DOCUMENTAȚIE

Restaurant

Chișluca Miruna

Grupa 30222

Contents

[1. Obiective. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .3](#_Toc6531299)

[2. Analiza problemei. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4](#_Toc6531300)

[3. Proiectare. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5](#_Toc6531301)

[4. Implementare. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6](#_Toc6531302)

[5. Rezultate. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .9](#_Toc6531303)

[6. Concluzii. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .9](#_Toc6531304)

[7. Bibliografie. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9](#_Toc6531305)

# *Obiective*

Obiectivul principal al temei reprezinta proiectarea si implementarea unei aplicatii pentru administrarea unui restaurant.

Pentru a indeplini acest obiectiv, se are in vedere atingerea unor alte obiective secundare precum folosirea unor design patterns pentru functionarea corecta a aplicatiei.

# *.Analiza problemei*

Aplicatia proiectata trebuie sa implementeze functionarea unui restaurant. Astfel, exista 3 tipuri de utilizatori si anume administrator, chelner si bucatar.

Fiecare dintre cei 3 utilizatori mentionati mai sus are de efecuat anumite lucruri.

Administratorul trebuie sa vizualizeze meniul restaurantului si poate sa adauge produse in acest meniu, sa le stearga, sau sa le modifice. Aceste produse care se afla in meniu pot fi de 2 tipuri. Ele pot fi produse simple, pe care un client le poate comanda in orice combinatie cu alte produse, sau produse compuse din alte produse aflate in meniu. Aici intervine composite desgin pattern-ul.

Chelnerul poate sa creeze o noua comanda, sa calculeze totalul pentru o comanda si sa genereze nota de plata.

Bucatarul are rolul de a efectua comenzile primite de la chelner, cand acesta ia o comanda noua.

Aplicatia functioneaza in urmatorul mod:

1. La pornirea aplicatiei, se deschid 2 ferestre: una pentru administrator si una pentru chelner.
2. Fiecare din aceste ferestre are 3 butoane si o lista afisata.
3. Lista afisata in fereastra administratorului reprezinta meniul restaurantului.
4. In fereastra administratorului exista 3 butoane, fiecare executand unul din lucrurile permise pentru acest tip de angajat.
5. Butonul pentru a adauga un nou produs in meniu va deschide o noua fereastra, in care se alege daca produsul dorit pentru adaugare este un produs simplu, sau un produs format din alte produse. Pentru un produs simplu se deschide o fereastra in care se completeaza numele produsului si pretul acestuia, dupa care se adauga in meniu prin apasarea butonului „OK”. Pentru un produs alcatuit din alte produse aflate deja in meniu, se deschide o fereastra care contine o lista cu produsele aflate deja in meniu. Se selecteaza produsele care intra in alcatuirea noului produs de adaugat, dupa care se apasa butonul „OK”
6. Butonul pentru a sterge un produs din meniu va deschise o noua fereastra care contine o lista cu toate produsele existente in meniu, se selecteaza produsul dorit pentru stergere si se apasa butonul „OK”
7. Butonul pentru a edita un produs existent in meniu va deschide o fereastra care contine o lista cu toate produsele din meniu, se selecteaza produsul pentru modificat iar in urma apasarii butonului „OK” se deschide o noua ferestra in care se completeaza datele pentru modificare.
8. In fereastra chelnerului exista butoane pentru adaugarea unei noi comenzi, calcularea totalului si generarea bonului si o lista care contine informatii despre comenzi.
9. Butonul pentru adaugarea unei noi comenzi va deschide o fereastra in care se completeaza masa de la care s-a luat comanda, precum si data curenta. Apoi se apasa butonul „OK” si se deschide o noua fereastra cu meniul restaurantului in care se aleg produsele comandate.
10. Butonul pentru nota de plata genereaza un fisier .txt care contine informatii despre comanda: data, produsele comandate si totalul.

# *Proiectare*

Aplicatia a fost proiectata dupa diagrama de clase primita.

Este impartita pe 3 nivele:

- Nivelul de prezentare care reprezinta partea vizuala a aplicatiei. Utilizatorul comunica cu aceasta componenta. Acesta contine clasele care reprezinta interfetele pentru cei 3 utilizatori.

- Business Logic Layer: conține clase care incapsuleaza logica aplicatiei; Aceasta este partea principală a modelului care este independent de GUI.

- Nivelul de date: contine clasele care asigura serializarea (pentru extragerea datelor din fișierul "menu.txt" sau pentru aplicarea modificarilor) si crearea fisierului cu nota de plata.

Pentru a retine comenzile am folosit o structura de tipul HashMap<key, value>, unde key este cheia, reprezentata de un obiect de tipul Order care contine un id, data in care a fost efectuata comanda si masa corespunzatoare comenzii iar value, adica valoarea este reprezentata de o lista de produse din meniu. Pentru a se putea realiza acest hash map, obiectele de tipul Order contin o metoda care calculeaza valoarea la care la fi stocata componenta value, printr-o functie avad ca variabile id-ul si masa.

Pentru a retine produsele din meniu am folosit un ArrayList.

Fiind o aplicatie Java se respecta principiile programarii orientate pe obiect. Asadar s-a folosit incapsularea prin utilizarea modificatorilor private asupra campurilor clasei. De asemenea se foloseste principiul polimorfismului, iar abstractizarea este cheia de modelare a aplicatiei.

# *Implementare*

Aplicatia proiectata contine urmatoarele clase: WaiterGUI, AdministratorGUI, ChefGUI, Order, Restaurant, MenuItem (abstracta), BaseProduct, CompositeProduct.

Pe langa aceste clase, mai contine si o interfata, RestaurantProcessing.

WAITERGUI

Aceasta clasa reprezina interfata grafica pentru chelner. Este formata dintr-o fereastra in care sunt afisate 3 butoane si o lista cu comenzi. Aceasta lista de comenzi este realizata folosind un Jtable, in care datele afisate se obtin din HashMap-ul ce contine orders. Clasa WaiterGUI contine componente swing si actionListener-i pentru butoane.

Pe langa contrustor, aceasta clasa are si niste metode care sunt folosite pentru a afisa datele referitoare la comenzi atunci cand se adauga comenzi noi si metode care returneaza Jtable-urile folosite mai apoi pentru a crea listele care se vor afisa.

ADMINISTRATORGUI

La fel ca si clasa anterioara, aceasta reprezinta interfata grafica pentru administrator. Este alcatuita dintr-o fereastra cu 3 butoane si un JList care reprezinta meniul restaurantului. La fel ca si clasa destinata chelnerului, are metode care returneaza tabele din care se vor crea componentele afisate si o metoda updateMenu() care reafiseaza meniul de fiecare data cand se executa o modificare asupra lui.

ORDER

Clasa Order are 3 atribure: id, care reprezinta id-ul comenzii; date, care reprezinta ziua in care a fost efectuata comanda si table, care reprezinta masa pentru care s-a efectuat aceasta comanda.

Metodele clasei sunt getteri pentru a obtine id-ul, data si masa care se vor folosi in alte functii mai complexe pentru a obtine detalii despre comenzi.

RESTAURANT

Clasa Restaurant este cea mai importanta clasa a aplicatiei. Ea implementeaza interfata RestaurantProcessing si contine un HashMap<Order, ArrayList<MenuItem> denumit orders, care reprezinta comenzile efectuate de restaurant si un ArrayList<MenuItem> denumit menu care reprezinta meniul restaurantului.

Metodele implementate de aceasta clasa sunt operatiile care pot fi efectuate de angajatii restaurantului.

Metoda createNewMenuItem creeaza un nou produs care este adaugat in meniu. Cum produsele car epot fi adaugate in meniu pot fi fie produse simple, fie produse compuse, aceasta metoda primeste date pentru ambele tipuri de produse. In cazul in care al treilea parametru al functiei, si anume o lista de MenuItem nu contine niciun element, se creeaza un nou obiect de tip BaseProduct, care reprezinta un produs simplu si se adauga in meniu. Daca lista contine elemente, atunci se creeaza un obiect de tipul CompositeProduct, adica un produs compus, care se adauga si el in meniu.

Metoda deleteMenuItem sterge din meniu produsul dorit, fie el produs simplu sau produs compus.

Metoda editMenuItem primeste 3 parametri care reprezinta numele produsului si pretul acestuia si produsul din meniu pentru care se doreste modificarea. Pentru ca modificarea unui produs compus ar presupune o logica mult mai complicata, editarea unui produs din meniu se poate face doar pentru produsele simple.

Metoda createNewOrder primeste 3 parametri din care creeaza un obiect de tipul Order si o lista cu produse din meniu corespunzatoare comenzii.

Metoda computePriceForOrder returneaza un float care reprezinta totalul de plata al comenzii respective. Acesta este calculat iterand prin lista de produse si obtinand pretul fiecarui podus, cu ajutorul metodei computePrice din clasele BaseProduct si CompositeProduct.

Metoda generateBill genereaza un fisier .txt in care se scriu datele care apar in mod normal pe un bon fiscal.

MENUITEM

Clasa MenuItem este o clasa abstracta, care contine metoda computePrice si o metoda getName care returneaza un String ce contine numele produsului, pe care am folosit-o pentru a crea datele pentru JTable la afisarea in interfata.

BASEPRODUCT

Clasa aceasta reprezinta produsele simple din meniu si are 2 atribute: numele produsului si pretul acestuia. BaseProduct extinde clasa MenuItem.

Metodele clasei sunt getteri si setteri pentru cele 2 atribute si metoda computePrice care calculeaza pretului produsului. In cazul unui produs simplu, pretul exista deja, nu mai trebuie calculat. O alta metoda a acestei clase este getName care returneaza numele produsului.

COMPOSITEPRODUCT

Clasa aceasta contine un ArrayList<MenuItem> si reprezinta un produs compus. La fel ca si clasa BaseProduct, ea extinde clasa MenuItem. Aceste 3 clase sunt proiectate dupa composite design pattern. Metoda computePrice a acestei clase calculeaza pretul unui produs compus iterand prin lista de produse simple si insumand pretul acestora. Metoda getName functioneaza asemanator, returnand un string cu numele produselor din care este alcatuit, separate prin virgula.

Interfata RestaurantProcessing este implementata de clasa Restaurant si contine declaratiile metodelor. Pe langa cele 6 metode care efectueaza operatiile necesare pentru indeplinirea cerintelor administratorului si chelnerului, mai contine niste metode care usureaza crearea Jtabel-urilor pentru afisarea informatiilor despre comenzi si meniu.

In clasa Restaurant, am folosit pattern-ul design by contract, adica am verificat niste preconditii si postconditii prin instructiuni de assert. Aceste preconditii si postconditii reprezinta validarea operatiilor efectuate, avand rolul de a verifica ca operatiile efectuate au facut exact ce trebuie, nemodificand alte date decat cele necesare.

# *Rezultate*

Aplicatia rezultata este una simpla cu o interfata usor de folosit care reprezinta in linii mari o aplicatie care gestioneaza un restaurant in toata regula, respectand paradigmele OOP amintite mai sus.

# *Concluzii*

Aplicatia este un sistem complex de gestiune a unor comezi si produse dintr-un meniu. Utilizarea fiserului in care scriem si din care citim informatiile despre produs este o modalitate buna de stocare a obiectelor serializabile.

Aplicatia poate avea o multituidine de dezvoltari viitoare. Pentru a securiza accesul asupra celor doua roluri am putea crea o zona de autentificare utilizand un username si o parola. De asemenea s-ar putea crea o modalitate de a semnala finalizarea pregatirii preparatelor de catre chef.

Tema a constituit un bun mod de invatare si aprofundare a celor 2 noi pattern-uri Composite si Observer, de asemenea am folosit si interfata Serializable cu ajutorur careia am scris si citit obiectele in fiser, in format binar.

# *Bibliografie*

<https://dzone.com/articles/layers-standard-enterprise>

<https://stackoverflow.comhttps://docs.oracle.com/javase/7/docs/api/javax/swing/JScrollPane.html>

<https://www.geeksforgeeks.org/for-each-loop-in-java/>

<https://www.tutorialspoint.com/design_pattern/composite_pattern.htm>

<https://www.tutorialspoint.com/design_pattern/observer_pattern.htm>