**Documentatie**

Assignment 4

Nume: Neacă Radu-Sabin

Grupa: 30222

**Cuprins**

[1. Obiectivul temei 3](file:///C:\Downloads\FACULTATE\Tehnici%20de%20programare\PT2021-2022_Documentation_Template.doc#_Toc95297885)

[2. Analiza problemei, modelare, scenarii, cazuri de utilizare 3](file:///C:\Downloads\FACULTATE\Tehnici%20de%20programare\PT2021-2022_Documentation_Template.doc#_Toc95297886)

[3. Proiectare 5](file:///C:\Downloads\FACULTATE\Tehnici%20de%20programare\PT2021-2022_Documentation_Template.doc#_Toc95297887)

[4. Implementare 6](file:///C:\Downloads\FACULTATE\Tehnici%20de%20programare\PT2021-2022_Documentation_Template.doc#_Toc95297888)

[5. Rezultate 8](file:///C:\Downloads\FACULTATE\Tehnici%20de%20programare\PT2021-2022_Documentation_Template.doc#_Toc95297889)

[6. Concluzii 8](file:///C:\Downloads\FACULTATE\Tehnici%20de%20programare\PT2021-2022_Documentation_Template.doc#_Toc95297890)

[7. Bibliografie 9](file:///C:\Downloads\FACULTATE\Tehnici%20de%20programare\PT2021-2022_Documentation_Template.doc#_Toc95297891)

Obiectivul temei

Obiectiv principal:

Scopul lucrarii consta in crearea unei aplicatii, un food delivery, ce efectueaza diferite operatii pe care le face un administrator , un client, sau un angajat. Administratorul este responsabil cu inserarea produselor, stergerea si modificarea lor. Acesta poate crea meniuri, ce sunt create din mai multe produse. Clientul este responsabil cu vizualizarea lor, si selectarea produselor dar si meniurilor pe care odreste sa le cumpere. Angajatul este responsabil cu vizualizarea comenzilor, astfel incat sa observe data si ora cand clientul a cumparat produsele dorite.

Obiective secundare:

* Analiza problemei si identificarea cerintelor.
* Organizarea codului corespunzaor: Precum structura Composite Design Pattern si Observer Design Pattern. Acestea sunt folosite pentru definirea claselor MenuItems, BaseProduct, CompositeItems, dai si pentru notificarea angajatului in momentul in care un client iti plateste comanda.
* Fiecare clasa creata nu trebuie sa depaseasca o dimensiune mai mare de 300 de linii. Fiecare metoda trebuie sa cuprinda maxim 30 de linii. Clasele trebuie impartite in pachete astfel incat sa existe o vizibilitate si organizare a proiectului mult mai buna.
* Testarea aplicatiei, verificarea codului astfel incat metodele folosite pe parcursul programului sa fie corect implementate, fara a prezenta cazuri de eroare.
* Implementarea si simularea unei interfete ce sa dispuna si de o fereastra ce afiseaza rezultatul dorit.
* Folosirea diverselor tehnici de programare precum serializarea, ce dispune salvarea unui obiect intr-un fisier, acesta urmand sa fie deserializat, adica informatia din el sa fie citita oricand dorim. Astfel rezultatul serializarii este de a salva intr-un fisier informatii astfel incat in momentul reportnirii aplicatiei, produsele schimbate sa nu iti revina la valoarea initiala.
* Alta tehnica de programare importanta o reprezinta folosirea expresiilor lambda.

Analiza problemei, modelare, scenarii, cazuri de utilizare

Primul pas inainte de implementarea proiectului ar fi analiza problemei si modul cum ar trebui rezolvata aceasta. Un bun inceput in crearea aplicatiti il poate reprezenta citirea din fisierul excel. Aceasta se poate realiza cu ajutorul librariei “Serialization”. Se salveaza locatia fisierului excel, cu extensia “.csv”, urmand sa fie citit prin clasa “BufferedReader” . Trebuie tratate si cazurile de exceptie, pentru o implementare corecta a codului. Casutele din excel sunt separate princ aracterul “,” asadar, datorita lui putem sa salvam linile si din excel intr-o lista de produse.

Text

Description automatically generated

Din fereastra de logare avem mai multe posibilitati de a continua. Aici ne sunt prezente mai multe cazuri. Conectarea ca administrator, ca client, sau ca angajat. Pe langa acestea este prezenta si o actiune de inregistrare, ce permite fiecarui utilizator sa iti creeze un cont nou.

Graphical user interface, application

Description automatically generated

In fereastra de inregistrare am implementat urmatoarea actiune. De fiecare data cand cineva doreste sa isi creeze un nou cont, acesta este obligat sa insereze un nume si o parola, iar mai apo isa selecteze calea prin care doreste sa se inregistreze (administrator, client, angajat).

A screenshot of a computer

Description automatically generated with medium confidence

In cazul in care acesta s-a conectat ca si administrator, butonul de “Connect as Administrator” il va conduce catre o noua fereastra, in care acesta poate sa obsearva, sa insereze, sa editeze sau sa stearga produse. O alta abilitate a acestuia o reprezinta alcatuirea unui meniu din mai multe produse. Tot ce trebuie sa faca este sa dea o denumire meniurilor, si sa selecteze cate un meniu, iar apoi sa apese butonul de “AddProductInMenu”.

Graphical user interface, application

Description automatically generatedGraphical user interface, table

Description automatically generated

In cazul in care ne conectam ca un client, putem observa produsele si meniurile in ultima lor faza de dupa modificare. Clientul dispune de urmatoarele actiuni. Cautarea produselor dorite, folosind cateva criterii date, precum minimul si maximul raitingului, al caloriilor, al proteinelor si al pretului. De plasarea unor produse sau meniuri intr-o comanda. In momentul in care clientul doreste sa isi cumpere produsele dorite, acesta poate observa pretul total dar si fiecare produs ales. Dupa apasarea butonului “Buy Now” lista de comenzi se sterge asteptand sa primeasca altele noi.

Graphical user interface

Description automatically generated with medium confidence

In cazul in care dorim sa ne conectam ca si angajat, putem observa toate comezile cumparate de client. Acesta poate observa si data cand a fost data comanda.

Graphical user interface, text, application, chat or text message

Description automatically generated

Proiectare

Pentru proiectarea codului, am urmat structura Composite Design Pattern si Observer Design Pattern. ,astfel incat fiecare pachet adaugat sa aibe un rol specific. In pachetul “data.layer”, are loc serializarea si desearializarea fisierului cu extensia “.csv”. In pachetul “business.layer” am create clasele cu metodele ce realizeaza operatiile administratorului, clientului si a angajatului. Iar in clasa “presentation.layer” am create clasele ce se ocupa cu interfata proiecutlui.

Graphical user interface

Description automatically generated

Se poate observa faptul ca nu am realizat vreo legatura intre clase, neavand vreo instanta a unei alte clase intr-o alta clasa. Pe langa pachetele de mai sus, sunt prezente si resursele primite datorita aplicatiei SceneBuilder. Aici sunt prezente toate ferestrele aplicatiei, cum ar fi urmatoarele:

Graphical user interface

Description automatically generated with low confidence

Implementare

In implementarea aplicatiei am urmat structura Composite Design Pattern si Observer Design Pattern care sa respecte totodata si conditiile unei bune organizari a codului, astfel incat sa nu am clase ce depasesc de 300 de linii sau metode ce au mai mult de 30 de linii. Dupa cele prezentate in capitolul anterior, acum putem discuta si despre clasele si metodele apelate. Astfel, incepem cu Clasa unde se lanseaza aplicatia. Primul lucru care trebuie respectat cand o aplicatie se leaga cu o baza de date, este conectarea codului la baza de date. Astfel, prima clasa de unde plecam este clasa “Main” unde am pornit programul “launch()”, si unde ar fi trebuit sa aibe loc serializarea si deserializarea programului, inainte de inchiderea aplicatiei. Aceasta duce catre o metoda “start()” ce reprezinta prima fereastra a programului nostru, fereastra de conectare. In acest moment, fereastra se deschide datorita comenzi oferite de javaFX, “FXMLouder” care ne permite deschiderea unui fisier cu extensia “.fxml”. Dupa deschiderea acesteia, avem cateva actiuni ce sunt realizat in clasa “Controller”. O prima actiune o reprezinta crearea unui nou cont “Register now”, unde utilizatorul poate sa isi creeze un nou cont. Am salvat numele, parola si tipul contului intr-o lista de conturi. Metoda de initializare “initialize()” se realizeaza de fiecare data cand o actiune din clasa respectiva are loc, astfel, “logInAsChoiceBox” se initializeaza mereu cu valorile respective cand fereastra de inregistrare de deschide. Alte 3 metode sunt cele in stilul “conectare ca”. Aici ne sunt prezente 3 variante, conectare ca administrator, client sau angajar. In metoda “clickOnLogInAsAdministrator()” am verificat daca stringurile din textField-urile inserate reprezinta un cont de tipul “Administrator” iar in caz afirmativ, o noua fereastra se deschide. Similar cu aceasta se realizeaza si celelalte 2 metode, “clickOnLogInAsClient ()” “clickOnLogInAsEmployee ()”.

In clasa “Administrator” sunt prezente mai multe metode cu diferite actiuni. Inainte de toate, m-am folosit de o lista de produse pentru a salva fiecare produs inserat. Afisarea acestei liste se realizeaza printr-un tableView, asadar m-am folosit de o metoda de tipul ObservableList<BaseProduct> pentru a returna lista de produse. Similar cu aceasta am creat si o lista de meniuri (CompositeProduct). Aceasta este afisata tot intr-un tableView, asa ca am folosit o metora de tipul ObservableList<MenuItems> pentru a salva lista de meniuri. In metodele “clickOnRefreshMenuItems()” si “clickOnUptadeTableView” am realizat actualizarea tabele de produse dar si a tabelei de meniuri mereu cand se apasa butonul cu actiunea respectiva. Metoda “clickOnImportProducts()” am realizat afisarea in tabela a produselor din fisierul cu extensia “.csv”. Asadar, la apasarea butonului cu cu actiunea respectiva, toate modificarile aduse tabelei, o sa revina la normal. In metoda “clickOnInsertButton()” am realizat deschiderea ferestrei ce se ocupa cu inserearea de produse. In metoda “clickOnAddPRoduct()” am realizat inserarea unui nou produs in tabela. Similar uc aceste 2 metode sunt si metodele “clickOnEditWindow()” si “clickOnEditButton()”. Prima metoda deschide fereastra ce se ocupa cu editarea produsului in timp ce a 2-a fereastra se ocupa cu editarea produsului respectiv din lista de produse. Medota de “clickOnDeleteButton()” are ca actiune, stergerea produsului care a fost selectat uc mouseul. Aceasta actiune este specifica librariei “tableView”. In continuare ne sunt prezentate cateva metode ce se leaga de crearea de meniuri (Composite Product). In metoda “clickOnAddProductInMenu()” are rolul de a adauga un produs selectat din tabela in meniu. In momentul in care se adauga 3 produse, se creaza un meniu nou cu numele ce a fost inserat in casuta de text field. Metoda numita “clickOnViewMenu()” are rolul de a deschide o fereastra ce contine toate meniurile inserate. Metoda “clickOnDleteMenu()” are rolul de a sterge meniul selectat cu mouse-ul.

In clasa “Client” au loc actiuni ce tin de interfata clientului. Asadar sunt folosite diferite variabile precum cele 2 liste de produse si de meniuri, ce au fost afisate si modificate similar ca la Administrator. De aceasta data sunt prezente metode diferite precum metoda “clickOnSearchItm” ce are rol de a cauta in tabela produsele ce respecta coditiile cerute (pretul minim si maxim, raitingul minim si maxim, caloriile minime si maxim, proteinele minime si maxime). Se regasesc metode cu actiuni similare ca la administrator, precum metoda de “clickOnUpdateTableView” si “clickOnRefreshTableView()” ce au scop de a actualiza tabela de produse, dar si pe cea de menmiuri. Metoda “clickOnBuyMenu()” are rolul de a adauga in lista de comenzi, meniul selectat cu mouse-ul. Metoda”clickOnBuyPRoduct()” are rolul de a adauga in lista de comenzi produsul selectat cu mouse-ul. Medota “clickOnBuyAll()” are rolul de a realiza comanda inserata si de a trimite catre angajat faptul ca o comanda a fost platita. Metoda “clickOnLogOutButton()” are rolul de a deconecta clientul de la server.

In clasa Employee sunt realizate metode specifice unui angajat, precum cea de observare a comenzilor date de catre client. In metoda.

In clasa BaseProduct am inserat toate atributele pe care trebuie sa le detina un produs. Aceste atribute sun: titlul, raitingul, numarul de calorii, numarul de proteine, numarul de grasimi, numarul de sodiu dar si pretul acestuia. Prin constructorul acestuia salvez toate informatiile pe care le primeste. Sunt prezente si settere si gettere.

In clasa CompositeProduct este prezenta o lista de produse, lista in care se salveaza toate produsele ce sunt trimise prin constructorul acestuia. O variabila noua o constituie totalPrice, ce reprezinta pretul total pe care il au produsele din lsita.

Clasa MenuItems realizeaza operatii intre cele 2 clase de mai sus, fiin o superclasa care contine atat clasa BasePRoduct cat si clasa CompositeProduct.

In clasa Order sunt prezente atributele pe care trebuie sa le contina o comanda. Aceste atribute ar fi: id-ul comenzii, id-ul clientului, dar si data cand a fost plasata comanda.

In clasa DeliveryService sunt realizate operatii ce tin atat de produse cat si de meniuri. Astfel, sunt prezente atat o lista de produse, cat si o lista de meniuri (CompositeProduct). O prima metoda prezenta ar fi “readDeliveryService()” ce are rol de a citi elementele din fisierul cu extensia “.csv”. locatia fisierului este data de path-ul trimis ca si string. In metoda addProductDeliveryService() se adauga un produs in lista de fiecare data cand este apelata metoda. Se returneaza de fiecare data lsita actualizata. In metoda “editProductDeliveryService()” are rolul de a returna lo lista ce se actualizeaza in functie de parametrii pe care ii primeste. Primul parametru il reprezinta lista needitata. In metoda “deletePRoductListDeliveryService()” are rolul de a returna o lista in momentul in care un element a fost sters. Metoda “addPRoductInMenu” are rolul de a adauga produsul inserat in lista de produse. Metoda “deleteDeliveryService()” are rolul de a sterge o lista de produse din lsita de meniuri. Primul parametru reprezinta lista de produse in timp ce al 2 lea reprezinta produsul cautat a fi sters. Cateva metode ajutatoare sunt folosite pentru afisarea listei de produse “showProductList()” dar si “showMenuItems()” pentru a afisa lista de produse.

In clasa Serialization si clasa FileWriter ar fi trebui implementate metode de serializare si deserializare ce sunt citite de fiecare data cand programul porneste, si sunt scrise de fiecare data cand programul se opreste.

Rezultate

Ca rezultate obtinute in urma realizarii proiectului ar fi folosirea expresiilor lambda, deschiderea fisierelor de tip “.csv” si afisarea lor in tabele. Programul functioneaza criteriilor minime cerute, cerintele realizate fiind implementate complet. Ce as mai fi adaugat la program ar fi fost implemetarea serializarii ce are ca scop principal salvarea datelor dupa inchiderea programului, si citirea lor dupa ce este pornit din nou.

Concluzii

Concluzia finala este data de experienta castigata in urma acestui proiect. Metodele de programare folosite in crearea acestei aplicatii m-au dezvoltat in cariera pe care doresc sa o urmez

Bibliografie

1. <https://dsrl.eu/courses/pt/materials/PT2021-2022_Assignment_4.pdf>
2. <https://dsrl.eu/courses/pt/materials/A4_Support_Presentation.pdf>
3. <https://javahungry.blogspot.com/2013/08/hashing-how-hash-map-works-in-java-or.html>
4. <https://howtodoinjava.com/java/collections/hashmap/how-hashmap-works-in-java/>
5. <https://www.baeldung.com/java-serialization>