Food

Delivery Management

Bîrle Silviu-Adrian

Cuprins

1. Obiectivul temei
2. Analiza problemei, modelare, scenarii, cazuri de utilizare
3. Proiectare (decizii de proiectare, diagrame UML, structuri de date, proiectare clase, interfețe , relații, packages, algoritmi, interfață utilizator)
4. Implementare
5. Rezultate
6. Concluzii
7. Bibliografie

# **Obiectivul temei - - - - - - - - - - - - - - - - - - - - - - - - - - -- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -**

# Ca si obiectiv am avut de realizat o aplicatie de food delivery precum Glovo sau FoodPanda. Se foloseste Serializarea fisierelor pentru stocarea produselor , clientilor , comenzilor si a altor rapoarte legate de comenzi.

# Cu ajutorul interfetei grafice , fiecare tip de utilizator ( Client , Angajat si Administrator) poate vedea functionalitatile disponibile. De ex. clientul poate cauta produse dupa anumite criterii si poate comanda . Angajatul primeste notificare cand un client pune comanda si poate vedea istoricul comenzilor , iar administratorul poate genera rapoarte dupa niste criterii specifice , poate creea meniuri noi , adauga , modifica si sterge produse.

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

# Analiza:

# Ca să simplificăm problema am împărțit aplicatia in mai multe pachete :

# **business** – Business Logic Layer

# **connection** – se face conexiunea cu baza de date

# **data** – Data Access Layer

# **model** – contine modelul unui client si a unui user

# **presentation** – contine interfata grafica

# **presentation.administrator** – contine interfata grafica pentru administrator

# **presentation.employee** – contine interfata grafica pentru angajat

# **presentation.client** – contine interfata grafica pentru client

# Modelare: - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -- - - - - - - - - - - - - - -

# In functie de tipul utilizatorului , se va deschide interfata corespunzatoare pentru Client sau Angajat sau Administrator.

# In interfata de administrator avem optiune sa adaugam , sa modificam sau sa stergem un produs , sa importam setul initial de produse , sa generam rapoarte dupa anumite criterii si creearea unui Meniu format din mai multe produse. Daca s-a importat o data lista de produse initiale , la urmatoarea logare , in tabelul existent va aparea lista cu produse.

# In interfata de employee , cand un client pune o comanda se va primi un mesaj care va fii afisat ca un pop-up si totodata se poate vedea istoricul comenzilor

# In interfata de client se afiseaza lista cu produsele si cu meniurile disponibile , iar clientul poate face o comanda sau sa caute produsele dupa un anumit criteriu.

# Scenarii: - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

# Pentru a nu exista niciun fel de eroare , utilizatorul trebuie să respecte următoarele condiții :

# La generarea rapoartelor de catre administrator , la raportul pentru comenzi intre anumite ore se va introduce ora de tipul : 10 , 12 , 23 etc.

# In formularul de inregistrare la Age(varsta) se va introduce un intreg.

# La adaugarea sau modificarea unui produs , inafara de rubrica Title unde se va introduce un string , in celelalte campuri se vor introduce valori de tipul double (ex 12.0);

# Pentru crearea unui meniu nou se vor selecta din tabel mai multe produse si se va apasa pe create menu.

# Pentru client la cautarea dupa anumite criterii , inafara de keyword unde se va introduce un string , in rest se vor introduce valori de tipul double

# Pentru stergerea unui produs se selecteaza din tabel linia dorita pentru stergere.

# Pentru modificarea unui produs administratorul trebuie sa modifice direct in tabel valorile respective si sa ramana cu linia respectiva selectata cand apasa pe butonul Update.

# **Proiectarea (decizii de proiectare , diagrama UML , structuri de date , proiectare clase , interfete , relatii , packages , algoritmi , interfață utilizator)**

# Programul a fost împărțit în mai multe pachete pentru a fi mai ușor de înțeles:

# Pachetul **business** care contine logica acestui program si asigura functionarea corecta :

# 

S-a folosit Composite Design Pattern pentru produse si este format din clasele MenuItem care este o clasa abstracta si contine metoda de computePrice ,BaseProduct (care contine modelul unui produs de baza) si extinde MenuItem si clasa CompositeProduct care la fel extinde MenuItem si contine o lista de BaseProduct , adica un produs compus din mai multe produse de baza.

Mai avem interfata IDeliveryServiceProcessing care contine metode de adaugare , sterge , modificare produs etc.

Clasa CreateAccount care insereaza userul format din user , parola si tip in baza de date si celelalte date despre client intr-un fisier serializabil cu clienti.

Clasa Order care contine structura unei comenzi.

Clasa DeliveryService care implementeaza interfata IDeliveryServiceProcessing si implementeaza metodele din interfata , aici avem si observerul care da notify cand se face o comanda si functia care genereaza chitanta plus un override la functia hashCode.

# Pachetul **connection** care contine conectarea la baza de date :



Clasa ConnectionFactory face legatura cu baza de date relationala MySQL.

* Pachetul **data** – Data Access Layer – contine o clasa unde se fac serializarile si deserializarile si o clasa care face citirea din fisierul initial de produse

# 

# Clasa ReadProducts face citirea produselor din fisierul initial products.csv si foloseste stream si serializare pentru stocarea produselor intr-un fisier serializabil.

# Clasa Serializator contine metode care fac serializarea si deserializarea Clientilor , Produselor , Comenzilor etc.

# Pachetul **model** care contine structura unui User si a unui Client :

# 

# Clasa Client contine modelul unui client care va fi stocat intr-un fisier serializabil clients

# Clasa User contine userul , parola si tipul clientului care se stocheaza intr-o baza de date MySQL.

# Pachetul **presentation** contine interfata grafica impartita in trei pachete cu cate o interfata proprie , una pentru Administrator , una pentru Client si una pentru Employee(Angajat) si contine 2 clase , cea de Login care se deschide la rularea aplicatie si formularul de inregistrare pentru un client nou.

# 

# Clasa Login este clasa care se deschide la rularea aplicatiei si permite utilizatorului sa introduca user si parola pentru conectare sau sa apeleze butonul de Register care il va duce in interfata din clasa Registration unde isi poate face un cont nou.

# Pachetul **presentation.administrator** contine interfata grafica a utilizatorului de tip administrator :

# 

# Clasa AdministratorGUI este clasa principala care se apeleaza daca in urma logarii utilizatorului el este de tipul administrator.

# Clasa CreateMenu reprezinta un JFrame care se deschide daca administratorul a creeat un Meniu (CompositeProduct din mai multe BaseProduct) si afiseaza un tabel cu meniurile existente.

# Clasa RapoarteGUI reprezinta tot un JFrame in care se pot genera rapoartele dorite de catre administrator.

# Clasa Reports contine metodele care genereaza rapoartele cerute.

# Pachetul **presentation.employee** contine interfata grafica pentru un angajat :

# Clasa EmployeeGUI reprezinta interfata grafica pentru utilizatorul de tip Employee.

# Pachetul **presentation.client** contine interfata grafica pentru un client:

# 

# Clasa ClientGUI reprezinta interfata care se deschide atunci cand in urma logarii , utilizatorul este de tip Client si contine fereastra principala pentru un client.

# Clasa Cos este un JFrame care se deschide in urma apasarii butonului Cos din ClientGUI si afiseaza produsele care se doresc a fii comandate + pretul total , iar in urma apasarii butonului confirma se va procesa comanda.

# Clasa Criteria este tot un JFrame in care exista niste campuri unde poti filtra produsele existente dupa anumite criterii .

# **Diagrama UML** : - - - - - - -

**Interfață Grafică :**

* **Butoane –** avem butoane care fac trecerea dintr-un JFrame in altul sau butoane care doar fac Search-uri sau pentru adaugarea , modificare , stergere produs , creare meniu, istoric comenzi.
* **JPanel-uri –** Avem paneluri unde afisam tabelele si care contin celelalte elemente de interfata : butoane , textfield , password field etc.
* **JTextField-uri –** Avem text-field-uri care preiau datele de la utilizator , de ex in formularul de inregistrare sau la creearea de produse noi , la generarea de rapoarte sau la cautarea dupa un anumit criteriu
* **JLabel-uri –** Avem cate un label pentru text-field-uri ca utilizatorul sa stie ce trebuie sa introduca si mai avem label-uri care contin imaginile de background ale aplicatiei.
* **Frame-uri -** fereastra care se deschide atunci când rulăm programul și care conține toate elementele legate de partea de interfață grafică a proiectului. Precum majoritatea aplicațiilor are în partea dreaptă sus 3 butoane : `--` Restore/Punerea in bară , `patratul` e minimizarea /maximizarea ferestrei si `X` reprezintă închiderea programului.

# **Implementare**

# A fost implementata o aplicatie care simuleaza o aplicatie de food delivery precum Glovo sau FoodPanda unde clientii pot alege din mai multe produse si sa comande , angajatii sa primeasca notificare cand s-a efectuat o comanda noua si sa vada istoricul comenzilor , iar administratorul sa introduca produse noi , sa le modifice , sa le stearga , sa creeze meniuri noi si sa genereze rapoarte pe anumite criterii.

# Dupa inserarea unei comenzi se va genera o chitanta in format .pdf cu produsele comandate , pretul fiecaruia si pret total si data la care s-a facut comanda.

# Programul a fost impartit in mai multe pachete (business,connection,data,model,presentation)

# Clasa **business.MenuItem** – este o clasa abstracta care contine metoda de computePrice()

# 

# Clasa **business.BaseProduct** – extinde clasa MenuItem si contine structura unui produs de baza + toString + getters si setters si un override la hashCode + implementarea metodei de computePrice()

# 

# Clasa **business.CompositeProduct** – extinde clasa MenuItem si contine o lista de BaseProduct + getters si setters + metoda de computePrice()

# 

# Clasa **business.Order** – contine structura unei comenzi :

# 

# Interfata **business.IDeliveryServiceProcessing** – este o interfata care are metodele de adaugare , stergere m modificare produs , creare comanda noua , calcularea pretului unei comenzi si generarea chitantei.

# 

# Clasa **business.DeliveryService** – implementeaza interfata IDeliveryServiceProcessing si implementeaza metodele respective si aici se face notify la observer cand se produce o comanda noua :

# 

# Clasa **business.CreateAccount** – insereaza in baza de date User-ul creat (doar datele de logare si tipul de utilizator) :

# 

# Clasa **connection.ConnectionFactory** – face legatura cu baza de date MySQL

# 

# Clasa **model.User** – contine structura unui User :

# 

# Clasa **model.Client** – contine structura unui client :

# 

# Clasa **data.ReadProducts** – face citirea din fisierul initial de produse (products.csv) si serializeaza produsele intr-un fisier serializabil :

# 

# Clasa **data.Serializator** – contine metodele care fac serializarea si deserializarea :

# 

# Si totodata mai avem clasele care reprezinta interfata grafica pentru fiecare tip de utilizator.

# **Rezultate**

# **Pagina Logare**

# **Formular inregistrare**

# **Interfata Administrator**

# **Formular adaugare produs nou**

# 

# Meniuri disponibile in urma creearii unui nou meniu

# Pagina unde se genereaza rapoartele administratorului

# Interfata Angajat

# Interfata Client

# Pagina unde clientul poate cauta dupa anumite criterii

# Cos-ul in urma adaugarii de produse pentru a fii comandate de catre client

# 

# **Concluzii**

Aplicatia are o interfata user-friendly , la pornirea ei se deschide fereastra de Login unde utilizatorul introduce datele pentru logare sau isi poate creea un cont nou. In urma logarii , i se va deschide fereastra corespunzatoare tipului de utilizator (Administrator , Client sau Employee) si are la dispozitie mai multe butoane pentru efectuarea a diferite actiuni.

# **Bibliografie**

# [**StackOverflow**](https://stackoverflow.com/)

# **[Baeldung](https://www.baeldung.com/java-observer-pattern)**

# [**TutorialsPoint**](https://www.tutorialspoint.com/java/java_serialization.htm)