Restaurant Management System

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

COORDONATOR PROIECT:DORIN MOLDOVAN

STUDENT: Rus Mihai-Tudorel

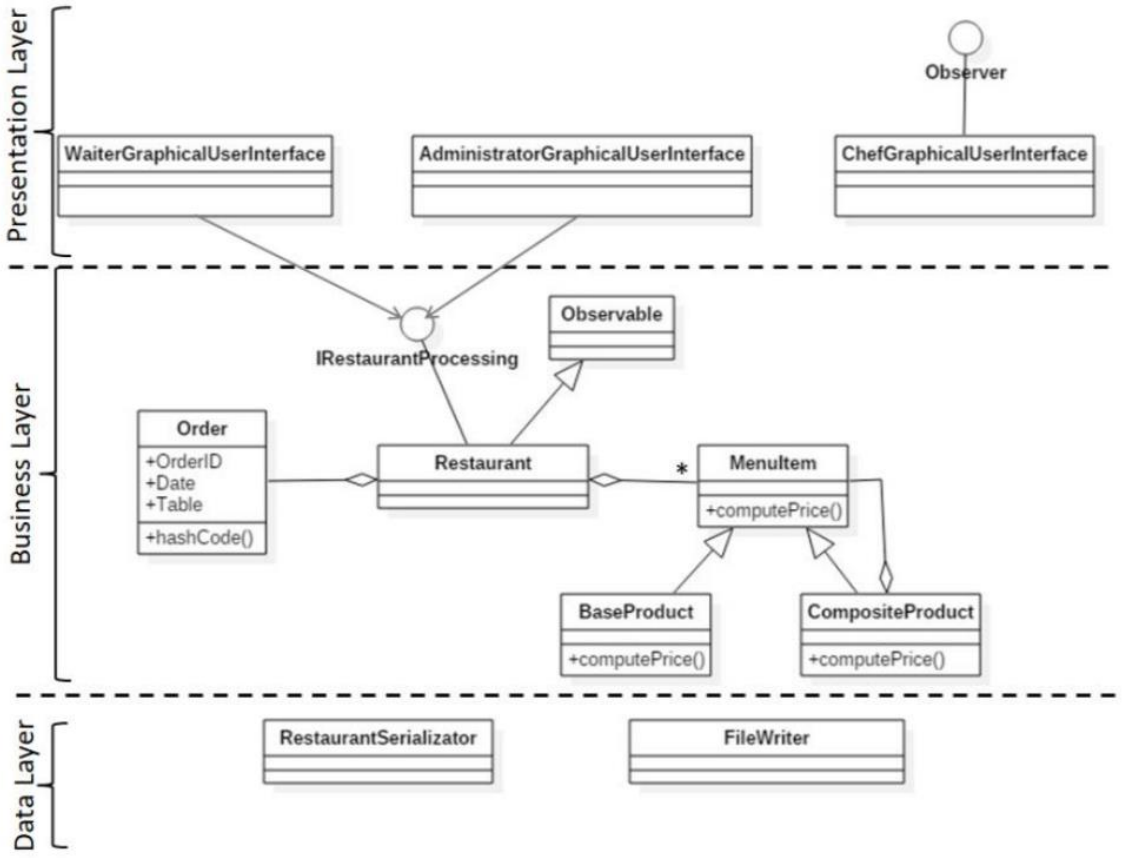
GRUPA : 30221

CUPRINS

I. Specificație

- Cerința

- Obiectivul acestei teme este implementarea unei aplicatii (unui sistem) **RestaurantManagement** pentru gestionarea unui restaurant. Sistemul are aceste trei tipuri de utilizatori: administrator, waiter si chef. **Administratorul** poate adauga, elimina si modifica produse existente in meniu. **Waiter**-ul poate crea o onua comanda cu elemente existente deja in meniu si sa calculeze nota de plata a comenzii. **Chef**-ul este notificat de fiecare data cand trebuie sa gateasca ceva ce a fost comandat prin inteermediul **waiter**-ului.
- Sistemul de clase respecta diagrama de mai jos:



II. Proiectarea și implementarea

In ceea ce priveste structurarea codului **java**, acesta este impartit in mai multe packages (architectural layers) pentru o mai buna intelegere a codului si pentru lizibilitate. Aceste packages sunt :

- Business: care contine clasele Order,Restaurant si RestaurantProcesing;
- Menu:care contine clasele Produs,CompositeProduct si MenuItem;
- Presentation care contine clasele:Bucatar,Client,Ospatar si Observer;

Cand se ruleaza aplicatia, apar pe ecran trei ferestre: fereastra pentru **Client ,**fereastra pentru **Ospatar** , si fereastra pentru **Bucatar.**

In cazul in care dorim sa:

- adaugam un nou produs un meniu
- editam pretul unui produs deja existent in meniu
- stergem un produs deja existent in meniu

atunci, ne vom folosi doar de ferestra CLIENT.

In cazul in care dorim sa:

- facem o noua comanda
- calculam nota de plata
- afisam nota de plata intr-un fisier .txt extern

atunci, ne vom folosi doar de fereastra OSPATAR.

In momentul in care adaugam in meniu un produs compus (adica un produs care este format din mai multe produse de baza din meniu) este notificat Bucatarul, adica va aparea pe fereastra sa un mesaj pop-up cu produsul care urmeaza sa fie gatit.

Ferestrele Client si Ospatar au ambele butoane de ~Help!~ in crae sunt explicate instructiunile de folosire ale ferestrelor.

De fiecare data cand scriem ceva intr-una dintre text field-uri, trebuie sa ne asiguram ca apasam tasta ENTER, iar apoi butonul de procesare, deoarece altfel valoare din text field nu va fi actualizata.

Exista doua coloane, si anume ~Produs ~ si ~Prêt sau Componente~.

- In momentul in care dorim sa adaugam un produs de baza in meniu, scriem numele produsului sub coloana cu numele ~Product name~ si pretul acestuia sub ~Price or Components~.
- In momentul in care dorim sa adaugam un produs compus in meniu, trebuie sa ne asiguram prima data ca am adaugat in meniu toate produsele baza din care este compus in prealabil. Dupa ce ne- am asigurat de acest lucru, vom introduce sub coloana ~Product name~ numele produsului, dupa care vom introduce sub coloana ~Price or Components~ numele fiecrui produs de baza din care este alcatuit (separate prin ", " virgula si spatiu).
- In momentul in care dorim sa editam un produs din meniu, scriem numele produsului sub coloana cu numele ~Product name~ si noul pret al acestuia sub ~Price or Components~.
- In momentul in care dorim sa stergem un produs din meniu, scriem sub ccoloana ~Product name~ nunele produsului pe care dorim sa il stergem din meniu.

Exista trei coloane, si anume ~Date~, ~Table~ si ~Ordered Items~.

- In momentul in care dorim sa adaugam o noua comanda, scriem data zilei, numarul mesei pentru care se face comanda si produsele comandate (separate prin ", " virgula si spatiu).
- In momentul in care dorim sa se calculeze nota de plata, scriem numarul de ordine al comenzii, iar rezultatul va fi afisat in consola; daca dorim sa generam o nota de plata intr-un fisier .txt, rezultatul va fi afisat in "billx.txt", unde txt este numarul de ordine al comenzii.

III. Lista de clase

Pentru a respecta principiile fundamentale ale OOP, programul conține mai multe clase:

- Order
- Restaurant
- Restaurant Processing
- CompositeProduct
- MenuItem
- Produs
- Bucatar
- Client
- Ospatar
- Observer

IV. Justificarea soluției alese

Am optat pentru această soluție de implementare deoarece mi s-a părut o modalitate atât ușoară, cât și eficientă pentru implementarea aplicatiei. Folosirea stărilor intermediare prin care trece programul ne ajută atât la simplificarea soluției, cât și la implementarea eficientă a aplicatiei.

V. Utilizarea și rezultatele

De fiecare data cand se porneste aplicatia, meniul va fi gol (similar unei baze de date nepopulate). Acesta trebuie populat prin efectuarea comenzii "Add MenuItem". Daca se doreste generarea unei note de plata, atunci rezultatul va fi afisat intr-un fisier .txt cu numele "billx.txt", unde x reprezinta numarul de ordine al comenzii respective.

VI. Posibilități de dezvoltare ulterioară

Consider ca, o dezvoltare ulterioara ar putea fii aceea de a adauga si o simulare in timp real a cozilor,nu in timp masurat in secunde ci, simularea cat mai realistica, cu aparitia clientilor la aceleasi momente de timp,cu intercalarea timpului,practic o aplicatie de simulare apropiata cu realitatea.

VII. Concluzii

Concluziile includ faptul că o planificare bine gândită a claselor ușurează lucrurile și economisește timp. Odată cu creșterea dificultății în proiecte, trebuie abordată problema cu o perspectivă inteligentă și nu doar să sari imediat la scrierea codului. Astfel, începând cu diagramele UML mi-am dat timp să reflectez pentru abordarea adecvată a implementării programului dorit.

Îmbunătățirile care ar putea fi realizate în cadrul proiectului includ o ușoară modificare, astfel încât calculatorul acceptă intrări float, care ar putea fi ușor realizate prin modificarea matcherelor din funcție care transformă intrarea utilizatorului șirului în obiectul respectiv. Într-o anumită măsură...

VIII. Bibliografie

http://docs.oracle.com/javase/8/docs/technotes/guides/language/assert.html

http://javarevisited.blogspot.ro/2012/01/what-is-assertion-in-java-java.html

http://stackoverflow.com/questions/11415160/how-to-enable-the-java-keywordassert-in-eclipse-program-wise

https://intellij-support.jetbrains.com/hc/en-us/community/posts/207014815-How-toenable-assert