**Restaurant Management System**

**Moldovan Vlad-Madalin**

**CUPRINS:**

**1.Obiectivul temei**

**2.Analiza problemei, modelare, scenarii , cazuri de utilizare**

**3. Proiectare (decizii de proiectare, diagrame UML, structure de date, proiectare clase, interfete, relatii , packages, algoritmi, interfata utilizator)**

**3.1 Decizii de proiectare**

**3.2 Diagrama UML**

**3.3 Packages**

**4.Implementare**

**4.1 Clasa BaseProduct**

**4.2 Clasa CompositeProduct**

**4.3 Clasa IProcessingRestaurant**

**4.4 Clasa Restaurant**

**4.5 Clasa MenuItem**

**4.6 Clasa Order**

**4.7 Clasa FileWrite**

**4.8 Clasa RestaurantSerializator**

**4.9 Clasa AdministratorGUI**

**4.10 Clasa ChefGUI**

**4.11 Clasa WaiterGUI**

**4.12 Clasa RestaurantManagementGUI**

**4.13 Clasa Start**

**5.Concluzii**

**6.Bibliografie**

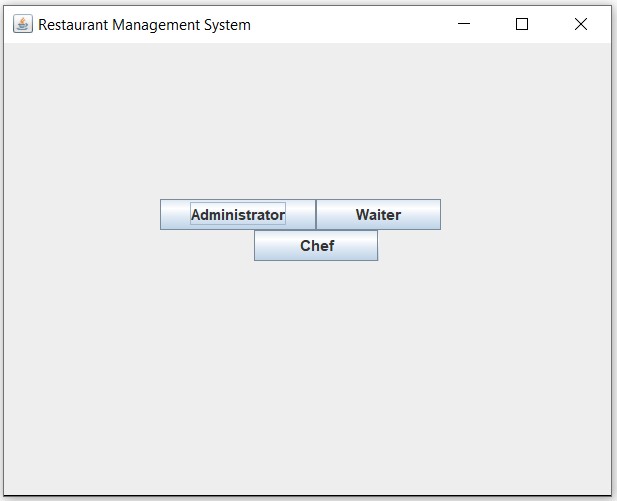
**1.Obiectivul temei**

Obiectivul principal al temei este de a implementa un Management System pentru un restaurant care sa contina operatii pentru Administrator ( adaugarea unui nou produs in meniu , stergerea unui produs din meniu , modificarea unui produs din meniu ) , Waiter ( crearea unei noi comenzi , adaugarea de produse pentru o comanda , generarea notei de plata pentru o comanda) , Chef ( afisarea produselor care trebuie sa fie gatite ) . Obiectivul secundar al temei este de a implementa o interfata grafica pentru aceasta aplicatie si de a folosi serializarea pentru salvarea si incarcarea datelor din restaurant .

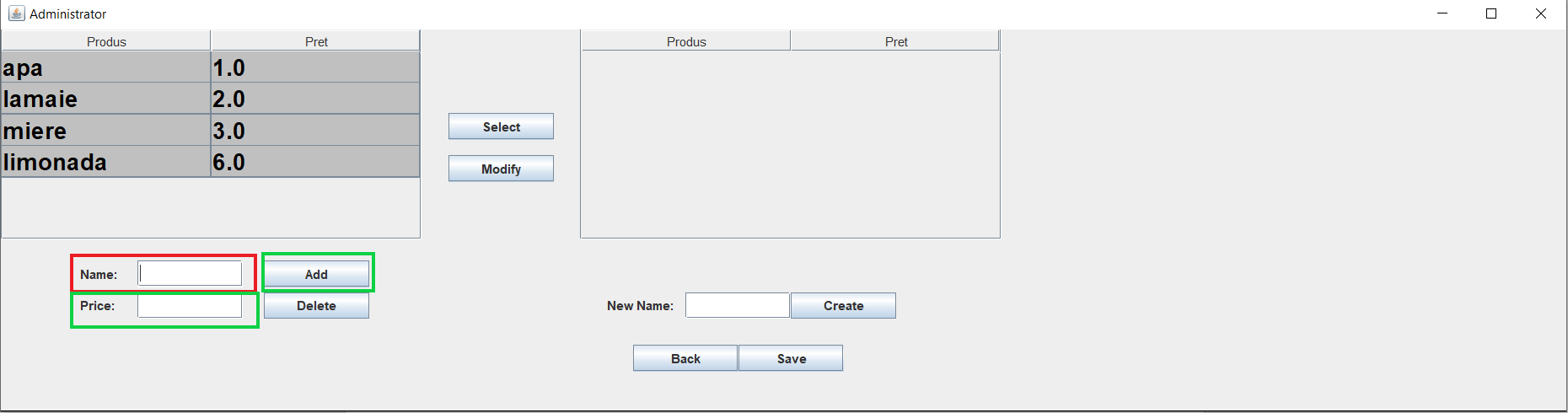
**2.Analiza problemei, modelare, scenarii, cazuri de utilizare**

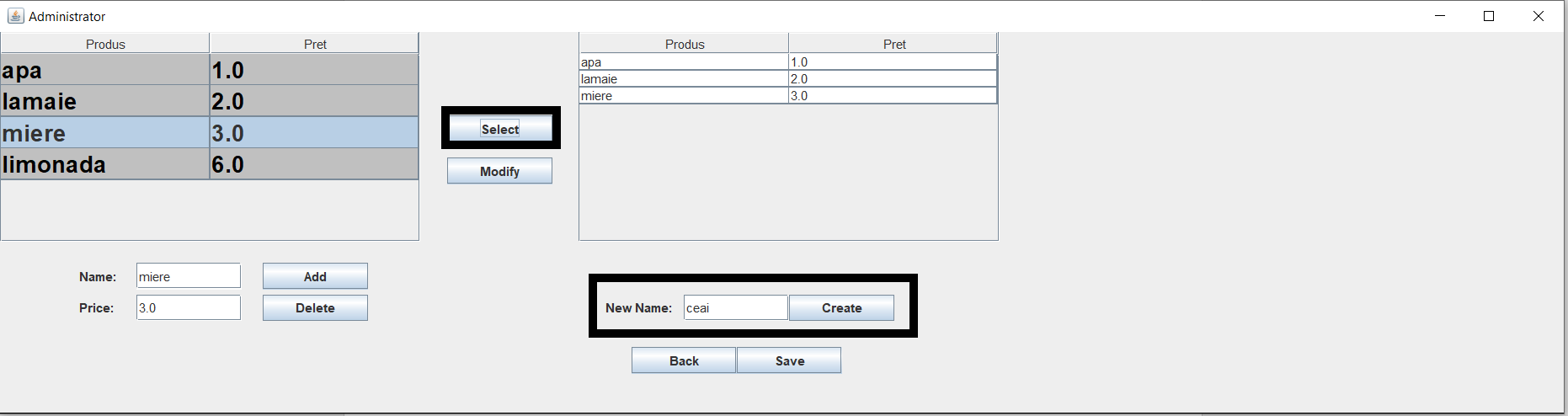
Cerinta problemei este de a implementa operatiile pentru Administrator , Waiter , Chef prezentate anterior intr-o interfata grafica pentru o utilizarea mai usoara a aplicatiei .

La deschiderea aplicatie se va alege tipul de utilizator dorit.

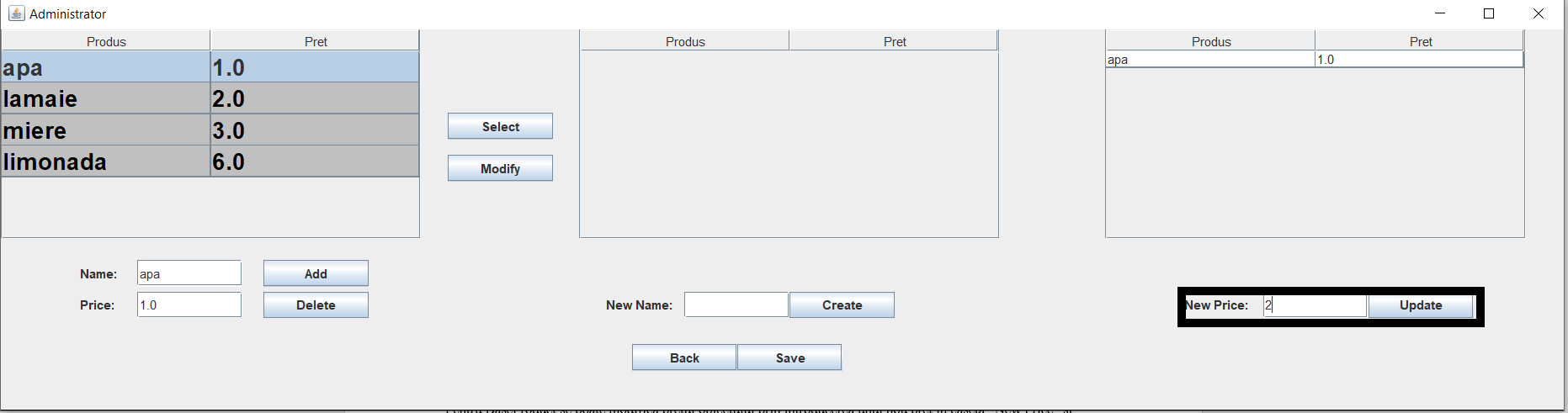


In continuare voi descrie utilizarea operatiilor disponibile pentru administrator.Daca se doreste adaugarea unui nou produs in meniu se vor introduce numele si pretul noului produs si se va apasa butonul “Add” .

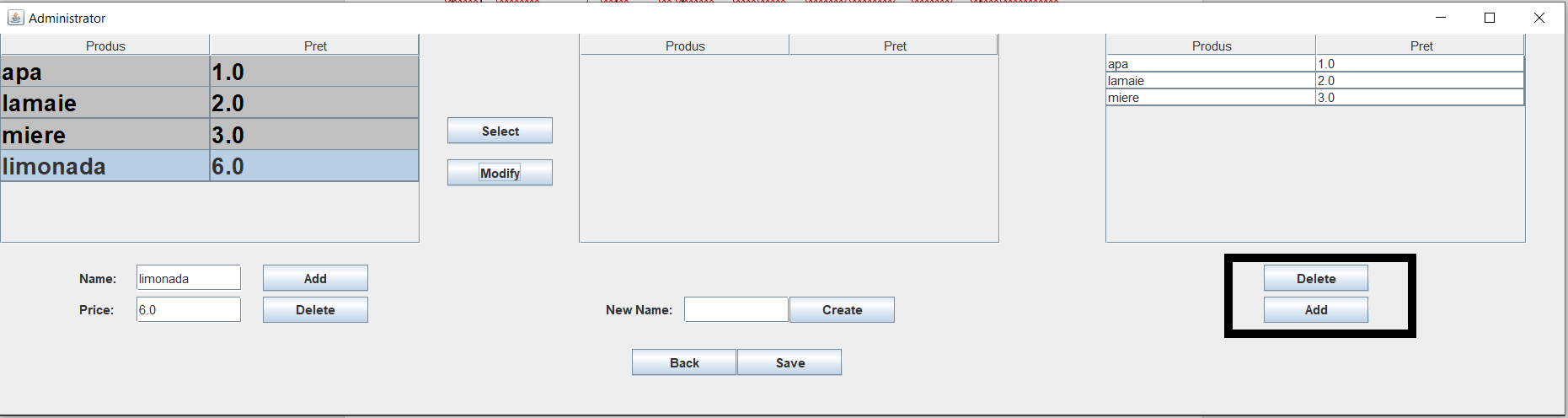


Daca se doreste adaugarea unui produs format din mai multe produse ( compositeProduct ) atunci se vor selecta produsele din primul tabel si se va apasa pe butonul select ( click pe un produs dupa apasati pe butonul “Select” si tot asa pana ati introdus toate produsele dorite , acestea vor aparea in cel de al doilea tabel ) dupa care se va introduce numele obiectului in castea cu numele “New Name” si se va apasa pe butonul “Create” , iar noul produs creat va aparea in primul tabel .

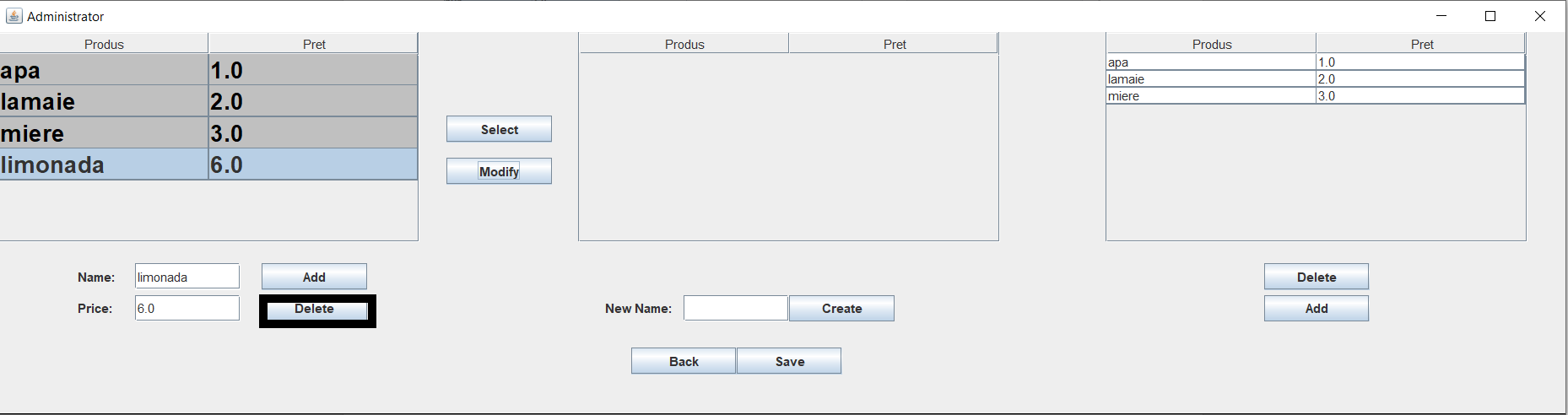
Daca se doreste modificarea unui produs se va selecata produsul din primul tabel ( click pe acesta ) dupa care se va apasa pe butonul “Modify” dupa care va aparea un nou tabel cu diferite butoane in functie de tipul obiectului care a fost selectat ( BaseProduct , CompositeProduct ).

Pentru BaseProduct se poate modifica pretul obiectului prin introducerea unui nou pret in caseta “New Price” si apasarea butonului “Update” (dupa apasarea in primul tabel se va modifica pretul obiectului selectat cat si a celorlalte obiecte care contin acest produs) .

Pentru Composite Product , in noul tabel aparut se va gasi lista de produse din care este alcatuit acest produs , iar acesta se poate modifica prin adaugarea unui alt produs la compozitia acestuia ( click pe un produs din primul tabel dupa care se v apasa pe butonul de “Add” ) sau stergerea unui produs ( click pe produsul din tabelul 3 care se doreste sa fie sters din composite dupa care se va apasa pe butonul “Delete” ) .

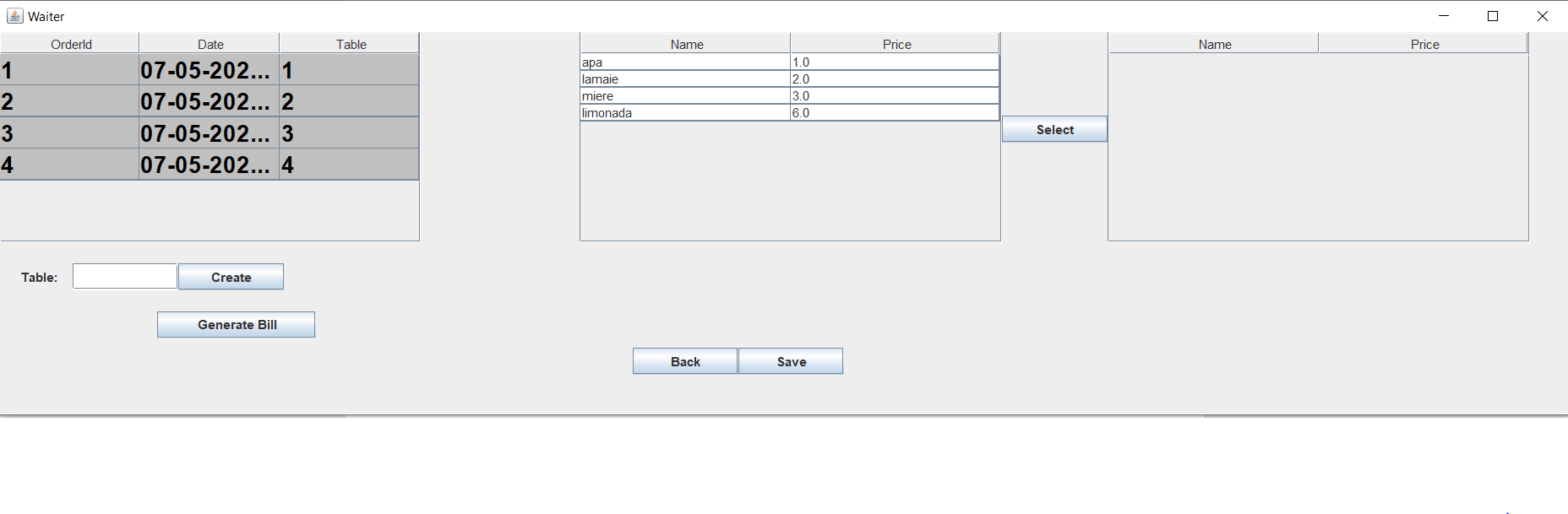


Daca se doreste stergerea unui produs din meniul restaurantului se va selecta ( click ) pe item-ul care se doreste sa fie sters din primul tabel dupa care se va apasa pe butonul “Delete” , daca acesta este un BaseProduct atunci se vor sterge si produsele care contin acest produs in lista lor de compozitie .

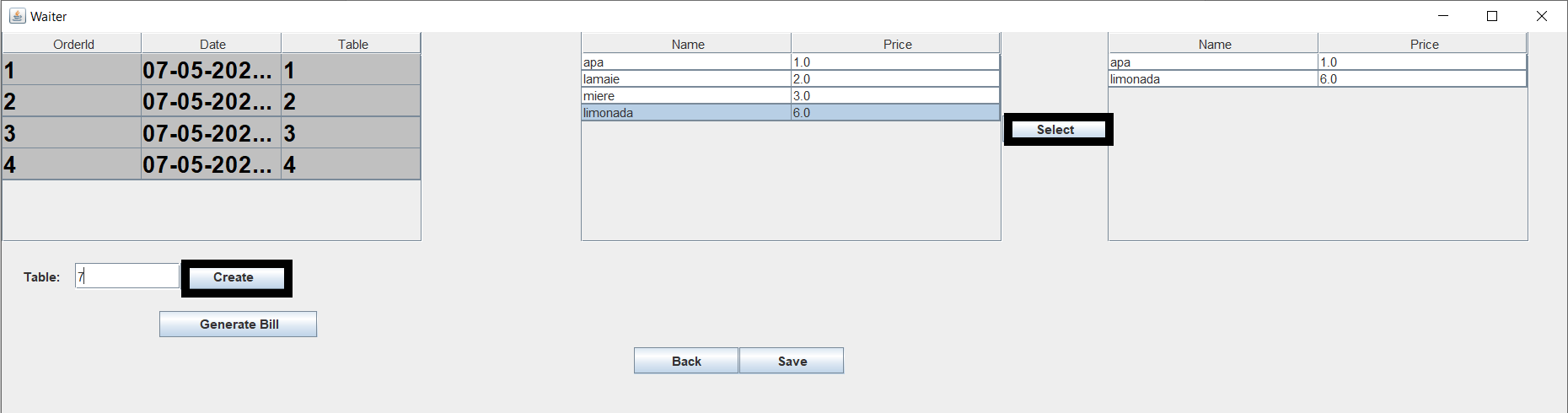


Pentru salvarea datelor noi introduse in aplicatie se va apasa pe butonul “Save” , daca se doreste revenirea la meniul principal se va apasa pe butonul “Back” .

Daca se va alege utilizatorul Waiter : in noul meniu deschis in primul tabel se vor gasi comenzile existemte , iar in cel de al doilea tabel se gaseste meniul restaurantului .



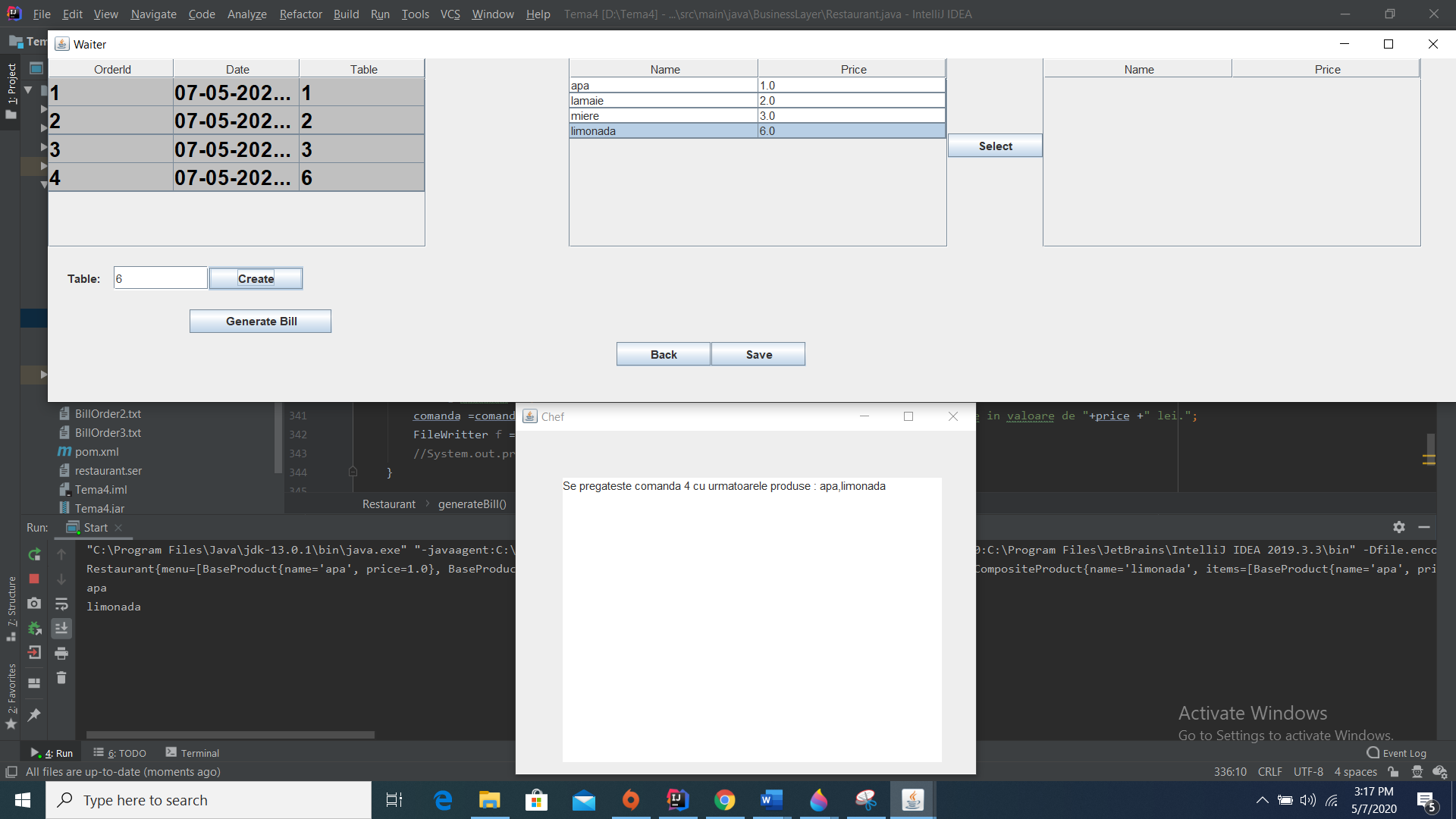
Daca se doreste crearea unui noi comenzi , atunci din al doilea tabel se vor selecta pe rand produsele care au fost comandate si se va apasa butonul “Select” ( click pe un item dupa care apasati pe select si tot asa pana ati introdus toate produsele dorite ) , produsele care au fost selectate vor aparea in cel de al treilea tabel , dupa care se va introduce in caseta Table numarul mesei la care a fost facuta comanda si se va apasa pe butonul create pentru a genera o noua comanda cu datele introduse anterior .



Daca se doreste generarea notei de plata se va selecta din primul tabel comanda la care se doreste sa se genereze si se va apasa dupa acea pe butonul “Generate Bill” , iar informatiile despre nota de plata se vor afisa intr-un fisier .txt

Pentru salvarea datelor noi introduse in aplicatie se va apasa pe butonul “Save” , daca se doreste revenirea la meniul principal se va apasa pe butonul “Back” .

Pentru a vedea produsele pregatite de Chef trebuie sa avem atat fereastra de Chef deschis cat si cea a Waiter-ului ( prima data se apasa pe butonul Chef , dupa care se apasa pe butonul “Waiter” ) . Dupa ce o noua comanda este plasata aceasta va aparea in interfata Chef ca este pregatita .In exemplul urmator am creat o noua comanda cu produsele : apa , limonada , iar dupa ce aceasta a fost creata comanda seful a fost notificat cu produsele pe care trebuie sa le pregateasca .

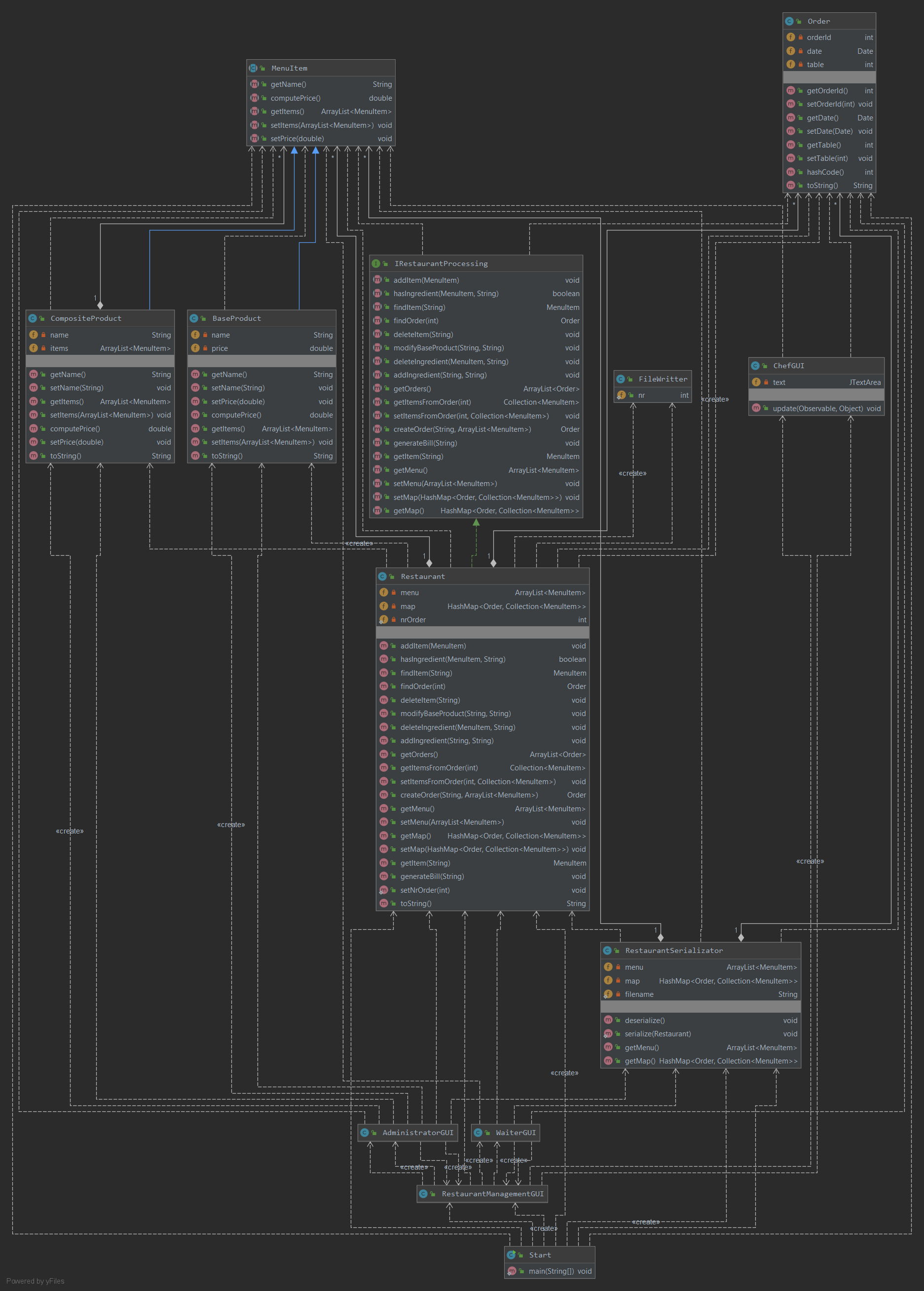


**3.** **Proiectare (decizii de proiectare, diagrame UML, structure de date, proiectare clase, interfete, relatii , packages, algoritmi, interfata utilizator)**

**3.1 Decizii de proiectare**

Pentru implementarea aplicatie am ales sa folosesc o interfata grafica care interactioneaza foarte mult cum mouse-ul pentru o utilizarea cat mai usoara a acesteia si de asemenea pentru o vizualizare cat mai buna a tuturor detalilor pe care aceasta aplicatie le foloseste .

**3.2 Diagrama UML**



**3.3 Packages**

* BusinessLayer -> Pachetul contine clasele destinate implementarii logicii aplicatiei . Acest pachet contine clasele BaseProduct , IRestaurantProcessing , CompositeProduct , MenuItem , Order , Restaurant .
* DataLayer -> Pachetul contine clasele destinate pentru citire , respective scriere a datelor . Acest pachet contine clasele FileWriter , RestaurantSerializator .
* PresentationLayer -> Pachetul contine clasele destinate pentru interfata grafica . Acest pachet contine clasele AdministratorGUI , ChefGUI , WaiterGUI , RestaurantManagementGUI .
* StarterLayer -> Acest pachet contine clasa Main a aplicatiei .

**4.Implementare**

**4.1 Clasa BaseProduct**

Clasa BaseProduct este sublcasa pentru MenuItem si este utilizata pentru formarea unu produs simplu . Aceasta clasa contine atributele name ( de tipul string) si price (de tipul double) .

Metode:

* public BaseProduct(String name, double price)

- constructorul clasei

* public String getName()

- getter pentru nume

* public void setName(String name)

- setter pentru nume

* public void setPrice

- setter pentru price

* public double computePrice()

- getter pentru price

**4.2 Clasa CompositeProduct**

Clasa CompositeProduct este subclasa pentru MenuItem si este utilizata pentru formarea unui produs compus . Aceasta clasa contine ca attribute name ( de tipul string ) si items ( de tipul ArrayList<MenuItem> ) care reprezinta lista de iteme din care este compus produsul .

Metode:

* public CompositeProduct(String name, ArrayList<MenuItem> items)

- constructorul clase

* public String getName()

- getter pentru name

* public void setName(String name)

- setter pentru name

* public ArrayList<MenuItem> getItems()

- getter pentru items

* public void setItems(ArrayList<MenuItem> items)

- setter pentru items

* public double computePrice()

- metoda care calculeaza pretul produsului

**4.3 Clasa IRestaurantProcessing**

Clasa IRestaurantProcessing reprezinta interfata pentru Clasa Restauranta , adica definitia metodelor implementate in aceasta si prezentate mai jos .

**4.4 Clasa Restaurant**

Clasa Restaurant contine informatile despre meniu si comenzile restaurantului . Aceasta clasa are ca atribute : menu de tipul ArrayList<MenuItem> care contine lista de lista de produse din meniul restaurantului , map de tipul HashMap<Order, Collection<MenuItem>> care contine comenzile si lista de produse associate fiecarei comenzi , nrOrder de tipul int care reprezinta numarul de comenzi efectuate.

Metode:

* public Restaurant(ArrayList<MenuItem> menu, HashMap<Order, Collection<MenuItem>> map)

- constructorul clasei

* public void addItem(MenuItem item)

- metoda pentru adaugarea unui item in meniul restaurantului

* public boolean hasIngredient(MenuItem menuItem, String name)

-Metoda care verica daca exista un produs are ca componenta un produs cu numele name

* public MenuItem findItem(String item)

-Metoda care gaseste un item in meniul restaurantului dupa nume

* public Order findOrder(int hashCode)

-Metoda care cauta un order in lista map dupa cheia hashcode.

* public void deleteItem(String name)

-Metoda care sterge un item din meniul restaurantului.

* public void modifyBaseProduct(String a,String c)

-Metoda care modifica pretul unui item.

* public void deleteIngredient(MenuItem m,String a)

-Metoda care sterge un produs din compozitia unui alt produs.

* public void addIngredient(String a,String b)

-Metoda care adauga un produs in lista componentelor a altui produs.

* public ArrayList<Order> getOrders()

-Un getter pentru lista de order-uri din lista map.

* public Collection<MenuItem> getItemsFromOrder(int hashCode)

-Metoda care genereaza lista de produse comandate pentru un order.

* public void setItemsFromOrder(int hashCode,Collection<MenuItem> items)

-Metoda care seteaza produsele comandate pentru un order.

* public Order createOrder(String table,ArrayList<MenuItem> items)

-Metoda care genereaza un nou order pentru o masa.

* public ArrayList<MenuItem> getMenu()

-Getter pentru meniul restaurantului.

* public void setMenu(ArrayList<MenuItem> menu)

-Un setter pentru meniul restaurantului.

* public HashMap<Order, Collection<MenuItem>> getMap()

-Un getter pentru lista map .

* public void setMap(HashMap<Order, Collection<MenuItem>> map)

-Un setter pentru lista map.

* public MenuItem getItem(String s)

-Metoda care returneaza un item din meniu dupa numele cautat.

* public void generateBill(String id)

-Metoda pentru generarea notei de plata.

* public static void setNrOrder(int nrOrder)

-Setter pentru numarl comenzii.

**4.5 Clasa MenuItem**

Clasa MenuItem este o clasa abstracta care este superclasa pentru clasele BaseProduct si CompositeProduct .

Metode:

* public MenuItem()

- constructorul clasei

* public abstract String getName()

- metoda care va trebui implementata de catre subclase

* public abstract double computePrice()

- metoda care va trebui implementata de catre subclase

* public abstract ArrayList<MenuItem> getItems()

- metoda care va trebui implementata de catre subclase

* public abstract void setItems(ArrayList<MenuItem> items)

- metoda care va trebui implementata de catre subclase

* public abstract void setPrice(double p)

- metoda care va trebui implementata de catre subclase

**4.6 Clasa Order**

Clasa Order este utilizata pentru crearea stocarea informatilor unei noi comenzi . Aceasta clasa are ca atribute : orderId care reprezinta id-ul comenzii ( de tipul int ) , date care reprezinta data la care a fost facuta comanda ( de tipul Date ) , table care reprezinta numarul mesei la care a fost facuta comanda ( de tipul int ) .

Metode:

* public Order(int orderId, Date date, int table)

- constructorul clasei

* public int getOrderId()

- getter pentru orderId

* public Date getDate()

- getter pentru date

* public void setDate(Date date)

- setter pentru date

* public int getTable()

- getter pentru table

* public void setTable(int table)

- setter pentru table

* public int hashCode()

- metoda pentru obtinerea hashCode-ului

**4.7 Clasa FileWriter**

Clasa FileWriter este implementata pentru scrierea in fisier ( .txt ) a datelor referitoare la o comanda . Este utilizata pentru generarea unei note de plata de catre Waiter .

**4.8 Clasa RestaurantSerializator**

Clasa RestaurantSerializator este implementata pentru scrierea si citirea datelor corespunzatoare pentru aplicatie . La pornirea aplicatiei datele vor fi citite din fisier , iar la apasarea butonului “Save” din interfata grafica datele vor fi salvate in fisier . Atributele acestei clase sunt : menu de tipul ArrayList<MenuItem> care reprezinta meniul restaurantului , map de tipul HashMap<Order, Collection<MenuItem>> care reprezinta inforamtiile despre o comanda si produsele care au fost comandate in aceasta , filename de tipul String care contine denumirea fisierului din care se citesc si se salveaza datele .

Metode:

* public RestaurantSerializator(String filename)

- constructorul clasei

* public void deserialize()

- metoda care citeste datele din fiser

* public static void serialize(Restaurant r)

- metoda care scrie datele in fisier

* public ArrayList<MenuItem> getMenu()

- getter pentru menu

* public HashMap<Order, Collection<MenuItem>> getMap()

- getter pentru map

**4.9 Clasa AdministratorGUI**

Clasa AdministratorGUI reprezinta interfata grafica pentru utilizatorul Administrator si a fost prezentata la inceput impreuna cu modul in care se utilizeaza.

**4.10 Clasa ChefGUI**

Clasa ChefGUI reprezinta interfata grafica pentru utilizatorul Chef si a fost prezentata la inceput impreuna cu modul in care se utilizeaza.

**4.11 Clasa WaiterGUI**

Clasa WaiterGUI reprezinta interfata grafica pentru utilizatorul Waiter si a fost prezentata la inceput impreuna cu modul in care se utilizeaza.

**4.12 Clasa RestaurantManagementGUI**

Clasa RestaurantManagementGUI reprezinta interfata grafica pentru maneiul principal al aplicatiei si a fost prezentata la inceput impreuna cu modul in care se utilizeaza.

**4.13 Clasa Start**

Clasa Start reprezinta clasa Main a aplicatiei care va fi rulata pentru a porni aplicatia .

**5.Concluzii**

Au fost implementate operatiile pentru fiecare utilizator : Administrator ( adaugare , modificare , stergere produs din meniul de produse al restaurantului ) , Waiter ( crearea unei comenzi , adaugarea de produse pentru o comanda , generea notei de plata ) , Chef ( afisarea produselor in curs de preparare ) impreuna cu o interfata grafica accesibila pentru o utilizare cat mai usoara a acestei aplicatii , dar si cu serializare pentru salvarea si citirea datelor corepsunzatoare aplicatiei . Din aceasta tema am invata sa lucrez cu o interfata grafica care interactioneza cu click-uri pe tabele , dar si sa folosesc serializarea pentru citirea si salvarea datelor intr-un fisier . O implementare ulterioara care se poate aduce este folosirea preconditilor , postconditilor .

**6.Bibliografie**

* <https://www.baeldung.com/java-serialization>
* <http://javarevisited.blogspot.ro/2011/02/how-hashmap-works-in-java.html>
* <https://netbeans.org/kb/docs/java/gui-functionality.html>
* <https://www.w3schools.com/java/java_files_create.asp>