**Food Delivery Management System**

Logo, company name

Description automatically generated

Student: Alexa Andrei Stefan

Grupa: 30225

1. **Obiectivul temei**

Proiectarea si implementarea unui sistem de management pentru livrarea mancarii de catre o companie de livrat. Clientul comanda produsele din meniul companiei. Sistemul trebuie sa aiba 3 tipuri de utilizatori care sa se autentifice folosind un nume de utilizator si o parola: Administrator, Angajat obisnuit si Client.

Administratorul poate sa:

* Importe setul initial de produse care va popular meniul dintr-ul file .csv dat.
* Sa administreze produsele din meniu: sa adauge / stearga / modifice produsele si sa creeze noi produse compuse din mai multe produse existente deja (un exemplu de produs compus ar putea fi numit “meniul zilei 1”, compus dintr-o supa, o friptura, o garniture si un desert).
* Sa genereze rapoarte despre performanta

Clientul poate sa:

* Sa se autentifice daca are cont, iar daca acesta nu are un cont poate sa se inregistreze apasand pe butonul de register folosind un nume de utilizator si o parola adecvata, iar mai apo isa se logheze in sistem introducandu-le in cele 2 campuri si apasand pe butonul login.
* Sa vada lista de produse
* Sa caute produse bazate pe unul sau mai multe criterii cum ar fi un cuvant cheie(de ex. “soup”), recenzii, numarul de calorii / proteine / grasimi / sodiu / pret.
* Creeze o comanda care sa contina mai multe produse – pentru fiecare comanda data si timpul vor fi notate si o factura va fi generata care va contine produsele comandate si pretul total al comenzii.

Angajatul este notificat de fiecare data cand o noua comanda este plasata de catre un client pentru a putea prepara comanda si toate celelalte lucruri ce tin de livrare.

De ce este util un astfel de sistem de livrare?

Acest proiect reda o necesitate din viata reala a unei astefel de aplicatii. Detinerea unui sistem de management pentru livrarea mancarii este vital in zilele noastre pentru a avea un restaurant functional. O astfel de aplicatie simplifica cu mult procesul de comandare si de livrare a mancarii atat pentru restaurant (staff si angajati) cat si pentru cel care a comandat (client). Clientul poate, simplu, sa vada o lista de produse si sa isi creeze o comanda care sa contina exact de ce are nevoie. Staff-ul restaurantului poate sa administreze ce este in lista, sa vada atat raporturile despre restaurant , cat si toate comenzile pentru a putea fi livrate cat mai repede cu putinta.

# Analiza problemei, modelare, scenarii, cazuri de utilizare

* 1. **Analiza problemei**

Am folosit o functie de hardcodare pentru introducerea unui administrator si al angajatilor.

Crearea unui design potrivit al claselor.

Adaugarea functionalitatilor cerute. (cerinta de mai sus)

Utilizarea serializarii.

Utilizarea stream processing – ului si al expresiilor lambda.

Creearea unei facturi.

* 1. **Modelare**

Am creat inferfete diferite pentru fiecare tip de utilizator, asftfel incat la introducerea datelor (username si parola) sa apara pagina corespunzatoare fiecaruia.

Utilizatorii normali pot sa se inregistreze la restaurant. Membrii staff-ului nu se pot inregistra (de ex. Intr-un caz real administratorii ar avea deja parola si numele de utilizator).

Pentru parole am folosit un JPasswordField pentru confidentialitatea parolei.

Procesul de comandare a mancarii ar trebui sa sa fie cat de usor si intuitive posibil, pentru client. Ei trebuie sa vada o lista cu cee ace poate fi comandat, inclusive detalii si pret, inainte de a confirma comanda.

Administratorul poate sa editeze meniul si sa adauge produse compuse din alte produse deja existente.

* 1. **Scenarii si cazuri de utilizare**

Exista trei tipuri de utilizatori: Administrator, Angajat si Client.

Clientul poate sa se logheze folosind numele de utilizator si parola corecte. Daca acestia nu au un cont, pot sa se inregistreze si mai apoi sa foloseasca contul inregistrat. Pot sa vada o lista cu toate produsele pe care le pot comanda, pot sa filtreze aceste produse in functie de o informatie specifica sau pot introduce criteriile prdusului pe care vor sa il comande. Dupa ce comanda a fost palasata, un fisier text va fi generat care va contine factura cu delatiile comenzii.

Angajatul poate sa se logheze si sa vada toate comenzile, acesta fiind notoicat de fiecare data cand o caomnda este plasata, el poate sa isi stearga lista de comenzi plasate prin apasarea butonului clear.

Administratorul poate sa editeze lista de produse: sa adauge / stearga / editeze / sa gaseasca un produs specific sau sa creeze un nou meniu din produsele deja existente.

Butoanele sunt denumite in asa fel incat utilizatorul sa poata sa is idea seama de ceea ce fac si de cee ace ar trebui sa faca pentru a folosi aplicatia in mod corespunzator, dar de asemenea am inclus si cateva erori care apar in cazul in care au fost gresite informatiile introduse.

1. **Proiectare**
   1. **Decizii de proiectare**

Baza acestui proiect este arhitectura prezentata in prezentarea suport a temei.

Interfata trebuie sa fie cat de usor si de intuitiva posibil de folosit si sa aiba toate functionalitatile cerute.

In cazul introducerii unor date gresite, am introdus afisarea unui mesaj de eroare pentru ca utilizator sa stie ce trebuie sa reintroduce correct.

* 1. **Clase si pachete**

Am folosit Model, View, Controller pentru o mai buna organizare si claritate a acestui proiect.

In pachetul View am introdus clasele ce tin de interfata: Admin, Client, Employee, CreateMenu, CreateProduct, EditMenu, EditProduct, Login, Register.

In pachetul Model am create clasele: BaseProduct, CompositeProduct, DeliveryService, FileWriter, MenuItem, Order, Serializator, User.

In pachetul Controller am create clasa Controller, care se ocupa cu interactiunile dintre utilizator si aplicatie.

Clasa Serializator se ocupa cu serializarea si deserializarea datelor din program.

Clasa Order se ocupa cu generarea datelor unei comenzi.

Clasa Rank este un ENUM care contine cele 3 tipuri de utilizatori posibili.

Clasa User o folosesc pentru crearea unui nou client.

* 1. **Proiectarea claselor, Interfetelor si Pachetelor**

Toate clasele sunt proiectate respectand modelul sugerat in prezentare, dar am mai adaugat si cateva clase care ajuta la o mai buna lizibilitate. Clasa User este specifica tuturor tipurilor de utilizator , iar cu ajutorul enum-ului alegem la hardcodare ce fel de utilizator dorim sa cream.

Clasa de serializare se ocupa cu serializarea si deserializarea datelor.

Am folosit hashmap pentru maparea datelor din DeliveryService in tabel.

Ideea principala a aplicatiei este de a avea butoane in interfata,, care apasate sa declanseze un set de evenimente. (ex : client plaseaza o comanda -> noi pregatim comanda, cream facture si adaugam comanda la datele programului, serializam informatia).

* 1. **Interfata**

Interfata utilizator rebuie sa fie cat mai usor posibil de folosit si cat mai intuitive. Aceasta aplicatie are cateva ferestre specific pentru fiecare tip de utilizator si contine butoane pentru actiunile specific acelui utilizator.

Admin-ul are cea mai complexa fereastra. Contine campuri pentru fiecare atribut al unui BaseProduct pe care administratorul poate sa le completeze si a apo isa foloseasca alte comenzi asupra datelor inserate. Admin-ul poate sa adauge un produs, sa stearga un produs, sa editeze un anumit produs, sa caute un anumit produs sau sa creeze un meniu care e compus din produsele de baza selectate. Admin-ul poate de asemenea sa vada o lista cu toate comenzile si sa le filtreze bazat pe cateva criterii precum, ora, minut, zi sau comanda.

Clientul are si el o pagina destul de complexa. Si el are de asemenea campuri pentru toate produsele caracteristice pentru filtrarea lor bazata pe cee ace doreste. El poate sa adauge un produs in cosul de cumparaturi, sau sa stearga un produs din cos, la plasarea comenzii, clientul apasat pe butonul Order, iar comanda va fi procesata, genrand o facture care contine datele, detaliile si pretul produselor comandate.

Angajatul poate sa vada o lista cu toate comenzile plasate si poate sa isi stearga lista de comenzi plasate prin apasarea butonului clear.

Exista de asemenea si o pagina pentru logare sau inregistrare. La apasarea butonului de inregistrare, utilizatorul va fi trimis pe pagina specifica, unde i se va cere un nume de utilizator si o parola care trebuie confirmata. In cazul in care parola difera, un mesaj de eroare se va afisa pe ecran, astefel incat utilizatorul sa stie ce trebuie sa aiba in vedere pentru a-si corecta greselile.

Graphical user interface

Description automatically generated

Graphical user interface, application, table

Description automatically generated

Graphical user interface

Description automatically generated with medium confidence

Graphical user interface

Description automatically generated

1. **Implementare**

In aceasta sectiune voi prezenta si explica cateva din partile de cod implementate, printre care metode si clase.

Text

Description automatically generated

Clasa Controller contine functia tabelProduse care creeaza un tabel dintr-un set de produse si functia start care porneste aplicatia, deserializeaza obiectul, deliveryService si in cazul terminarii aplicatiei se serializeaza obiectul.

Text

Description automatically generated

Cateva din functiile mai importante din CompositeProduct. addProduct adauga un produs in meniu, removeProdus sterge un produs din meniu, findProduct gaseste un produs din meniu.

Text

Description automatically generated

Clasa DeliveryService este clasa de baza, aici este clasa principala unde sunt stocate datele.

Am hardcodat cateva valori pentru toate cele 3 tipuri de utilizatori posibili in proiectul nostru de food delivery management system pentru a putea accesa fiecare dintre ferestrele speficice acelui utilizator.

Text

Description automatically generated

Aceasta fucntie ne ajuta la transformarea obiectelor din file-ul products.csv intr-un set de MenuItem.

Text

Description automatically generated

Parseaza o linie din file si o transforma intr-un Menuitem.

Text

Description automatically generated

Aceasta functie de serializare codeaza intr-un fisier bitstream, care se poate deschide pentru a se face load.

Text

Description automatically generated

Aceasta functie decodifica dintr-un fisier bitstream.

Text

Description automatically generated

In aceasta functie cream cate un utilizator in functie de rank-ul acestuia, el poate fi conform si enum-ului de tipul administrator, client sau angajat, iar in fucntie de rank-ul sau va putea indeplini diferite servicii.

Administratorul are cea mai complexa activitate, el fiind responsabil cu toata organizarea listei de produse care pot fi comandate de catre clienti si care se afla pe stoc in momentul comenzii.

Filtrarea produselor este implementata de o functie care presupune folosirea unor expresii lambda.

1. **Rezultate**

La sfasitul acestui proiect am reusit sa realizez intreaga functionalitate a programului droit.

Admin-ul poate sa adauge noi produse in meniu, poate cauta / sterge / edita produse deja existente si poate de asemenea sa creeze un meniu alcatuit din produse deja existente in lista de produse. Adminul are de asemenea access la rapoarte, unde poate sa vada toate comenzile plasate de catre clienti. Angajatul poate sa vada toate comenzile si sa le filtreze intr-o oarecare ordine aleasa de el si poate sa isi stearga lista de comenzi plasate prin apasarea butonului clear. Clientul poate sa se logheze in contul deja existent, sau poate sa se inregistreze, el poate filtra produsele in functie de dorinta lui si sa plaseze o comanda cu produsele dispoibile in lista. Un raport si o factura pentru comanda va fi generata dup ace comanda este plasata. Inainte de a plasa comanda, clientul poate sa isi vada produsele in cosul de cumparaturi aflat in partea stanga.

Am testat acest program in cateva cazuri si am incercat sa le fac fata tuturor execeptiilor ai problemelor care au aparut. Precum introducerea de catre client a unor cuvinte unde el ar trebui sa introduca numere etc.

1. **Concluzii**

Acest proiect a fost o minunata experienta de invatat deoarece am folosit o multimer din tot ce am invatat pana acum pe parcursul cursurilor si al laboratoarelor asupra acestui subiect. Pe deasupra a ceea ce am mai folosit pana acum in proiectele anterioare, aici am utilizat serializarea, deserializarea, procesarea stream si expresiile lambda. Serializarea a fost folosita pentru a stoca intr-un fisier text datele programului pentru a putea fi deserializata si utilizata la inceputul programului sau unde este nevoie de aceasta. Stream processing ul este o cale interesanta de a procesa array-uri pentru a obtine datele dorite (de ex. Folosirea filter() pentru a obtine doar o parte specifica din array list sau products() pentru a obtine un set dintr-un array). Am folosit de asemenea un enum pentru accesarea rapida tipurilor de utilizator posibile.

Acest proiect ar putea implementa ceva mai multe lucruri incat sa fie folosita ca o adevarata aplicatie de comandare a mancarii.

1. **Bibliografie**

* Documentatia pusa la dispozitie pe site si cursurile
* JAVADOC
* [Introduction to JavaDoc | Baeldung](https://www.baeldung.com/javadoc)
* Lambda expressions and stream processing
* <https://docs.oracle.com/javase/tutorial/java/javaOO/lambdaexpressions.html>
* <https://docs.oracle.com/javase/tutorial/java/javaOO/methodreferences.html>
* [https://www.oracle.com/technical-resources/articles/java/ma14-java-se-8- streams.html](https://www.oracle.com/technical-resources/articles/java/ma14-java-se-8-%20streams.html)
* <https://winterbe.com/posts/2014/07/31/java8-stream-tutorial-examples/>
* <https://howtodoinjava.com/java8/java-stream-distinct-examples/>
* Java serialization
* [Java - Serialization - Tutorialspoint](https://www.tutorialspoint.com/java/java_serialization.htm)
* [Introduction to Java Serialization | Baeldung](https://www.baeldung.com/java-serialization)
* [Serialization and Deserialization in Java with Example - GeeksforGeeks](https://www.geeksforgeeks.org/serialization-in-java/)
* [Serializable (Java Platform SE 8 ) (oracle.com)](https://docs.oracle.com/javase/8/docs/api/java/io/Serializable.html)