DOCUMENTATIE

TEMA 4

NUME STUDENT: MARGINEAN TEODOR IOAN

GRUPA: 6

# CUPRINS

[1. Obiectivul temei 3](#_Toc95297885)

[2. Analiza problemei, modelare, scenarii, cazuri de utilizare 3](#_Toc95297886)

[3. Proiectare 4](#_Toc95297887)

[4. Implementare 5](#_Toc95297888)

[5. Rezultate 7](#_Toc95297889)

[6. Concluzii 8](#_Toc95297890)

[7. Bibliografie 8](#_Toc95297891)

# Obiectivul temei

Obiectivul principal al acestei teme este realizarea unui program care sa aiba o interfata pentru client pentru angajat si pentru admin. Fiecare interfata va avea functii complexe care vor trebuii implementate si folosite. Obiectivele secundare sunt:

* + Realizare de interfete pentru login, admin si client (optional si pentru angajat);
  + Realizarea diagramei de clase prezente in cerinta;
  + Realizarea composit design pentru cele 3 clase Menu Item, Base List, Composit List;
  + Stabilarea atributelor acestora, constructorilor, getteri si setteri;
  + Realizarea clasei de Delivery Service cu interfata Delivery Service Interface;
  + Realizarea de metode pentru Delivery Service, creare de clase Observable si Order;
  + Realizarea incarcarii din fisier de tip CSV;
  + Realizarea de functii pentru client: afisarea listei de produse, filtre de produse pentru calorii, proteine, sare, grasimi, pret si titlu si generare de comenzi (orders);
  + Crearea chintelor dupa realizarea unui order;
  + Realizarea de serealizare;
  + Realizare pentru admin: incarcare din lista CSV, modificare produs, adaugare produs, stergere produs, creare Composit List, generare de rapoarte (nu functioneaza foarte bine, doar uneori);
  + Generare de Java Doc;
  + Realizarea de Register;

Clasele de timp GUI ale proiectului sunt in numar de 3: GUI Login, GUI Admin, GUI Client. Fiecare clasa indeplineste anumite cerinte in functie de cine acceseaza prin Login aplicatia.

Clasele Controller ale proiectului sunt create individual pentru fiecare buton, aceastea asigura functionalitatea proiectului. Sunt create individual pentru a fi mai usor de urmarit. In total sunt paisprezece clase fiecare controland un buton din diferite view-uri.

In Bussines Layers se gasesc sase clase si o interfata. Trei dintre aceste clase merg pe modelul de Composit Design si anume: clasele Base List si Composit List mostenesc clasa Menu Item, preluand atributele si metodele acesteia. Tot in Bussines Layers se gaseste si clasa de Orders si clasa Delivery Services care implementeaza interfata Delivery Service Interface. Tot aici se mai gaseste si clasa Observable care momentan nu este folosita.

In clasele File Writers si Serialization se realizareaza operatiile care presupun creare de noi fisiere. In File Writers se creeaza chitanta (Bill) pentru comanda (Order) si Serialization se realizeaza serealizarea Derlivery Services, dar si deserializarea acestuia.

# Analiza problemei, modelare, scenarii, cazuri de utilizare

Sunt treisprezece use-case-uri prezente in aceasta tema: Log In, Register, Afisare, Sortare, Adaugare Cos, Plasare comanda, Modificare Pordus, Stergere Produs, Adaugare Produs, Adaugare Produs din Composit, Incarcare Lista Produse, Adaugare in lista de produse a produsului composit, Gererare Rapoarte.

Primul use-case este Log In in urma carauia se conecteaza un utilizator in aplicatie si este redirectionat spre o fereastre specifica rolului acestuia.

Al doilea use-case Register care preia datele din text field si creeaza un nou cont pentru client.

Al treilea use-case este Afisarea din interfata clientului care afiseaza in consola toate produsele din lista.

Al patrulea use-case este Sortare din interfata clientului care preia infoematiile din text field si sorteaza produsele din lista de baza in functie de cee ace a introdus utilizatorul.

Al cincelea use-case este Adaugare Cos din interfata clientului in urma carauia se adauga un produs din lista de baza in cos cu ajutorul denumirii pe care clientul l-a intordus in text flield-ul corespunzator (doar daca acesta exista).

Al saselea use-case este Plasare Coamnda din interfata clientului care preia toate produsele din cosul clientului si comanda este plasata.

Al saptelea use-case este Modificare Produs din interfata adminului care poate modifica un produs cu numele introdus in text field,in rest continand valori dummy, aceasta optiune nefiind accesibila si din partea clientilor.

Al optulea use-case este Stergere Produs din interfata adminului care sterge un produs din lista de baza si nu mai poate fi regasita din interfata de client.

Al noualea use-case este Adaugare Produs din interfata adminului care adauga un produs cu numele introdus in text field, in rest continand valori dummy, aceasta optiune nefiind accesibila si din partea clientilor.

Al zecelea use-case este Adaugare Produs din Composit din interfata adminului care poate combina anumite produse regasite prin denumirea introdusa din text field din lista de baza (este acelasi principiu ca la Adaugare in Cos).

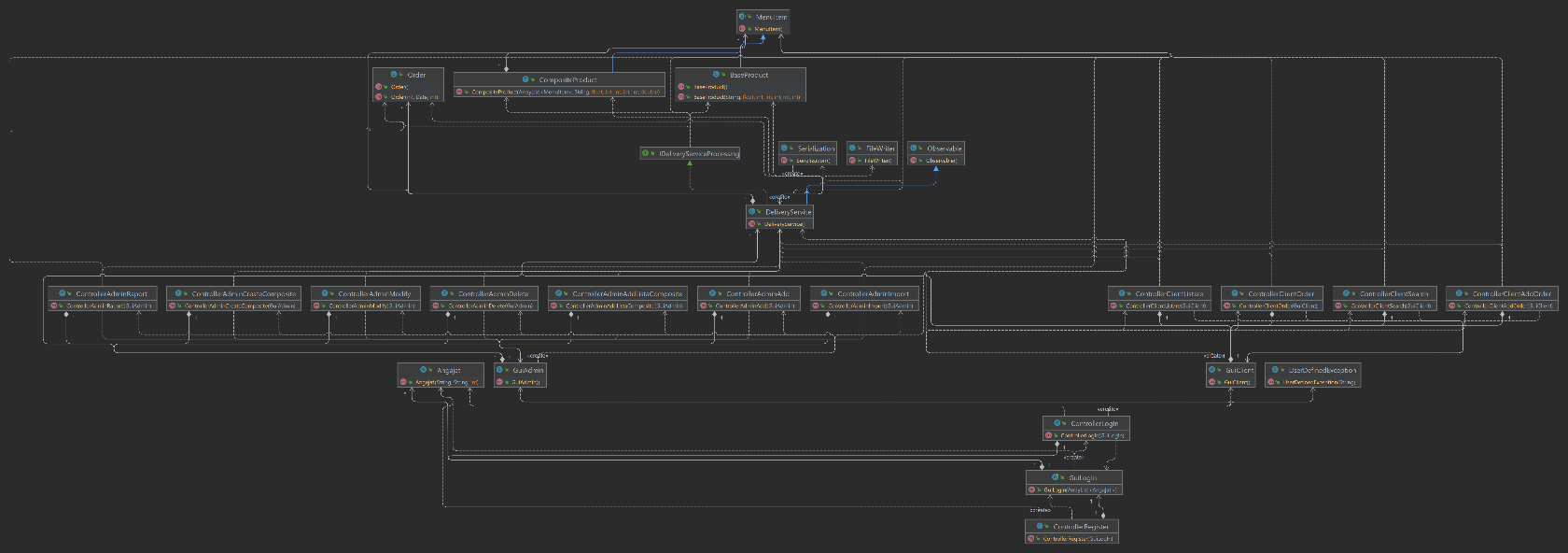
Al unsprezecelea use-case este Incarcare Lista Produse din interfata de admin care preia toate produsele adaugate pentru crearea unui produs Composit si creeaza un meniu.

Al doisprezecelea use-case este Adaugare in lista de produse a produsului composit care preia noul produs Composit (meniul alcatuit din mai multe produse din lista de baza) si adauga acest meniu pentru a fi accesibil si pentru a putea fi comandat de catre client.

Al treisprezecelea use-case este Generare de rapoarte din interfata de admin care realizeaza rapoarte in functie de activitatea desfasurata pe interafata de client (ex: cate produse s-au comandat etc)

# Proiectare

In cadrul acestei teme au fost folosite patru pachete denumite: Businesslayer, controller, DataLayer, gui. Pachetul Businesslayer are sase clase denumite Base List, Composit List, Menu Item, Order, Delivery Services, Observable si o interfata Delivery Service Processing. Pachetul controller are 14 clase: Controller Admin Add, Controller Admin Add Lista Composite, Controller Admin Create Composite, Controller Admin Delete, Controller Admin Import, Controller Admin Modify, Controller Admin Raport, Controller Client Add Order, Controller Client Listare, Controller Client Order, Controller Client Search, Controller Log In, Controller Register, User Defined Exception. Pachetul DataLayer are doua clase: File Writer si Serealization. Pachetul gui are patru clase: Angajat, GUI Admin, GUI Client, GUI Log In.



# Implementare

Proiectul este un proiect complex care are mai multe tehnici de programare prezente in acesta.

Tot proiectul pleca de la metoda main in clasa GUI Log In din pachetul gui. Aceasta instantiaza un gui care prezinta trei butoane. Fiecare buton are un controller instantiate in constructorul view-ului. Prin apasarea butonului Log In se instantiaza un nou gui care prezinta mai multe textfield-uri care trebuie comeltate pentru a putea continu executia programului. Exista alte patru butoane care realizeaza operatii aspura tabelului GuiClient din baza de date.

Prin apasarea butonuluilist, se intra in metoda insert din clasa Delivery service, in care se iau textul se inlocuiesc elementele notate din text field-uri cu elementele transmise ca parametrii, elemente preluate in controllerClientAdaugare, functia continua executia si face legatura la lista de date prin transform din Delivery Service si scrie in consola o noua instanta de lista.

Prin apasarea butonului Delete, se intra in metoda delete din clasa Delivery Sercice, in care se iau textul din text frame se inlocuiesc elementele cu elementele transmise ca parametrii, elemente preluate in controllerAdminAdaugare, functia continua executia si face legatura la baza de date prin intermediul clasei ConnectionFactory si sterge din tabelul Admin instanta de Delivery Service cu id-ul transmis ca parametru.

Prin apasarea butonului Modify, se intr in metoda update din clasa Delivery Service , in care se iau textul se inlocuiesc elementele? cu elementele transmise ca parametrii, elemente preluate in controllerAdmin Adaugare, functia continua executia si face legatura la baza de date prin intermediul clasei ConnectionFactory si modifica din tabelul Client instanta de Client cu id-ul egal cu id-ul trimis ca parametru si valoare de pe coloanal age este inlocuita cu valoara transmisa ca parametru.

Aceasta este functiomalitate asociata tabelului client. Astfel se satisfac toate cerintele care apar in preznetare pentru tabele List.

Prin apasarea butonului List se instantiaza un nou gui care prezinta mai multe textfield-uri care trebuie comeltate pentru a putea continu executia programului. Exista alte patru butoane care realizeaza operatii aspura tabelului List din baza de date.

Prin apasarea butonului Select, se intr in metoda insert din clasa Delivery Service, in care se iau textul se inlocuiesc elementele cu cu elementele transmise ca parametrii, elemente preluate in controllerClientAdaugare, functia continua executia si face legatura la baza de date prin intermediul clasei ConnectionFactory si returneaza toate datele referitaore la Client-ul cu id-ul transmis ca parametru. Aceasta metoda din Delivery Service se apeleaza de mai mutle ori pentru a fi preluate toate datele din tabel. Dupa preluarea datelor cu ajutorul metodei refelction se determina antetul tabelului si este instantiate o noau clasa de tip viewTabel in care se introduc intr-un JTable taote datele si afisat pe ecran.

Prin apasarea butonului Add, se intra in metoda insert din clasa Delivery Service, in care se iau textul se inlocuiesc elementele cu cu elementele transmise ca parametrii, elemente preluate in controllerListAdaugare, functia continua executia si face legatura la baza de date prin intermediul clasei ConnectionFactory si adauga in tabelul List o noua instanta de List.

Prin apasarea butonului Delete, se intra in metoda delete din clasa Delivery Service, in care se iau textul se inlocuiesc elementele cu cu elementele transmise ca parametrii, elemente preluate in controllerAdminAdaugare, functia continua executia si face legatura la baza de date prin intermediul clasei ConnectionFactory si sterge din tabelul List instanta de List cu id-ul transmis ca parametru.

Prin apasarea butonului Edit, se intr in metoda update din clasa Delivery Service, in care se iau textul se inlocuiesc elementele cu cu elementele transmise ca parametrii, elemente preluate in controllerListAdaugare, functia continua executia si face legatura la lista prin intermediul metodei transform si modifica din mao-ul de Order instanta de Order cu id-ul egal cu id-ul trimis ca parametru si valoare de pe coloanal age este inlocuita cu valoara transmisa ca parametru.

Prin apasarea butonului Select, se intr in metoda insert din clasa Delivery Service, in care se iau textul se inlocuiesc elementele cu cu elementele transmise ca parametrii, elemente preluate in controllerListAdaugare, functia continua executia si face legatura la baza de date prin intermediul clasei ConnectionFactory si returneaza toate datele referitaore la List-ul cu id-ul transmis ca parametru. Aceasta metoda din Delivery Service se apeleaza de mai mutle ori pentru a fi preluate toate datele din tabel. Dupa preluarea datelor cu ajutorul metodei refelction se determina antetul tabelului si este instantiate o noau clasa de tip viewTabel in care se introduc intr-un JTable taote datele si afisat pe ecran.

Aceasta este functionalitatea programului asociata tabelului List. Astfel se satisafac taote criterile din cerinta legate de tabelul Client.

Prin apasarea butonului Client se instantiaza un nou gui care prezinta mai multe textfield-uri care trebuie completate pentru a putea continu executia programului. Exista un butoan care realizeaza crearea unei noi comenzi.

Prin apasarea butonului Generare, se inta in metoda insert din clasa Delivery Service, in care se iau textul se inlocuiesc elementele cu cu elementele transmise ca parametrii, elemente preluate in controllerListAdaugare, functia continua executia si face legatura la baza de date prin intermediul clasei ConnectionFactory si adauga in tabelul ordered doar daca ceea ce introduce ca parametru este mai mica decat stocul din produs, pentru a putea compara cele doua se foloseste o apelare a metodei select din Delivery Service, care retureneaza sub forma de string toate datele necesare pentru produsul cu id-ul timis ca parametru. In urma verificarii daca rezultatul este pozitiv se peleaza o metoda de afisare care va genera cu ajutorul textwriter-ului un text care va contine clientul, prdusul si cantitatea care va fi solicitata.

Astfel se satisfac toate cerintele necesare pentru tabelul Order.

# Rezultate

In urma realizarii programului, am reusit sa indeplinesc aproape toate cerintele. Mai exista mici probleme la generarea de rapoarte care ulterior sper sa fie rezolvate. Am reusit:

* Sa realizez adaugarea, stergerea, modificarea de catre admin;
* Sa realizez adaugarea in cos a produselor de catre client;
* Sa realizez crearea unor produse Composit;
* Sa realizez importarea din fisiere de tip CSV;
* Sa realizez sortarea si filtrearea dupa anumite criterii in interfata de client in functie de infomartiile introduse de catre utilizator;
* Sa creez interfata de Log In si Register si ferestrele aferente fiecaruia;

# Concluzii

In concluzie in urma realizarii temei, am invatat sa folosim conversie de string prin intermediul parsarii, am revenit asupra conceptelor invatate si le-am reimprospatat, am refacut un model architectural de tip model controller view. Am invatat sa:

* avem un proiect structurat;
* sa facem ferestre embricate in view;
* sa creez interfete de Log In si de Register;
* sa realizez serealizarea, dar si deserealizarea intr-un proiect cu diferite fisiere;

# 7. Bibliografie

<https://dsrl.eu/courses/pt/>

<https://cadredidactice.ub.ro//sorinpopa/files/2018/10/L1_diagrame_use_case.pdf>

[o http://javarevisited.blogspot.ro/2011/02/how-hashmap-works-in-java.html](o%20http:/javarevisited.blogspot.ro/2011/02/how-hashmap-works-in-java.html)

<https://howtodoinjava.com/java8/java-stream-distinct-examples/>

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