Class Design and Diagram

After determining our users and the functions they would need to perform in the function list, we designed some classes that will be implemented for the application. All our classes, described below, they have all have different constructors, and getters and setters that complement the specialized methods.

User Classes

The superclass ***User*** defines the common attributes for the cooks, admin, and customers. For example, these three subclasses have username, password, name, user Id in common, and each one inherits these attributes and extends them based on their specific functions. The variables allow the users to have and log in to accounts on the site. The subclasses extend to more methods and attributes as follows:

Admin: Apart from a no argument constructor, admin has one constructor which get an integer. The admin subclass collaborates with both the cook and customer classes to allow for deactivation and deletion functions for both customers and cooks, and changing the status of cooks.

Customer: A customer also extends the user superclass by adding more attribute such as the physical address, phone number, email address stored for delivery purposes. The customer class has different constructors as well. The default no argument constructor, and the other is used to instantiate the attribute.

Cook: Of all our users, the cook has the most attributes and methods because of the functions we would like them to perform. Some of the additional cook attributes include country detail, cook rating, address, availability, approval and active statuses. Additional methods include refreshing the cook's average rating, and update the availability of the cooks are also implemented.

Non-User Classes

We have two classes that show the transaction of each customer's order and cook. ***Order*** and ***OrderLine*** classes collaborate to store the information for each order and customer receipt respectively. The order line class defines each meal item ordered, the price, and quantity. The Order class has methods that allow us to insert rows of order in our database.

The ***Meal*** class allows for the storage of each cook's meals offered. This class' attributes include the name and origin of the meal, and also stores pictures from the Image class that highlight the dish. The purpose of the Image class is just for storing and updating meal pictures. This class has lots of constructors to initiate the values of the fields. Get and set method in addition of methods are also located in this class.

Both the ***CookRating*** and ***MealRating*** classes have been created to complete the same set of functions, but for different objects. The cook rating class variables include a cook object and rating given by the customer. For example, the meal rating attributes include mealID and mealRating as integer and double in respect. Both classes have methods that collect the cook and meal rating and add the rate to the database.

Country is another class that is made into a class for holding information used by other main classes. We have designed this class to stay on the object oriented programming path. We store the names and flag images of the countries, so that we can display the details for our cooks and the meals they add. This class has three variables for flag, name and the continent, two construction and few specialized methods.

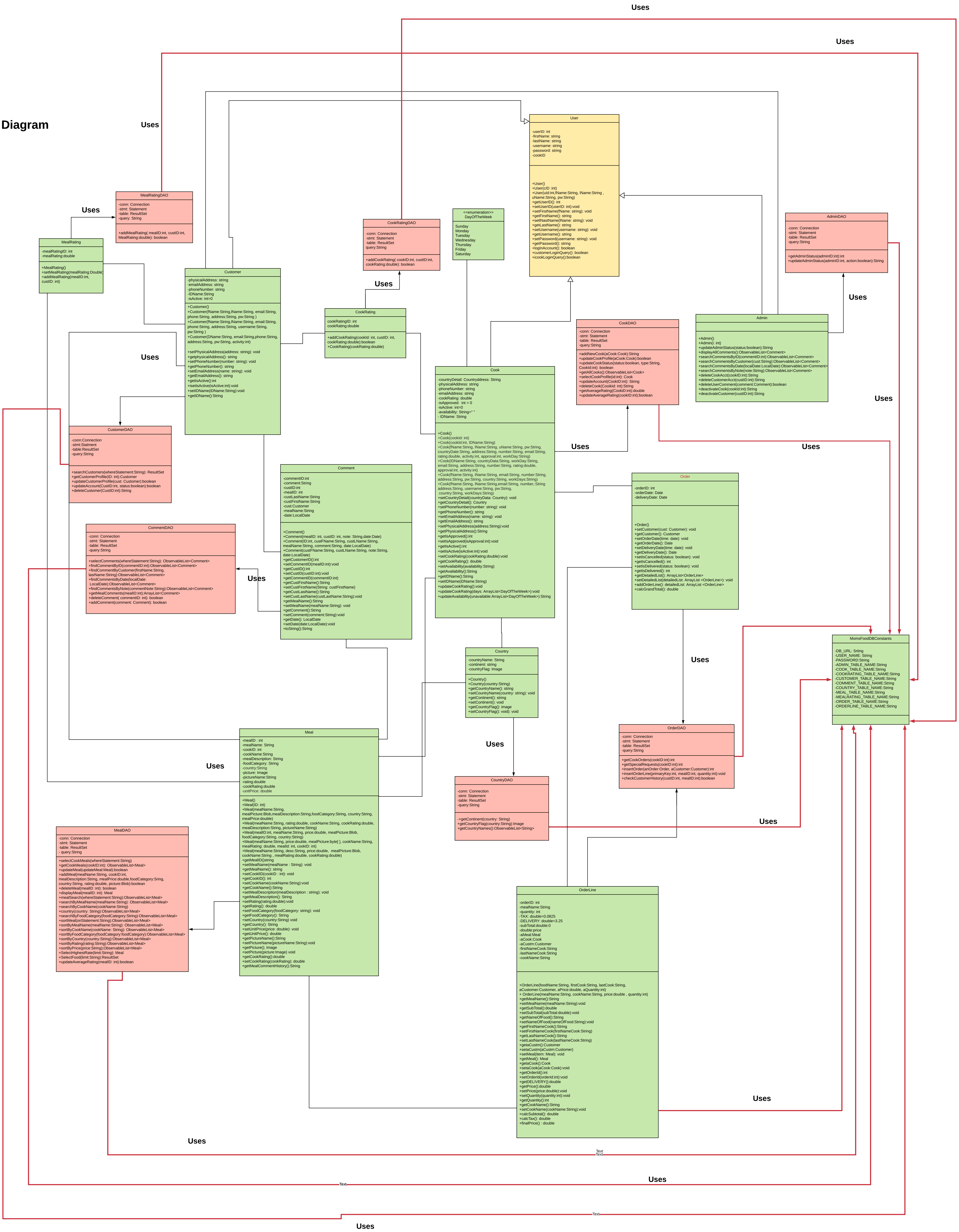
DayOfTheWeek one that helps us capture the cook's availability. This class has the enumeration stereotype.

CustomerDAO, AdminDAO, MealRatingDAO, cookRatingDAO, mealDAO, orderDAO, orderDAO, CountryDAO, commentDAO

These classes have methods that are related to the database. Usually all of them have four variables such as Connection, Statement, ResultSet and String for a query as attribute and some methods for connecting and retrieving data from database. These classes help us to manage our classes more efficiently.

MomsFoodDBConstants: This class helps us to store a range of constant attribute which are related to the name of the tables, URL, username and, password of the database.

Group 4: Class Diagram



MealRatingDAO

-conn: Connection
-stmt: Statement
-table: ResultSet
-query: String

+addMealRating(mealID:int, custID:int, MealRating:double): boolean

MealRating

-mealRatingID: int
-mealRating:double

+MealRating()
+setMealRating(mealRating:Double)
+addMealRating(mealID:int, custID: int)

Customer

-physicalAddress: string
-emailAddress: string
-phoneNumber: string
-IDName:String
-isActive: int=0

+Customer()
+Customer(fName:String,lName:String, email:String, phone:String, address:String, pw:String)
+Customer(fName:String,lName:String, email:String, phone:String, address:String, username:String, pw:String)
+Customer(DName:String, email:String,phone:String, address:String, pw:String, activity:int)

+setPhysicalAddress(address: string): void
+getphysicalAddress(): string
+setPhoneNumber(number: string): void
+getPhoneNumber(): string
+setEmailAddress(name: string): void
+getEmailAddress(): string
+getIsActive():int
+isetIsActive(isActive:int):void
+setIDName(iDName:String):void
+getIDName():String

CookRatingDAO

-conn: Connection
-stmt: Statement
-table: ResultSet
query:String

+addCookRating(cookID:int, custID:int, cookRating:double): boolean

CookRating

cookRatingID: int
cookRating:double

+addCookRating(cookId: int, custID: int, cookRating:double):boolean
+CookRating(cookRating:double)

<<enumeration>>
DayOfTheWeek

Sunday
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday

User

-userID: int
-firstName: string
-lastName: string
-username: string
-password: string
-cookID

+User()
+User(cID :int)
+User(uId:int,fName:String, lName:String , uName:String, pw:String)
+getUserID(): int
+setUserID(userID: int):void
+setFirstName(fName: string): void
+getFirstName(): string
+setlNastName(lName: string): void
+getLastName(): string
+setUsername(username: string): void
+getUsername(): string
+setPassword(username: string): void
+getPassword(): string
+logInAccount(): boolean
+customerLoginQuery(): boolean
+icookLoginQuery():boolean

AdminDAO

-conn: Connection
-stmt: Statement
-table: ResultSet
-query:String

+getAdminStatus(adminID:int):int
+updateAdminStatus(adminID:int, action:boolean):String

Admin

+Admin()
+Admin(i: int)
+updateAdminStatus(status:boolean):String
+displayAllComments():ObservableList<Comment>
+searchCommentsByID(commentID:int):ObservableList<Comment>
+searchCommentsByCustomer(cust:String):ObservableList<Comment>
+searchCommentsByDate(localDate:LocalDate):ObservableList<Comment>
+searchCommentsByNote(note:String):ObservableList<Comment>
+deleteCookAcct(cookID:int):String
+deleteCustomerAcct(custID:int):String
+deleteUserComment(comment:Comment):boolean
+deactivateCook(cookId:int):String
+deactivateCustomer(custID:int):String

CookDAO

-conn: Connection
-stmt: Statement
-table: ResultSet
-query:String

+addNewCook(aCook:Cook):String
+updateCookProfile(aCook:Cook):boolean
+updateCookStatus(status:boolean, type:String, CookId:int) :boolean
+getAllCooks():ObservableList<Cook>
+selectCookProfile(id:int): Cook
+updateAccount(CookID:int): String
+deleteCook(CooikId: int):String
+getAverageRating(CookiD:int):double
+updateAverageRating(cookID:int):boolean

Cook

-countryDetail: Countryddress: String
-physicalAddress: string
-phoneNumber: string
-emailAddress: string
-cookRating: double
-isApproved: int = 0
-isActive: int=0
-availability: String=" "
- IDName: String

+Cook()
+Cook(cookId: int)
+Cook(cookId:int, IDName:String)
+Cook(fName:String, lName:String, uName:String, pw:String, countryDate:String, address:String, number:String, email:String, rating:double, activity:int, approval:int, workDay:String)
+Cook(IDName:String, countryData:String, workDay:String, email:String, address:String, number:String, rating:double, approval:int, activity:int)
+Cook(fName:String, lName:String, email:String, number:String, address:String, pw:String, country:String, workDays:String)
+Cook(fName:String, lName:String,email:String, number,:String address:String, username:String, pw:String, country:String, workDays:String)
+setCountryDetail(countryData: Country): void
+getCountryDetail(): Country
+setPhoneNumber(number: string): void
+getPhoneNumber(): string
+setEmailAddress(name: string): void
+getEmailAddress(): string
+setPhysicalAddress(address:String):void
+getPhysicalAddress():String
+getIsApproved():int
+setIsApproved(isApproval:int):void
+getIsActive():int
+setIsActive(isActive:int):void
+setCookRating(cookRating:double):void
+getCookRating(): double
+setAvailability(availability:String)
+getAvailability():String
+getIDName():String
+setIDName(iDName:String)
+updateCookRating():void
+updateCookRating(days: ArrayList<DayOfTheWeek>):void
+updateAvailability(unavailable:ArrayList<DayOfTheWeek>):String

CustomerDAO

-conn:Connection
-stmt:Statment
-table:ResultSet
-query:String

+searchCustomers(whereStatement:String): ResultSet
+getCustomerProfile(ID: int):Customer
+updateCustomerProfile(cust: Customer):boolean
+updateAccount(CustID:int, status:boolean):boolean
+deleteCustomer(CustID:int):String

CommentDAO

-conn: Connection
-stmt: Statement
-table: ResultSet
-query:String

+selectComments(whereStatement:String): ObservableList<Comment>
+findCommentByID(commentID:int):ObservableList<Comment>
+findCommentByCustomer(firstName:String, lastName:String):ObservableList<Comment>
+findCommentsByDate(localDate: LocalDate):ObservableList<Comment>
+findCommentsByNote(commentNote:String):ObservableList<Comment>
+getMealComments(mealID:int):ArrayList<Comment>
+deleteComment(commentID: int): boolean
+addComment(comment: Comment): boolean

Comment

-commentID:int
-comment:String
-custID:int
-mealID: int
-custLastName:String
-custFirstName:String
-cust:Customer
-mealName:String
-date:LocalDate

+Comment()
+Comment(mealID: int, custID: int, note: String,date:Date)
+Comment(ID:int, custFName:String, custLName:String, mealName:String, comment:String, date:LocalDate)
+Comment(custFName:String, custLName:String, note:String, date:LocalDate)
+getCustomerID():int
+setCommentID(mealID:int):void
+getCustID():int
+setCustID(custID:int):void
+getCommentID(commentID:int)
+getCustFirstName():String
+setCustFirstName(String: custFirstName)
+getCustLastName():String
+setCustLastName(custLastName:String):void
+getMealName():String
+setMealName(mealName:String): void
+getComment():String
+setComment(comment:String):void
+getDate(): LocalDate
+setDate(date:LocalDate):void
+toString():String

Order

-orderID: int
-orderDate: Date
-deliveryDate: Date

+Order()
+setCustomer(cust: Customer): void
+getCustomer(): Customer
+setOrderDate(time: date): void
+getOrderDate(): Date
+setDeliveryDate(time: date): void
+getDeliveryDate(): Date
+setIsCancelled(status: boolean): void
+getIsCancelled(): int
+setIsDelivered(status: boolean): void
+getIsDelivered(): int
+getDetailedList(): ArrayList<OrderLine>
+setDetailedList(detailedList: ArrayList <OrderLine>): void
+addOrderLine(): detailedList: ArrayList <OrderLine>
+calcGrandTotal(): double

Country

-countryName: String
-continent: string
-countryFlag: Image

+Country()
+Country(country:String)
+getCountryName(): string
+setCountryName(country: string): void
+getContinent(): string
+setContinent(): void
+getCountryFlag(): image
+setCountryFlag(): void): void

MomsFoodDBConstants

-DB_URL: Srting
-USER_NAME: String
-PASSWORD:String
-ADMIN_TABLE_NAME:String
-COOK_TABLE_NAME:String
-COOKRATING_TABLE_NAME:String
-CUSTOMER_TABLE_NAME:String
-COMMENT_TABLE_NAME:String
-COUNTRY_TABLE_NAME:String
-MEAL_TABLE_NAME:String
-MEALRATING_TABLE_NAME:String
-ORDER_TABLE_NAME:String
-ORDERLINE_TABLE_NAME:String

OrderDAO

-conn: Connection
-stmt: Statement
-table: ResultSet
-query:String

+getCookOrders(cookID:int):int
+getSpecialRequests(cookID:int):int
+insertOrder(anOrder:Order, aCustomer:Customer):int
+insertOrderLine(primaryKey:int, mealID:int, quantity:int):void
+checkCustomerHistory(custID:int, mealID:int):boolean

CountryDAO

-conn: Connection
-stmt: Statement
-table: ResultSet
-query:String

-+getContinent(country: String)
+getCountryFlag(country:String):Image
+getCountryNames():ObservableList<String>

Meal

-mealID : int
-mealName: String
-cookID: int
-cookName:String
-mealDescription: String
-foodCategory: String
-country:String
-picture: Image
-pictureName:String
-rating:double
-cookRating:double
-unitPrice: double

+Meal()
+Meal(ID: int)
+Meal(mealName:String, mealPicture:Blob,mealDescription:String,foodCategory:String, country:String, mealPrice:double)
+Meal(mealName:String, rating:double, cookName:String, cookRating:double, mealDescription:String, pictureName:String)
+Meal(mealID:int, mealName:String, price:double, mealPicture:Blob, foodCategory:String, country:String)
+Meal(mealName:String, price:double, mealPicture:byte[ ], cookName:String, mealRating: double, mealId: int, cookID: int)
+Meal(mealName:String, desc:String, price:double, mealPicture:Blob, cookName:String , mealRating:double, cookRating:double)
+getMealID()string
+setMealName(mealName : String): void
+getMealName(): string
+setCooklID(cookID : int): void
+getCookID(): int
+setCookName(cookName:String):void
+getCookName():String
+setMealDescription(mealDescription : string): void
+getMealDescription(): String
+setRating(rating:double):void
+getRating(): double
+setFoodCategory(foodCategory: string): void
+getFoodCategory(): String
+setCountry(country:String) void
+getCountry(): String
+setUnitPrice(price: double): void
+getUnitPrice(): double
+getPictureName():String
+setPictureName(pictureName:String):void
+getPicture(); Image
+setPicture(picture:Image) void
+getCookRating():double
+setCookRating(cookRating): double
+getMealCommentHistory():String

MealDAO

-conn: Connection
-stmt: Statement
-table: ResultSet
- query:String

+selectCookMeals(whereStatement:String)
+getCookMeals(cookID:int): ObservableList<Meal>
+updateMeal(updateMeal:Meal):boolean
+addMeal(mealName:String, cookID:int, mealDescription:String, mealPrice:double,foodCategory:Sring, country:String, rating:double, picture:Blob):boolean
+deleteMeal(mealID :int): boolean
+displayMeal(mealID: int): Meal
+mealSearch(whereStatement:String):ObservableList<Meal>
+searchByMealName(mealName:String): ObservableList<Meal>
+searchByCookName(cookName:String)
+country(country: String):ObservableList<Meal>
+searchByFoodCategory(foodCategory:String):ObservableList<Meal>
+sortMeal(onStatement:String):ObservableList<Meal>
+sortByMealName(mealName:String): ObservableList<Meal>
+sortByCookName(cookName: String): ObservableList<Meal>
+sortByFoodCategory(foodCategory:foodCategory):ObservableList<Meal>
+sortByCountry(country:String):ObservableList<Meal>
+sortByRating(rating:String):ObservableList<Meal>
+sortByPrice(price:String);ObservableList<Meal>
+SelectHighestRate(limit:String): Meal
+SelectFood(limit:String):ResultSet
+updateAverageRating(mealID: int):boolean

OrderLine

-orderID: int
-mealName:String
-quantity: int
-TAX: double=0.0825
-DELIVERY: double=3.25
-subTotal:double:0
-double:price
-aMeal:Meal
-aCook:Cook
-aCustm:Customer
-firstNameCook:String
-lastNameCook:String
-cookName:String

+OrderLine(foodName:String, firstCook:String, lastCook:String, aCustomer:Customer, aPrice:double, aQuantity:int)
+ OrderLine(mealName:String, cookName:String, price:double , quantity:int)
+getMealName():String
+setMealName(mealName:String):void
+getSubTotal():double
+setSubTotal(subTotal:double):void
+getNameOfFood():String
+setNameOfFood(nameOfFood:String):void
+getFirstNameCook():String
+setFirstNameCook(firstNameCook:String)
+getLastNameCook():String
+setLastNameCook(lastNameCook:String)
+getaCustm():Customer
+setaCustm(aCustm:Customer)
+setMeal(item: Meal): void
+getMeal(): Meal
+getaCook():Cook
+setaCook(aCook:Cook):void
+getOrderId():int
+setOrderId(orderId:int):void
+getDELIVERY():double
+getPrice():double
+setPrice(price:double):void
+setQuantity(quantity:int):void
+getQuantity():int
+getCookName():String
+setCookName(cookName:String):void
+calcSubtotal(): double
+calcTax(): double
+finalPrice() : double