DOCUMENTATIE

*TEMA 4*

# Obiectivul temei

Obiectivul temei este de a proiecta un sistem de livrarea a mancarii . Obiectivele principale sunt proiectarea functionalitatii a celor trei tipuri de utilizatorii pentru aplicatie : manager , client si angajat . Obiective secundare includ folosirea diferitor design parten precum composite design pattern , observer design pattern , desing by contract .

# Analiza problemei, modelare, scenarii, cazuri de utilizare

Programul este organizat in 4 ferestre principale : login , admin , client si angajat . Dupa rularea programului se va deschide o fereastra de login unde utilizatorul va putea sa se inregistreze cu un nou cont de client sau sa introduca datele si sa se audentifice . In cazul in care datele nu au fost introduse corect se va afisa o fereastra de eroare . Dupa audentificare in cazul admin acesta va putea importa produsele din fisierul products.csv , sa adauge un nou produs baseproduct sau sa modifice acest produs , sa stearga produse din tabela , sa creeze produse de tip menu ( composite product ) si la final sa genereza rapoarte pe baze unor conditii . Pentru angajat se va deschide o fereastra care va contine doar notificarile generate in momentul in care se efectueaza o comanda si un buton pentru a strege aceste notificari . Pentru clienti fereastra va contine doua tabele unul pentru carucior si celalalt pentru a vizualiza produsele din cadrul aplicatiei . Clientul poate sa adauge si sa stearga produse pentru carucior . In cadrul butonului order se va efectua o comanda trimite notificare catre angazati si generarea unui bon cu produsele din carucior si total . Clientul poate alege sa filtreze produse din tabela prin diferitele optiuni puse la dispozitie . Daca pentru o optiune nu se doreste un filtru din cele afisate se poate alege No choice ,sau se poate alege un interval intre care numerele din categoria respectiva se vor afla . Daca se apasa pe numele coloanei coloana repectiva va fi sortata crescator sau descrescator in functie de decizia clientului .

# Proiectare

Proiectarea se bazeaza pe design pattern Model View Controller avand 20 clase impartite in 3 pachete Controller care are doar clasa sa respctiva , View care contine diferitele fereste GUI necesare aplicatiei precum login\_window , admin\_widow , client\_window etc. si pachetul de Model care contine majoritatea claselor ce implementeaza functionalitate aplicatiei precum delivery\_service , menuitem , person etc .

Clase de produse folosesc composite design pattern pentru a crea o structura de arbore ,unde modificarea unui base\_product nu afecteaza o ramura din arbore in cazul acesta composite\_product , pentru modificarile separate si posibilitatea de a crea o structura recursiva ambele clase mostenesc o clasa principala Menu\_item.

Se implementeaza Observer design pattern pentru a trimite notificari din cadrul clase delivery\_service catre o lista de angajati in acest caz employe\_window care extind clasa obsever . Clasa delivery\_service implementeaza observeable pentru a putea trimite update catre observers angajati ce primesc notificarile prin intermediul unui string.

Clasele FileWriter si Serializator sunt folosite pentru operatii cu fisiere.Serializator implementeaza metode de a serializa si deserializa un obiect in timp ce FileWriter primeste argumente pentru a fi scrise intr-un fisier specificat precum bon sau reports sau citeste dintr-un fisier lista de produse.

# Implementare

Clasa de persoana este o clasa generica utilizata pentru a stoca si identifica diferile persoane ce ar putea accesa aplicatia . Clasa contine un atribut de tip enum si anume clasa JOB care identifica rolul persoanei in cadrul aplicatiei : ADMIN , CLIENT , EMPLOYEE . Un detaliu important este ca de asemenea in cadrul crearii unui nou obiect de tip persoana acesta va avea asignat un int crescator ca id printr-o variabila statica din cadrul clasei . Clasa implementeaza doare Setters si Getters , precum si un constructor .

Clasa abstracta menu\_item este utilizata pentru a defini obiecte intr - un mod general acesta are ca atribute datele unui produs title , price , proteins ect . Clasa impelemteaza setters si getters si un constructor . Pe langa aceste clasaa suprascrie metoda equals astfel incat fiecare obiect de tip menu item sa poate fi identficat unic doar prin numele sau . Deasemenea clasa contine o metoda abstract computeprice() ce va fi impelementata de catre clasele care o mostenesc .

Clasa BaseProduct este utilizata pentru a defini obiecte simple prin atributele ce se definesc de la clasa Menu\_item . Metoda computeprice() este similara unui getter .

Clasa Composite product este utilizata pentru a defini produse complexe printr-un atribute set ce poate contine obiecte menu\_item creand astfel o structura de arbore ce la randul sau pot fi obiecte de tip menuitem . Metodele implementate de clasa sunt specifice atributului set prin adaugare stergere si gasite a produselor . Metoda computeprice() calculeaza pretul total al meniului prin adunarea pretului obiectelor din setul de produse . Clasa impelenteaza setters si gettes .

Clasa Order este utilizata pentru a crea comenzi de catre clienti prin salvarea id-ului clientului si data la care sa efectuat comanda . Clasa implementeaza setters si getters precum si un contructor .

Clasa serializator implementeaza 2 metode . Metoda de serializate primeste obiectul de serializat si un string care va fi numele fisierului in care va fi serializat obiectul . Metoda de deserializare primeste ca argument un string care este numele fisierului in care se afla obiectul serializat anterior .

Clasa fileManager implementeaza cateva metode specifice fisierelor . Metoda bon primeste orderul si lista sa de pruduse si scrie in file bon.txt valorile to string ale acestora . Metoda mapping este folosite pentru a transforma un String intr-un obiecte de tip base\_product . Metoda products utilizeaza streams pentru a citi liniile din fisierul products.csv si le mapeaza intr-un set de produse si colecteaza rezultatul operatiilor efectutate . Deoarece se foloseste un set iar menu item suprascrie metoda equal ne asiguram ca obiecte vor fi unice in functie de titlu .

Clasa delivery\_service reprezinta clasa principala din punct de vedere al manipulari datelor . Atributele sale contin un hashmap pentru persoane prin string username si persoana ,un hashmap <order si lista de produse a comenzii respective> , un set ce contine toate produsele din cadrul aplicatiei , o lista de observers pentru a trimite notificari angajatiilor si o persoana curent persona care este persoana ce acceseaza ferestrele din cadrul aplicatiei dupa operatia de login . Metodele de functionalitate pentru Observer sunt addObserver , removeObserver si notifyUpdate . Metodele specifice liste de persoane sunt addPerson , removePerson si findPerson ce returneaza un obiect de tip persoana doar daca se gaseste username-ul in set si parola este corecte pentru username-ul dat . Metodel specifice setului de produse sunt addProduct , removeProduct si findProduct . Metoda filterout primeste doua obiecte de tip produs ce contin date pentru care atributele de gasit se afla intr-un interval , se folosete stream in setul de produse si se filtreaza datele produselor astfel incat numele sa contine un substring prin numele unui paramentru , iar celelatle atribure ale produsului sa fie mai mari decat atributele lui minimum si mai mici decat ale lui maximul , la final metoda retuneza un set cu obiectele gasite in functie de parametri . Clasa implementeaza Setters si getters si un constructor . Metodele de reports genereaza o lista de paramentri in functie de cei cautati . Aceste metode reprezinta reports din cadrul assigmentului in ordinea in care sau cerut . Aceste metode folosesc stream pentru a filtra o anumita lista de obiecte precum persoana , produse si returneaza un string ce este operatia toString a listei gasite.

Clasa controller este clasa principala a aplicatiei acesta face legatura intre interfata si partea de manipulare a datelor . Acesta porneste aplicatia prin metoda start care initializarea atributele din clasa sa ce sunt interfete si clasa delivery\_service . Metoda filter out primeste alegerile efectuate de catre client in cadrul ferestrei GUI .Pentru fiecare atribut al lui menu\_item se creaza doua variabile una maxima si una minima . Cu aceste variabile se creaza doua obiecte menuitem maxim si minim care vor intra in metoda filterout din delivery\_service dupa returnareaza liste de obiecte filtrate se updateaza tabela de produse din interfata . Metoda choice process primeste un string ce reprezinta alegereaza din cadrul unui ComboBox din cadrul intefetei GUI. Daca alegerea contine ‘-‘ atunci acesta este un interval iar valorile retunate vor fi numerele dinaintea si dupa acest caracter . Daca alegerea contine ‘NO’ anume No choice atunci aceste valori vor fi setate la plus si minus infinit astfel incat toate produsele sa se regaseasca in acesta interval ,iar daca contine “>” atunci minimul va fie seta la valoarea dinaintea caracterului respectiv si maximul va fi plus infinit . Metoda Computetotal se uita peste toate produsele din carucior in cadrul interfetei si creaza un nou rand de inserat ce va contine elemente cu totalul coloanelor elementelor dinainte. Metodele de start\_window initializeaza fereasta pentru care este apelata si va adauga functionalitatea specifica butoanelor din cadrul interfetei . Metoda importProduct porneste inserarea produselor in setul delivery\_service si il afiseaza .Metoda de edit product cauta produsul care sa selectat in setul de produse al lui delivery\_service dupa care va verifica ce fel de instance este obiectul gasit . Daca obiectul gasit este de tipul base\_product se va deschide fereastra utilizata pentru a crea un produs\_simplu dar de acesta data cu campurile completate astfel incat utilizatorulul sa le poate edita. Daca obiectul gasit este de tipul composite\_product atunci se va deschide fereastra de creare a obictelor meniu dar cu lista initializata din cadrul listei obiectului gasit si datelele campurilor completate cu datele obicetului de modificat ,aici se vor putea adauga produse noi sau sterge cele vechi . Metoda register primeste username si parola de doua ori daca parolele sunt diferite atunci va aparea o fereasta de eroare , daca username-ul exista atunci va fi o eroare iar in caz contrar in hashmapul din delivery\_service se va adauga un nou client. Metoda login primeste datele din cadrul intefetei in functie de acestea se cauta persoana in delivery\_service in cazul in care datele sunt incorecte se va afisa un mesaj de eroare , in cazul datelor corecte se va initializa si deschide fereastrea resprectiva pentru rolul persoanei . Metoda setup\_product\_table creaza tabela de afisat din cadrul interfei acolo unde este utilizata pentru fiecare rand din tabela se insereaza un item din lista de produse din delivery\_service .

# Rezultate

Se pot obseva rezultatele efectuate din cadrul interfetei vor persista dupa inchiderea aplicatiei se va serializa obiectul delivery\_service ce contine datele aplicatiei . La pornirea aplicatiei delivery service se va deserializa astfel va primi datele serializate anterior . O buna metoda de verificare a operatiilor este cea de generarea a bonului care va afisa itemele comandate precum si datele specifice comenzii si deasmenea generarea de rapoarte ce vor fi scrise intr-un fisier de tip txt care va contine rezultatele filtrate in functie de cele introduse de catre admin in cadrul interfetei .

# Concluzii

In cadrul acestei teme se implude implementarea a unor diferite Design Pattern si concepte java . Intefetele de stream ofera o simpla metoda de a manipularea a colectiilor java prin metodele sale specifice precum filter , clolect , distinct , map . Un alt concept important utilizat sunt expresile lambda ce ofera functionalitatea astementatoare unui metode astfel putand avea un cod mai vizibil . O interfata utila este cea de serializare a obiectelor astfel putand salva datele in mod permantent in fisiere si astfel putand fi utilizate dupa inchidera aplicatiei sau utilizatea in cadrul altor aplicatii .