce comenzi. Aceasta metoda genereaza toate produsele care au aparut in lista de comenzi de minimum nrOfTimes ori.Lista va fi afisata in consola. Metoda clientThatOrdered() primeste ca parametrii 2 intregi int nrOfTimes si int value. Metoda afiseaza acei clienti care au comandat de un numar minim de nrOfTimes si comanda facuta a avut pretul de minim int value. Metoda productsOnDay primeste ca si parametru LocalDate date si genereaza toate produsele comandate in acea zi si de cate ori a fost comandat fiecare produs.

* Clasa ViewandController

Aceasta clasa are 2 atribute de tipul Persoana si de tipul DeliveryService si reprezinta controlul si interfata programului. ViewandController are doar constructorul fara parametrii in care se fac toate operatiile necesare ca aceasta aplicatie sa functioneze optim. Metodele din Delivery service vor fi apelate la fiecare apasare de buton si vor furniza metodelor valorile pe care utilizatorul le va introduce in casutele de text din interfata grafica.Chitanta fiecarui utilizator este generata in aceasta clasa folosindu-ne de un FileWriter si metoda .write().

* Clasa Main are metoda statica main in care avem instantiat un obiect de tipul ViewandController.

**5.Rezultate**

Proiectul a fost supus unei testari amanuntite pe tot parcursul implementarii acestuia.M-am gandit la multe cazuri care ar putea provoca erori si am incercat sa le remediez prin cele mai bune metode pe care le-am putut implementa la momentul respectiv. Programul poate sa aiba erori pe anumite date de intrare, erori care mi-au scapat.

**6.Concluzii**

Chiar daca am rezolvat tema aceasta cu o saptamana intarziere, aceasta tema aratandu-si dificultatea si punerea la incercare pentru a o rezolva. Am invatat multe din aceasta tema, concepte noi (stream - urile), dar si manuirea mai buna a interfetelor grafice si o scriere de cod mai cursiva. Am realizat faptul ca pentru a aborda o problema ai nevoie de o analiza amanuntita, concreta si un plan bine pus la punct pentru a o rezolva. De – a lungul implementarii solutiei mele, am realizat ca o aplicatie nu este niciodata perfecta, tot timpul mai poate fi imbunatatit cate ceva, si trebuie sa treci prin tot felul de dificultati si perioade lungi de gandire asupra metodologiei aplicate. Te confrunti cu cazuri speciale si erori nemaivazute pana acum pentru a duce la bun sfarsit acest proiect. Ca dezvoltari ulterioare, ar fi

**7.Bibliografie**

<https://stackoverflow.com/questions/18604408/convert-java-string-to-time-not-date><https://www.oracle.com/technical-resources/articles/java/ma14-java-se-8-streams.html> <https://winterbe.com/posts/2014/07/31/java8-stream-tutorial-examples/> <https://howtodoinjava.com/java8/java-stream-distinct-examples/> <https://www.geeksforgeeks.org/how-to-convert-a-stream-into-a-map-in-java/>

<https://www.javacodeexamples.com/get-all-values-of-hashtable-in-java-example/3140>