Documentatie Tema 4

**1 . Obiectivul temei**

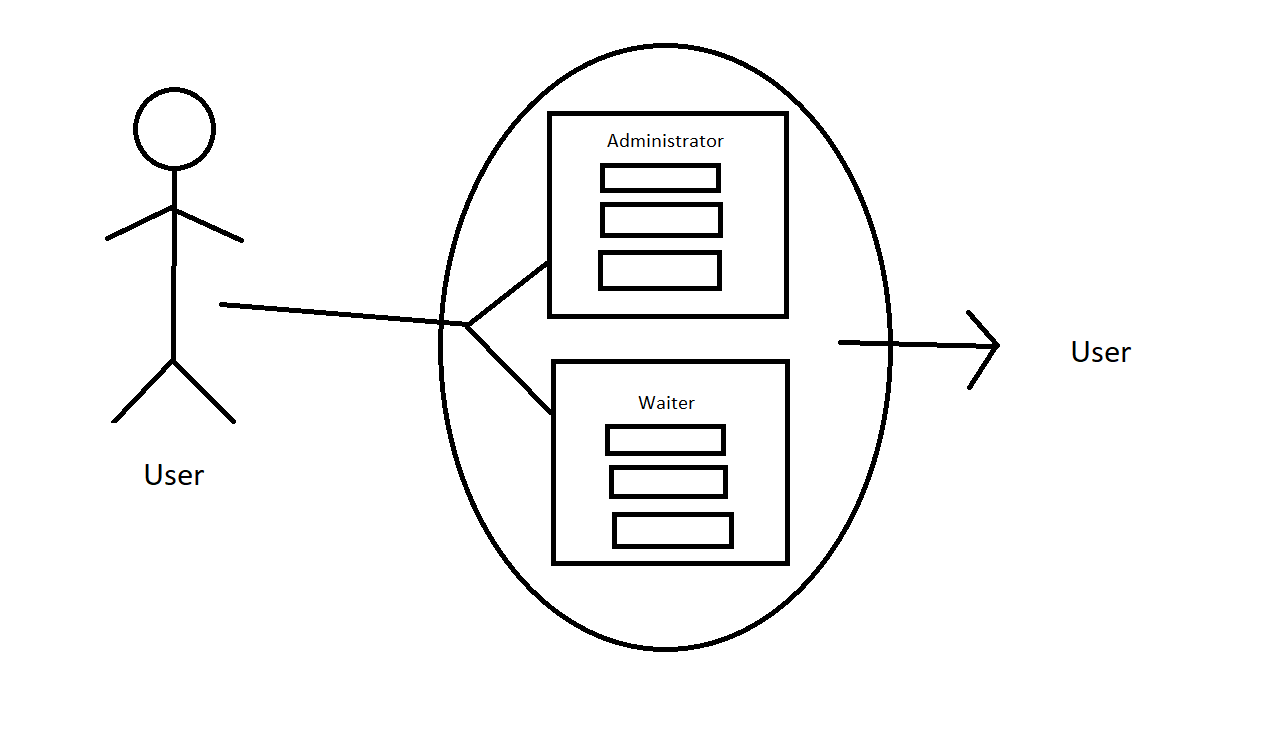
Obiectivul aceste teme este a crea o interfata pentru o program care manageriaza un restaurant . In restaurant sunt trei tipuri de oameni : Waiter, Administrator si Chef . Waiter-ul este cel care se ocupa cu luatul comenzilor si procesarea acestora . Administratorul se ocupa de Menu . Acesta poate sa adauge produse noi in meniu, poate sa scoata produse din meniu si poate sa modifice produse din Meniu . Bucatarul va primi ordine de a gati atunci cand cineva face o comanda .

**2 . Analiza problemei**

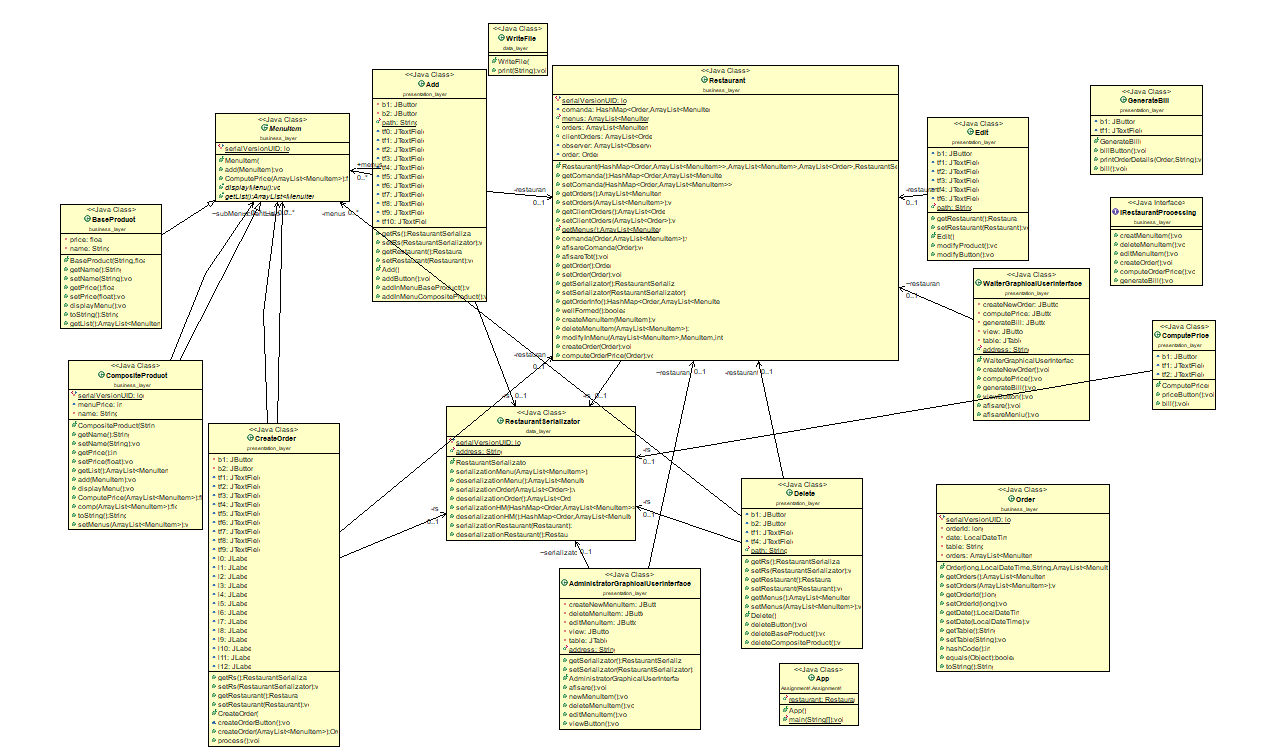
***Pasi de utilizare :***

1. Se deschide Command Prompt-ul in locatia fisierului . jar .
2. Pentru a rula se introduc urmatoarele : “ java –jar fisier . jar ” unde in loc de fisier se va scrie numele fisierului . jar, in cazul meu „ task . jar ” .
3. Dupa ce se ruleaza vor aparea doua ferestre : Waiter window si Administrator window . Acestea au fiecare cate 4 butoane . Prin apasarea butoanelor se vor descrie alte ferestre pentru a executa operatie ceruta . Dupa ce se completeaza casutele cu valorile potrivite, se poate apasa de butonul de validare . Pentru a verifica functionalitatea acestora se verfica prin apasarea butonului „ View ” . Se va deschide un tabel cu informatiile necesare .
4. De fiecare data cand se inchid ferestrele datele noi modificate sau adaugate se vor sterge, asa ca nu este indicat sa inchideti ferestrele daca nu vreti sa pierdeti informatiile noi adaugate .
5. Pentru a parasi programul, se apasa x pe ambele ferestre .

**2.2 Diagrama USE CASE**

****

**3 . Proiectare**

**3 . 1 . Diagrama UML**

**3 . 2 . Explicatii clase**

* Clasa App : contine metoda main . Aici se deserializeaza fisierul restaurant . Aici se creeaza obiectele de tip Administrator si Waiter .
* Clasa BaseProduct : clasa care contine un produs de baza .
* Clasa CompositeProduct : clasa care are un produs compus din mai multe produse de baza .
* Clasa MenuItem : clasa abstracta pentru BaseProduct si CompositeProduct .
* Clasa Order :clasa in care se retine informatia legata de comenzi .
* Clasa Restaurant : este clasa care sta la baza functionalitatii programului . Aici sunt implemetate cele mai multe metode si tot aici se retine informatia din toate clasele . Adica un obiect restaurant detine informatii despre Meniu, Comenzi, Produse etc…
* Clasa Restaurant Serializator este clasa in care se serializeaza si se deserealizeaza obiecte de tip ArrayList<MenuItem>, ArrayList<Order>, HashMap<order, ArrayList<MenuItem>> si Restaurant .
* Clasa WriteFile – este o clasa pentru folosita pentru scrierea intr-un fisier .
* Clasa AdministratorGraphicalUserInterface- clasa care creeaza interfata pentru Administrator .
* Clasa Add- clasa prin care se adauga elemente noi in meniu .
* Clasa Delete- clasa prin care se sterg elemente din meniu .
* Clasa Edit- clasa prin care se modifica elemente din meniu .
* Clasa WaiterGraphicalUserInterface – clasa care creeaza interfata pentru Waiter .
* Clasa CreateOrder- clasa care adauga comenzi noi in table .
* Clasa ComputePrice- clasa in care se calculeaza pretul unui produs/ meniu .
* Clasa GenerateBill – clasa care genereaza un fisier de tip txt pentru o anumita comanda .
* Clasa View- clasa care creeaza tabel pentru Meniul restaurantului in Admin window si table pentru comenzi in Waiter window .

**4 . Metode, variabile clase**

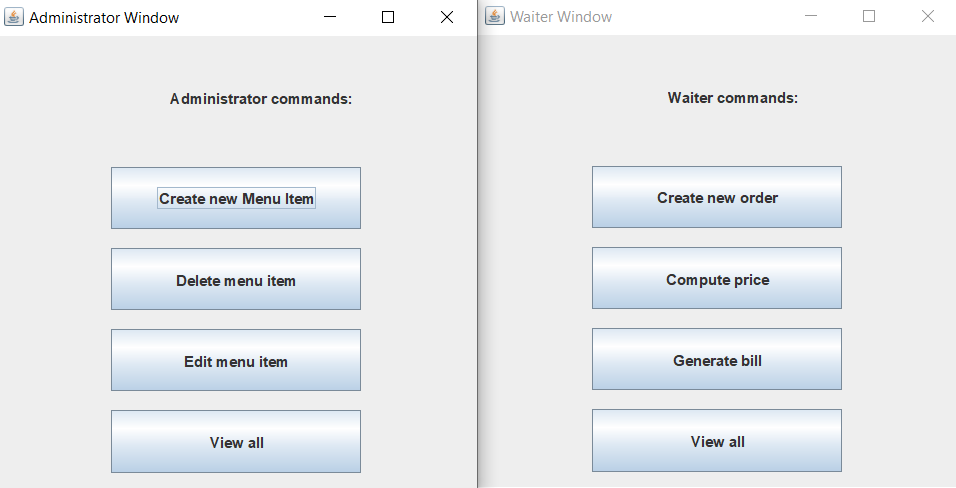
* Clasa App- aici de declara obiectul de tip Restaurant, tot aici se deserializeaza acesta si se creaza obiecte de tip Administrator si Waiter .
* Clasa BaseProduct : are ca si variabile instanta : String nume si float pret . Extinde interfata Serializable si Clasa abstracta MenuItem . Are o metoda getList() care returneaza obiectul sub forma unei liste de BaseProduct .
* Clasa CompositeProduct : are ca si variabile instant numele( String nume ) . Un composite product poate contine unu sau mai multe BaseProduct sau unul sau mai multe CompositeProduct . Tot aici se calculeaza pretul in functie de componenta sa cu ajutorul metodei ComputePrice() . Pe langa setteri si getteri mai are o metoda getList() care returneaza o lista cu toate componentele sale . De asemenea, la fel ca in cazul lui BaseProduct acesta extinde clasa abstracta MenuItem si implementeaza interfata Serializable .
* Clasa MenuItem : clasa abstracta . Obiectele de acest tip pot fi ori BaseProduct ori CompositeProduct .
* Clasa Order : aceasta clasa are ca variabile instanta :long Order Id, LocalDateTime date, String table, ArrayList < MenuItem > orders .
* Clasa Restaurant : clasa care are variabilele instanta : HashMap<Order, ArrayList < MenuItem >> comanda, ArrayList<MenuItem> orders, ArrayList < Order > clientOrders , RestaurantSerializator restaurantSerializator . Aceasta clasa implementeaza metodele impuse de interfata iRestaurantProcessing : public void creatMenuItem ( ); public void deleteMenuItem( ) ; public void editMenuItem( ); public void createOrder( ); public void computeOrderPrice( ); public void generateBill( ) .
* Clasa RestaurantSerialization . Aceasta clasa se ocupa de serializarea si deserializarea anumitor date . Contine mai multe metode : serializationMenu ( ) , deserializationMenu ( )

Pentru stocarea Meniului din restaurant . serializationOrder ( ), deserializationOrder ( ) pentru stocarea listei de comenzi . serializationHM(), deserializationHM() pentru stocarea structurii HashMap <Order, ArrayList<MenuItem>> . SerializationRestaurant ( ) si deserializationRestaurant( ) pentru stocarea obiectului de tip restaurant .

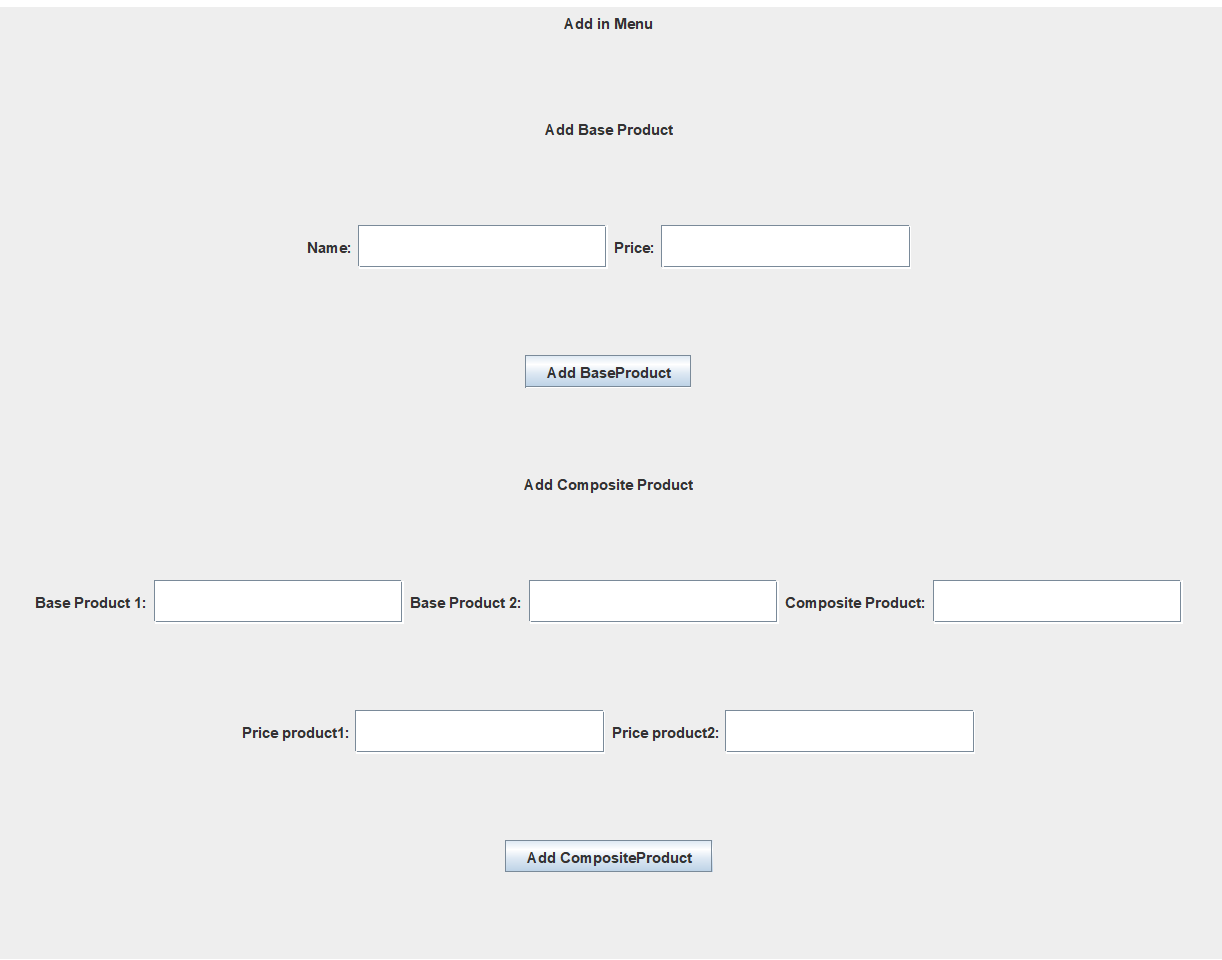
* Clasa WriteFile- scrie intr-un fisier un anumit string dat prin metoda Print( ) .
* Clasa AdministratorGraphicalUserInterface- aici se creeaza toate elementele care formeaza interfata pentru Administrator . Interfata contine 4 butoane care duc catre alte 4 interfete .
* Clasa Add- clasa care creeaza noi produse in meniul restaurantului . Aici se definesc elementele care formeaza interfata . Contine doua butoane …fiecare dintre ele avand un rol de validare . Unul este folosit pentru a valida in BaseProduct, iar celelalt valideaza un obiect de tip CompositeProduct ( ) . Pentru a crea un obiect de tip BaseProduct trebuie sa scrie un nume si un pret, iar pentru a crea un obiect de tip Composite product trebuie sa scrie numele a doua BaseProducts si pretul acestora . Dupa ce s-a scris in casutele corespunzatoare se apasa pe butoanele de validare .
* Clasa Delete- clasa care sterge elemente din meniul restaurantului . Aici se defines elemente pentru crearea interfetei, astfel ca interfata prezinta doua casute de text si doua butoane de validare . Pentru a sterge in BaseProduct se scrie in prima casuta numele produsului care vrea sa fie sters din meniu, iar in a doua casuta se scrie numele produsului de tip CompositeProduct .
* Clasa Edit- clasa care modifica produsele din meniul restaurantului . Aici de declara elementele care alcatuiesc interfata . Se poate modifica un Base Poduct sau Un base product dintr-un CompositeProduct . Se introducele numele produsului, numele nous i pretul nou si se valideaza .
* Clasa WaiterGraphicalUserInterface- clasa in care se creaza interfata pentru Waiter . La fel ca la interfata pentru Administrator, aceasta interfata contine 4 butoane, fiecare avand functionalitate proprie .
* Clasa CreateOrder- clasa care adauga un nou order in lista de comenzi . Tot in aceasta clasa se defines elemente care formeaza interfata . Pentru a introduce o comanda noua trebuie sa se scrie numele Meniului/ Produsului dorit , id-ul clientului si masa de la care s-a luat comanda .
* Clasa ComputePrice() – aceasta clasa calculeaza pretul pentru un id client si scrie in interfata valoarea rezultata .
* Clasa GenerateBill() – aceasta clasa genereaza nota de plata pentru clientul cu id-ul ales . Interfata contine textfield in care se introduce id-ul clientului, iar ca rezultat, un fisier . txt se va forma la adresa aleasa .
* Clasa View- este clasa care creeaza un JTabel cu informatia din meniul restaurantului si comenzile din acesta. Aceasta contine un buton, care in functie de unde este apasat, se deschide un alt table, astfel ca daca este apasat din fereastra Administrator va aparea tabelul cu meniurile din restaurant, iar daca este apasat din Waiter va apare lista cu toate comenzile din restaurant.

**5.Rezultate**

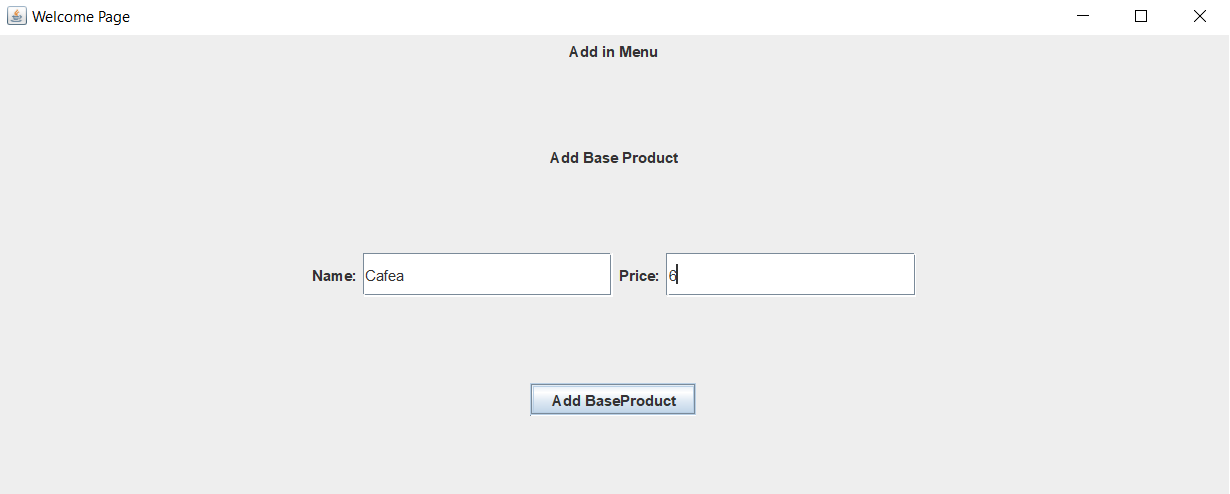
La inceput am salvat informatiile in fisier.ser pentru a putea sa le reutilizez. Astfel ca in restaurant avem deja 5 obiecte : MENU1, MENU2, MENU3, cirese si cartofi. Initial interfete vor arata astfel :



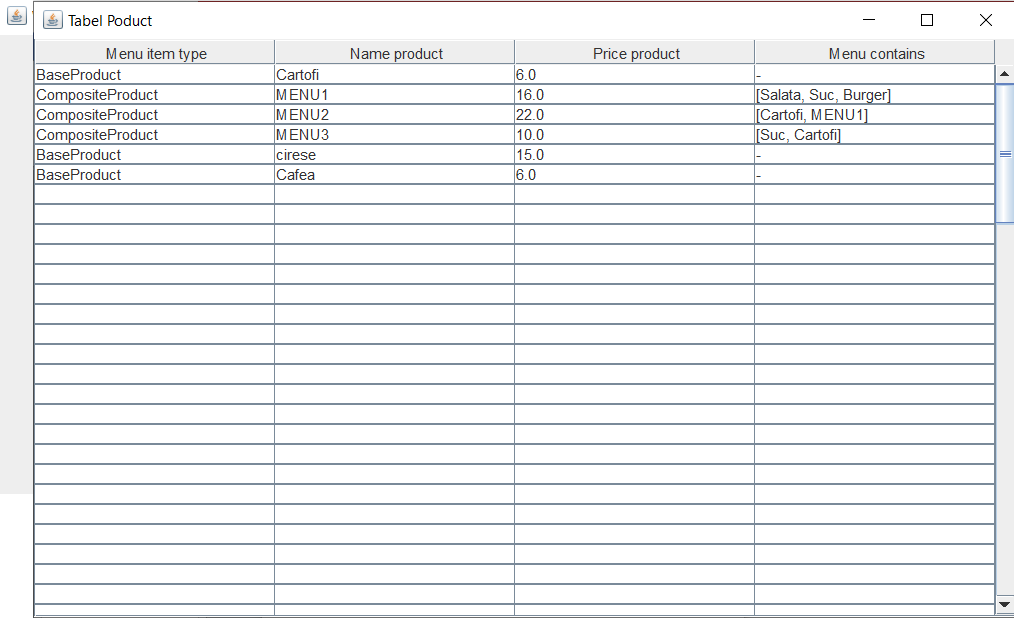
Fereastra Administrator contine 4 butoane : Create new Menu Item, Delete menu Item, Edit menu item si view all . Sa presupunem ca apasam butonul Create new Menu Item . Va aparea fereasta urmatoare :



Avem 2 parti importante1 : zona pentru BaseProduct ( cea de sus ) si zona pentru CompositeProduct ( cea de jos ) . La add Base Product avem 2 casute de text in care va trebui sa scriem numele si pretul . Dupa ce s-au scris se poate apasa pe butonul Add BaseProduct . Vom introduce un BaseProduct pentru a se observa cum functioneaza :

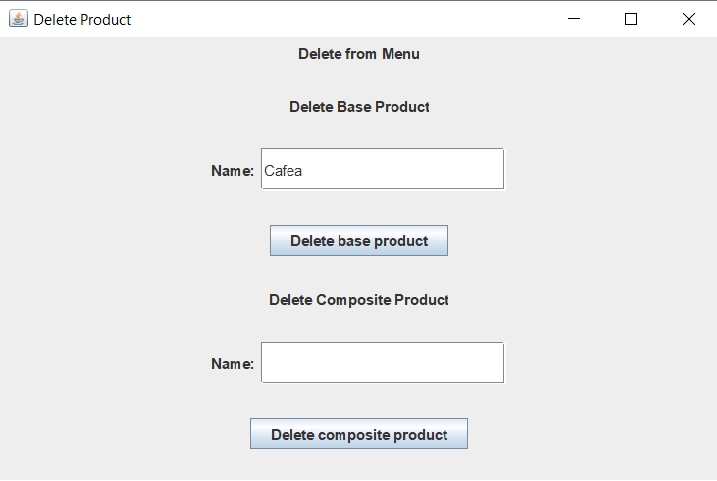


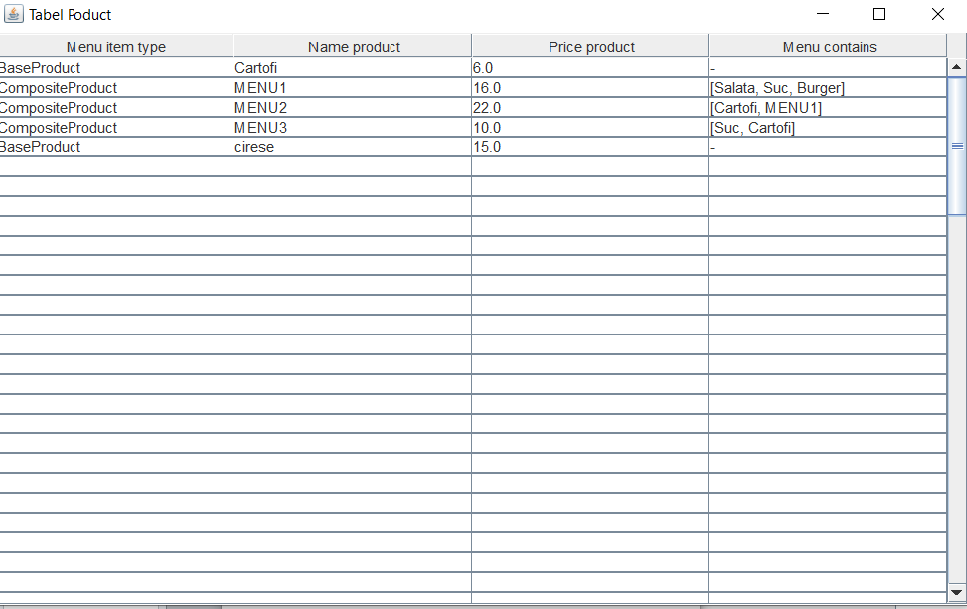
Am scris in casuta pentru nume : “Cafea” si in casuta pentru pret : “6” si am apasat Add BaseProduct . Dupa ce s-a apasat, se poate inchide fereastra si se apasa pe view all din fereastra Administrator .



In table se poate observa ca ultimul produs este de tip BaseProduct , numele produsului “Cafea” si pretul 6 .

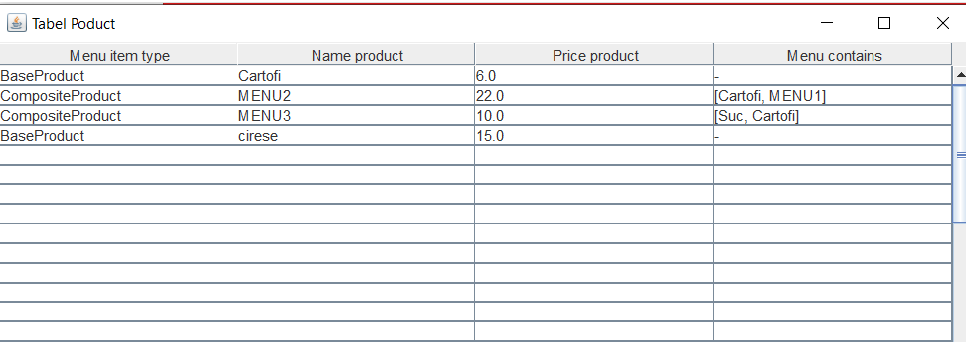
Pentru a sterge ceva din Menu se apasa pe Delete Menu Item .

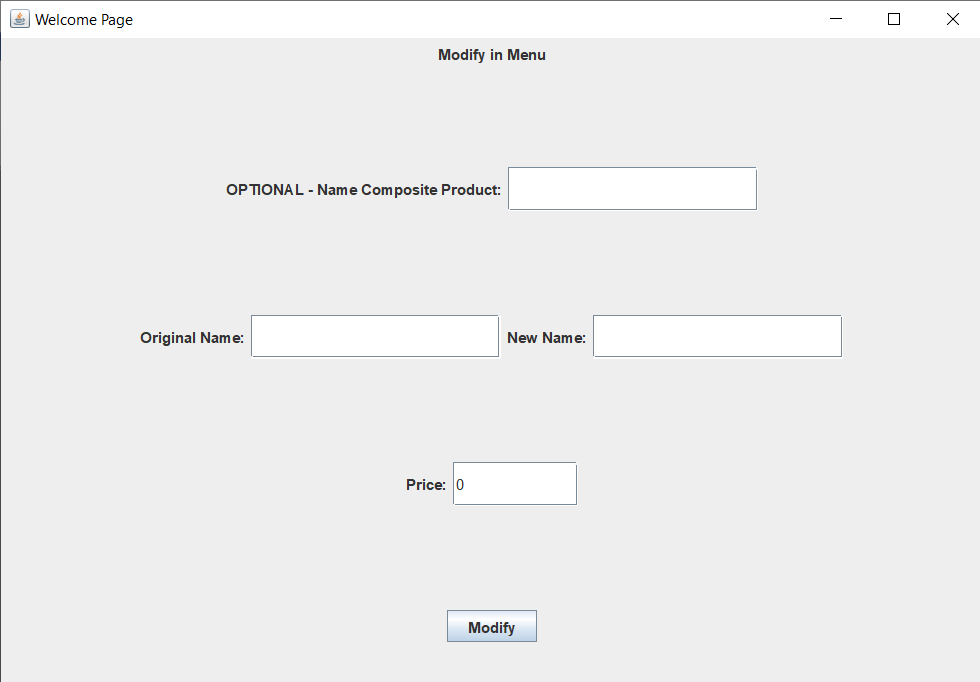


Acum vom sterge Cafeaua din Menu . Apasam Delete base product si mergem din nous a vizualizam tabelul .

Cafeaua a disparut din tabel .

Acum vom sterge un CompositeProduct, astfel ca apasam din nou pe delete menu item si completam in partea de jos . Stergem MENU1 :



Urmatoarea actiune este editarea :

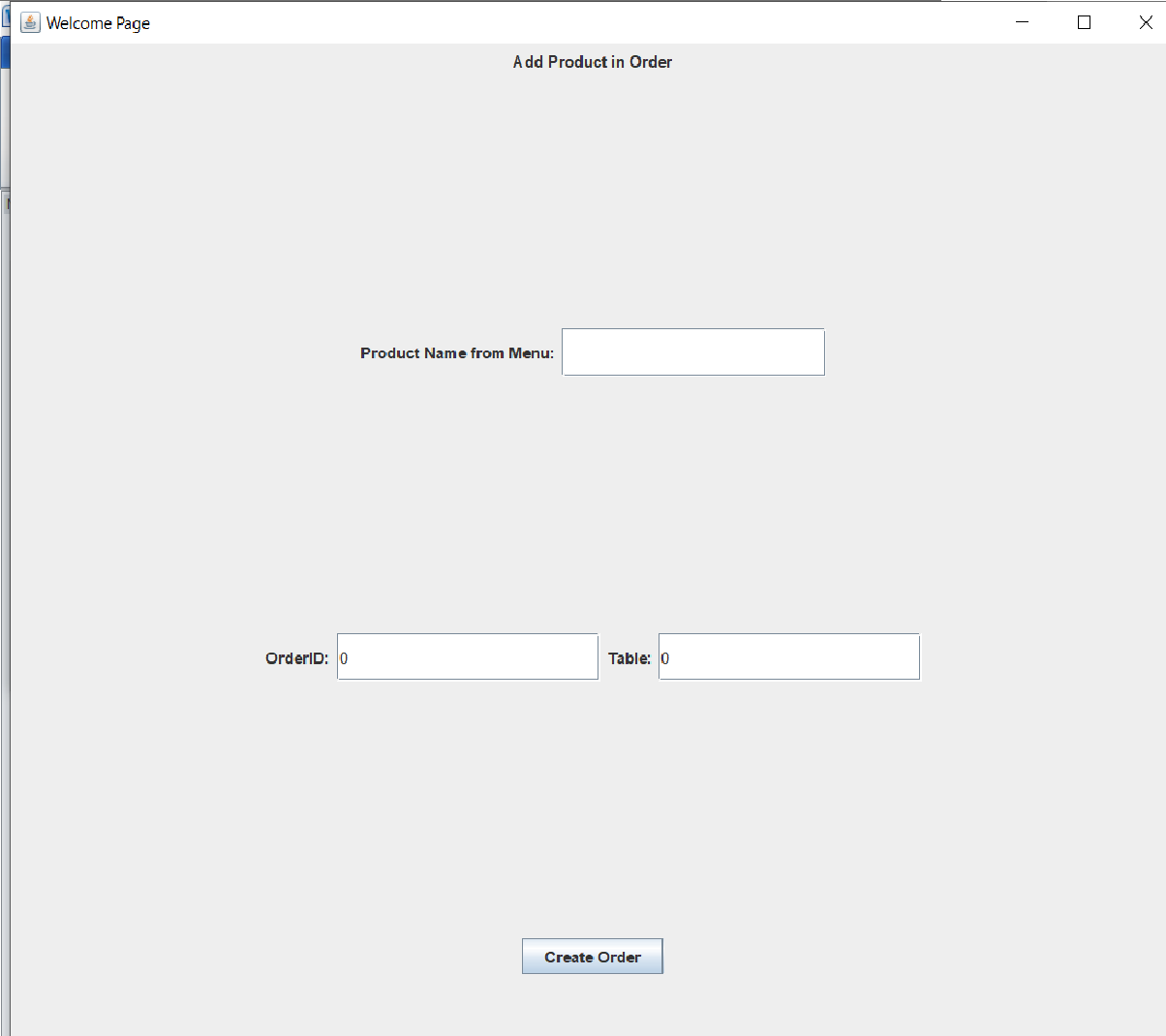
Prima casuta de text trebuie completata doar cand vrem sa modificam un BaseProduct dintr-un CompositeProduct . Eu o sa vreau sa modific schimb cartofii din MENU3 cu salata si o sa ii puna valoare de 15 . Completam in ordine : “ MENU3 ”, “ Cartofi ”, “ Salata ” si la price 15 si apasam Modify .



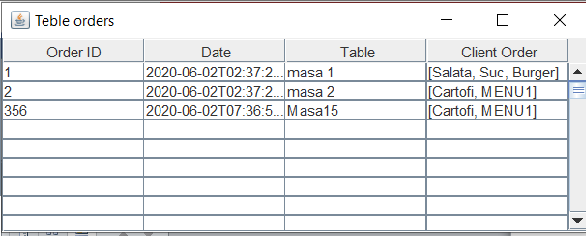
In locul cartofilor a aparut salata, iar pretul s-a modficat din cate se poate observa . Pentru un BaseProduct se procedeaza la fel doar ca se lasa necompletata prima casuta .

Trecem mai departe la Waiter . Acesta poate sa adauga comenzi, sa calculeze pretul sis a elibere note de plata sub forma de txt .

Prima instructiune este CreatNewOrder :



Cand se apasa acest buton se va deschide aceasta fereastra plus fereastra cu meniurile pentru a putea vedea ce Meniuri sunt disponibile . Eu voi alege MENU2, voi pune id-ul 356 si Table :”Masa 15” .

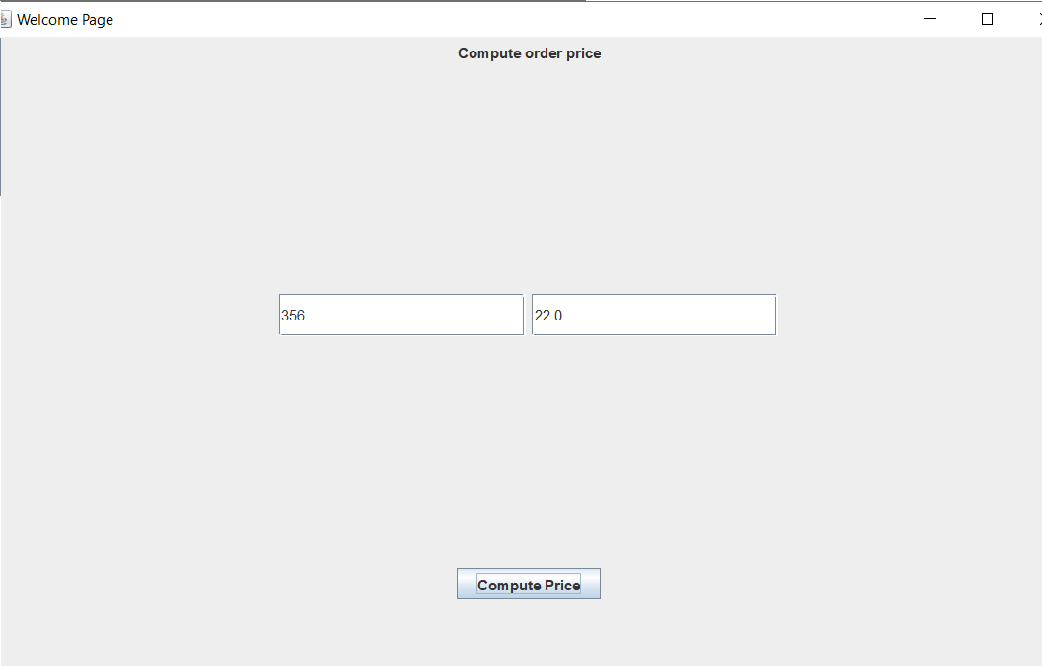


A aparut o noua comanda cu id 356 la Masa 15 cu MENU2 care contine cartofi si MENU1 .

Urmatoare instructiune este ComputePrice si cam asa arata interfata :

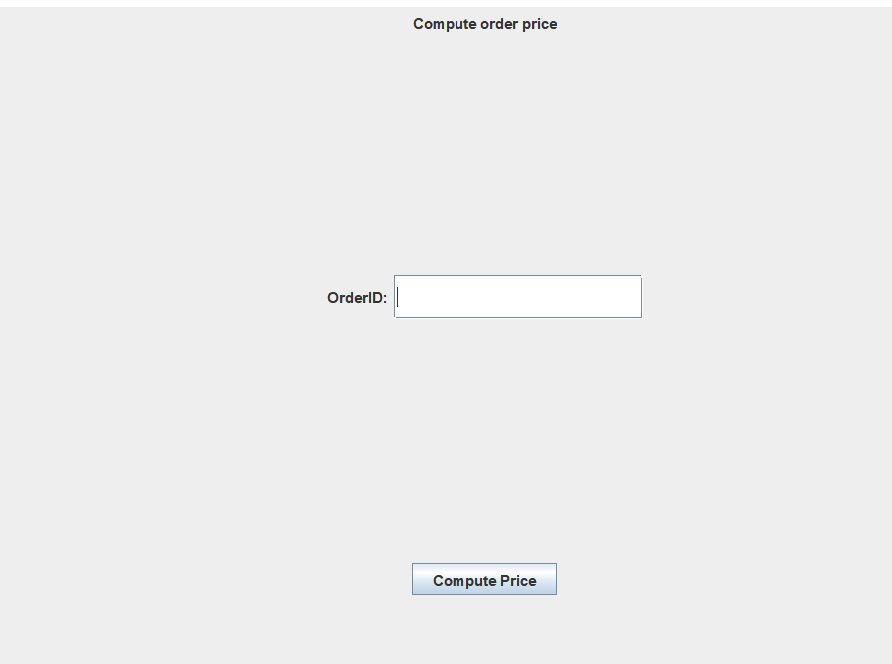


In prima casuta se pune id-ul si in a doua va aparea pretul . Eu voi pune id-ul pe care l-am introdus adineauri, 356 .

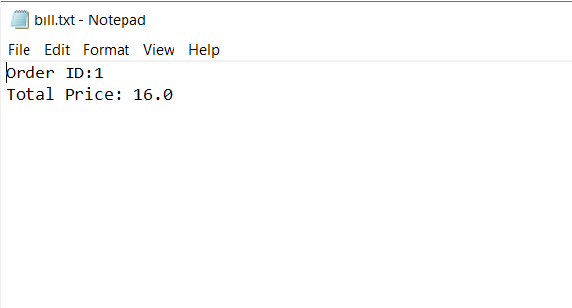


Pentru Id-ul 356 valoare comenzii este de 22 . 0

Urmatoare instructiune GenrateBill este un generator de bonuri pentru comenzi . Interfata acestuia arata astfel :



Pentru a genera un bon trebuie introdus id-ul comenzii in casuta de text . Eu o sa introduce 1 pentru a vedea care este valoare comenzii .



Pentru id-ul comenzii 1 s-a generat o suma de 16 care trebuie achitata . Asta este in mare functionarea programului .

6 . Concluzii

Aceasta a fost cea mai complexa tema de pana acum si totodata si cea mai grea din punctual meu de vedere pentru ca au fost foarte multe clase de implementat si cumva a fost la latitudinea noastra sa ne alegem ce sa punem in clase. Am invatat sa folosesc interfata Serializable si sa creez fisier. Ser . Am invatat sa fac o stuctura Composite Design Pattern pentru BaseProduct si CompositeProduct . Mi-am imbunatatit skill-urile cand vine vorba de interfete .

Imbunatatiri ulterioare :Desigur, ca oricare program, si acesta are nevoie de unele imbunatatiri, cum ar fi posibilitatea de introduce mai mult de 2 BaseProduct intr-un CompositeProduct in interfata . Actualizarea fisierelor de serializare pentru a nu pierde datele dupa momentul opririi aplicatiei . Interfata ar putea sa fie construita in Modelul MVC . Realizarea interfetei pentru chef si implementarea Observer Design Pattern-ului.O aranjare mai frumoasa a tabelului si al interfetelor .

7. Bibliografie

<http://www.tutorialspoint.com/java/java_serialization.html>

<https://www.baeldung.com/java-https://docs.oracle.com/javase/8/docs/api/java/io/Serializable.html>

<http://javarevisited.blogspot.ro/2011/02/how-hashmap-works-in-java.html>

<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>