

Documentatie

Tema 4

SISTEM DE GESTIONARE DE

LIVRARI DE ALIMENTE

Nume: Marinescu Roxana-Maria

Grupa: 30227

**Obiectivul temei:**

Cerința temei este de a crea o aplicație care sa reprezinte un sistem de gestionare a livrarilor de alimente dintr -un restaurant. Aplicația are ca fereastră principala, o fereastră de logare in care utilizatorul va introduce username-ul si parola, in funcție de rolul pe care îl indeplineste utilizatorul în fișierul user.txt (administrator, angajat, client). Dacă utilizatorul are rolul de administrator, acesta poate sa importe produsele aflate in fisierul products.csv, poate sa adauge produse noi in meniu, să modifice produsele, sa le șteargă si să creeze un produs nou care sa conțină base product -uri. Administratorul poate sa accese 4 rapoarte legate de livrari. Dacă utilizatorul este client, el poate să vadă meniul cu toate produsele disponibile si poate sa caute in lista in lista de produse, dupa unul sau mai multe criterii. Atunci cand alege produsele pe care le doreste, clientul le poate adauga în cărucior, i se va calcula pretul total, iar aceste va putea plasa o comanda. Comanda este transmisa catre angajat, iar atunci cand angajatul se logeaza va putea vedea notificarea legata de comanda.

Obiectivele secundare ale temei sunt:

* Intelegerea corectă a cerinței și crearea unor scenarii de utilizare.
* Proiectarea unei soluții care să respecte toate cazurile introduse de user.
* Implementarea soluției și a interfeței grafice.
* Rezultatele finale și concluziile.

**Analiza problemei, modelare, scenarii, cazuri de utilizare:**

Scenarii de utilizare:

La deschiderea aplicației user -ul trebuie să urmeze următorii pași :

* Logare sau Inregistrare
* Folosirea interfetei grafice in functie de rol
* Iesire

*Log:*

--User -ul introduce un username si o parola, iar in functie de rolul pe care il indeplineste se deschide fereastra corespunzatoare

-- Username-ul este unic

-- Daca utilizatorul nu are un cont creat, acesta poate sa isi creeze unul (numai in cazul clientilor).

*Administrator:*

--Poate sa importe produse din fisierul products.csv

--Poate sa adauge un produs nou

--Poate sa modifice field -urile oricărui produs

--Poate sa stearga un produs

--Poate sa creeze un produs compus din mai multe produse

--Poate sa genereze rapoarte legate de anumite interese statistice

*Client:*

--Poate sa vada lista cu produsele

--Poate sa dea search la produse in functie de unul sau mai multe criterii

--Poate sa selecteze produsele pe care doreste sa le cumpere si sa le adauge in cos

--Poate sa vada un pret total al produselor pe care le a adaugat

--Poate sa plaseze comanda

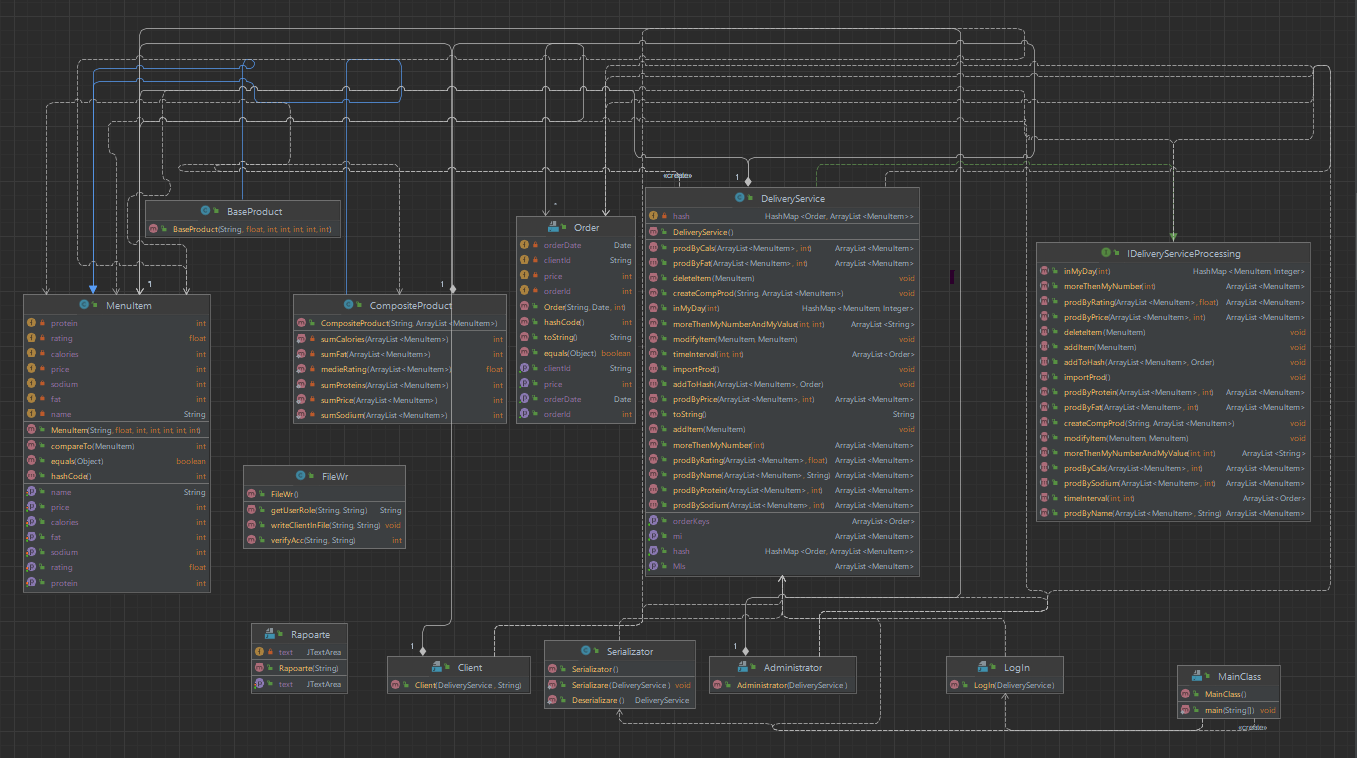
*Angajat:*

--Poate vedea notificarile legate de comenzile clientilor

**Proiectare:**

Am folosit modelul Layered Architecture folosind Business Layer, Presentation Layer, Data Layer.

Am generat și diagrama UML, in care se pot vedea mai amănunțit legăturile dintre clase, metodele folosite cat și structura proiectului.



**Implementare:**

***Clasa BaseProduct:***

Mosteneste clasa MenuItem.

***Clasa CompositeProduct:***

Mosteneste clasa MenuItem. Deoarece e format din mai multe produse, un composite product are proteinele, caloriile, grasimile, continutul de sodiu si pretul egale cu suma tuturor produselor din care e format, iar ratingul e media rating-urilor produselor.

***Clasa DeliveryService:***

DeliveryService implementeaza interfata IDeliveryServiceProcessing si contine metode pentru search (dupa nume, proteine, calorii, rating, grasimi, sodiu, pret), metode pentru cele trei rapoarte legate de comenzile plasate cat si metodele de import, adaugare, stergere, modificare si creare de produse.

***Clasa MenuItem:***

MenuItem contine ca variabile instanta : numele produsului, ratingul produsului, caloriile produsului, proteinele produsului, grasimile produsului, continutul de sodiu si pretul produsului. Clasa este mostenita de clasele BaseProduct si CompositeProduct.

***Clasa Order:***

Clasa Order contine ca variabile instanta : id-ul comenzii, numele clientului, data la care a fost plasata comanda si pretul comenzii.

***Clasa FileWr:***

In clasa FileWr am trei metode: getUserRole (are scopul de a identifica daca user-ul introdus este admin, client sau angajat; cu ajutorul metodei de poate identifica ce fereastra se va deschide dupa ce are loc logarea) , writeClientInFile (are scopul de a scrie clientul in fisierul user.txt in momentul in care acesta isi creeaza un cont) si metoda verifyAcc (care are rolul de a verifica daca persoana are deja un cont atunci cand vrea sa se logheze si sa afiseze un mesaj care sa il anunte pe user ca are deja cont sau sa il anunte ca a fost adaugat in cazul in care acesta chiar nu avea cont).

***Clasa Serializator:***

Clasa Serializator are doua metode, una de serializare si una de deserializare.

***Clasa Administrator:***

Clasa administrator reprezinta interfata grafica din care lucreaza adminul. Adminul poate sa importe, stearga, modifice, adauge un produs, sa creeze un composite product prin selectarea de produse din tabelul de produse din gui. Acesta poate genera rapoarte legate de comenzile plasate de clienti (referitoare la zile, ore, cantitati ).

***Clasa Client:***

Clasa Client reprezinta interfata grafica pusa la dispozitia clientului. Clientul poate sa caute un produs dupa toate criteriile produsului (poate folosi unul sau mai multe filtre de search). Poate accesa si tabelul cu produse folosind butonul de meniu. Ca sa adauge produse in cos acesta trebuie sa selecteze produsele din tabel. Se va genera automat pretul total pe masura ce adauga produse in cos. Urmeaza ca acesta sa apese pe butonul de order si se presupune ca acesta a finalizat comanda.

***Clasa LogIn:***

Clasa LogIn este frame-ul principal, folosit in scopul de a deschide una dintre cele trei ferestre (pentru client, admin sau angajat) in functie de rolul pe care il are (in conditiile in care acesta are un cont). Butonul de signIn e folosit doar de clienti, in cazul in care nu au un cont existent, cu scopul de a se înregistra.

***Clasa Rapoarte:***

Clasa Rapoarte este formata dintr-un textarea in care se vor afisa detalii despre comanda si clienti, in functie de care dintre rapoarte se doreste a se vedea.

***Clasa MainClass :***

In clasa MainClass am creat si instantiat un obiect de tip LogIn , care reprezinta interfata grafica principala in care user -ul urmeaza sa se conecteze sau sa se inscrie.

**Rezultate:**

După rezolvarea problemelor din cod și ajungerea la varianta finala a proiectului, a urmat sa testez aplicatia introducand diferite date de la tastatura. Am facut mai multe teste logandu-ma ca si cilent si administrator, si am verificat daca functioneaza fiecare operatie din proiect, am testat fuctionarea rapoartelor pentru diferite tipuri de comenzi, introducand o serie de date de la tastatura care sa ma ajute sa observ daca acestea functioneaza sau nu corect. Am creat un fisier user.txt unde pot sa introduc clienti, administratori si angajati.

**Concluzii:**

Dupa testare as spune ca proiectul este destinat utilizarii si ca ar trebui să funcționeze pentru orice caz pe care utilizatorul îl introduce.

Dupa acest proiect am reusit sa inteleg cum functioneaza design patternul Composite Design Pattern. Am reusit sa creez o pagina de logIn si am invatat cum functioneaza stream-urile is lambda expressions. A înțeles cum se utilizeaza serializarea pentru a se salva datele de la o pornire a aplicatiei la alta. A fost un proiect dificil, din punctul meu de vedere, cu o multime de task-uri, mult de lucrat la interfete si la partea de legaturi intre interfata si functionalitate, insa ma bucur ca am reusit sa termin si sa imi însușesc cunostinte.

**Bibliografie:**

*UML Diagram:* [*https://youtu.be/v8G7FWdiwqs*](https://youtu.be/v8G7FWdiwqs)

Presentation: <https://dsrl.eu/courses/pt/materials/A4_Support_Presentation.pdf>

Lambda expressions and stream processing:

<https://www.oracle.com/technical-resources/articles/java/ma14-java-se-8-streams.html>

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

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