POLITECNICO
MILANO 1863

Iqueue Project

UML Requirements

Software Engineering for Automation (2022-2023)

Giacomelli Gianluca, 10615105
Gottardini Andrea, 10617000
Veronese Niccolò Enrico, 10620278

Professors: Rossi Matteo Giovanni
Lestingi Livia

Summary

UML Requirements provides a detailed report of the UML class diagram of our Iqueue project. In this document a description of the main functionality of our app is present, together with an explanation of the classes, attributes, and methods of each component of it.

1. Classes

The first passage is a description of the classes present in the UML diagram, with some hints about attributes and methods, which will be described in detail in the following sections.

- ID

This class is instrumental to keep track of the various ids of the main actors of the system like the Customer, the Shop Owner, the Shop, the Product, the QR code and the Advertisement. Defining an ID convention helps to design the methods of the different classes because it allows to specify in a clear and univocal way the input of the operations and who they are addressed to.

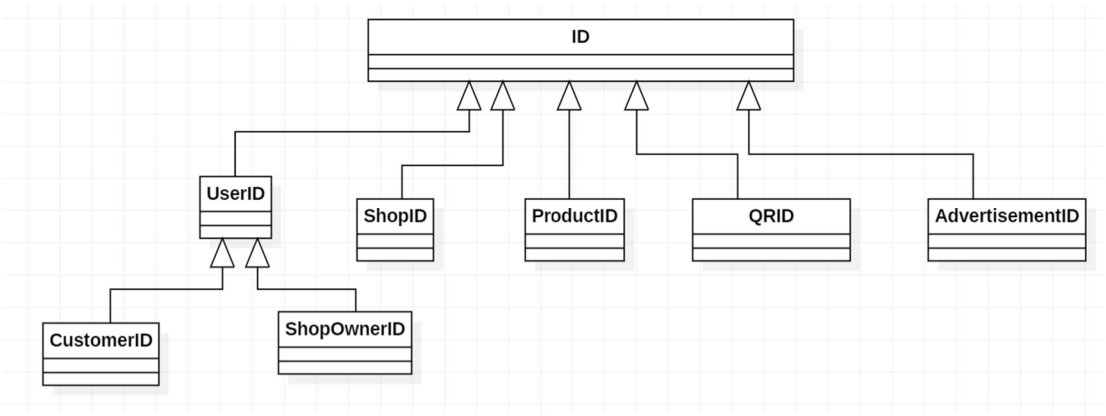

Figure 1: ID relationships

- Iqueue

Iqueue is the most important class of the diagram since it plays the role of the Iqueue application. It has no attributes, but many methods which represent the operations which have to be performed by the WebApp. It is connected via association with the User class, with the TimeSlot class, with the QR code class and with the Review class.

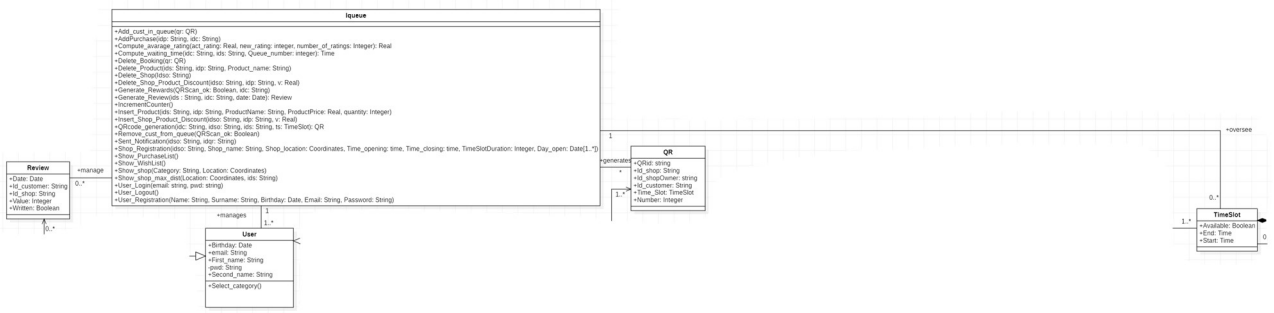

Figure 2: Iqueue class relationships

- User

The User class represents a potential purchaser of the Iqueue application and for this reason is directly connected with an association to the Iqueue class. In addition to that, it is characterized by a generalization relationship to the Customer and Shop Owner class, meaning that these latter will inherit the attributes of the User class and that they are part of it. Each User is characterized by attributes like first name, second name, birthday, password and an email address; all these features will be asked during the registration process in the Iqueue app. Apart from the attributes, the User class is also characterized by one method named Select_category().

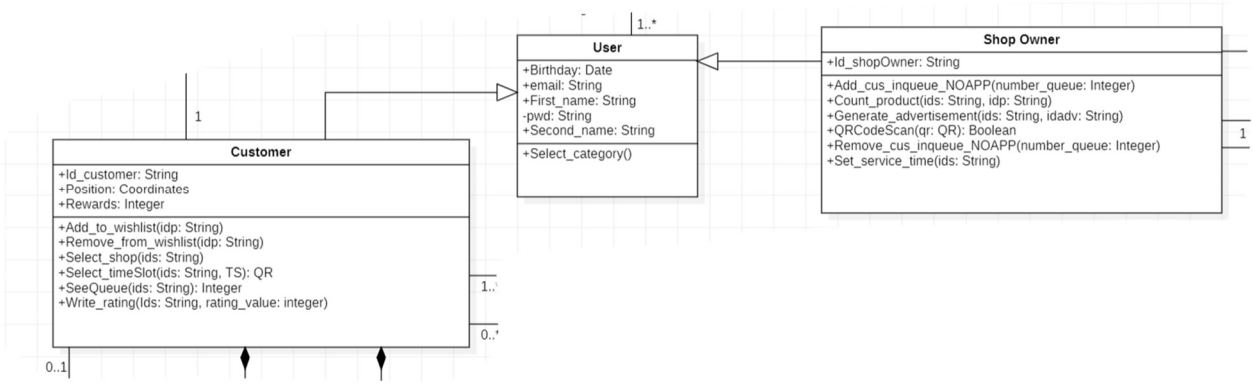

Figure 3: User Relationships

- Customer

The Customer class is the representation of a possible Customer for a shop, and it is directly connected to the User class via a generalization relationship, hence it will inherit the characteristics of the above class. This class is characterized by both attributes, like id_Customer, Position and Rewards (which keeps track of the number of rewards got by the client), and methods like Select_shop() and Select_timeSlot(). The Customer is also characterized by a WishList (list of the desired products) and a PurchaseList (list of purchased products with the date of purchase). These classes are represented as composition of the Customer class because they cannot be present without a Customer.

- Review

The class Review describes the review which will be generate by the Iqueue app just after the customer QR is scanned by the shop owner and thus it is connected to the Iqueue class and to the Customer, which has the role of write the review.

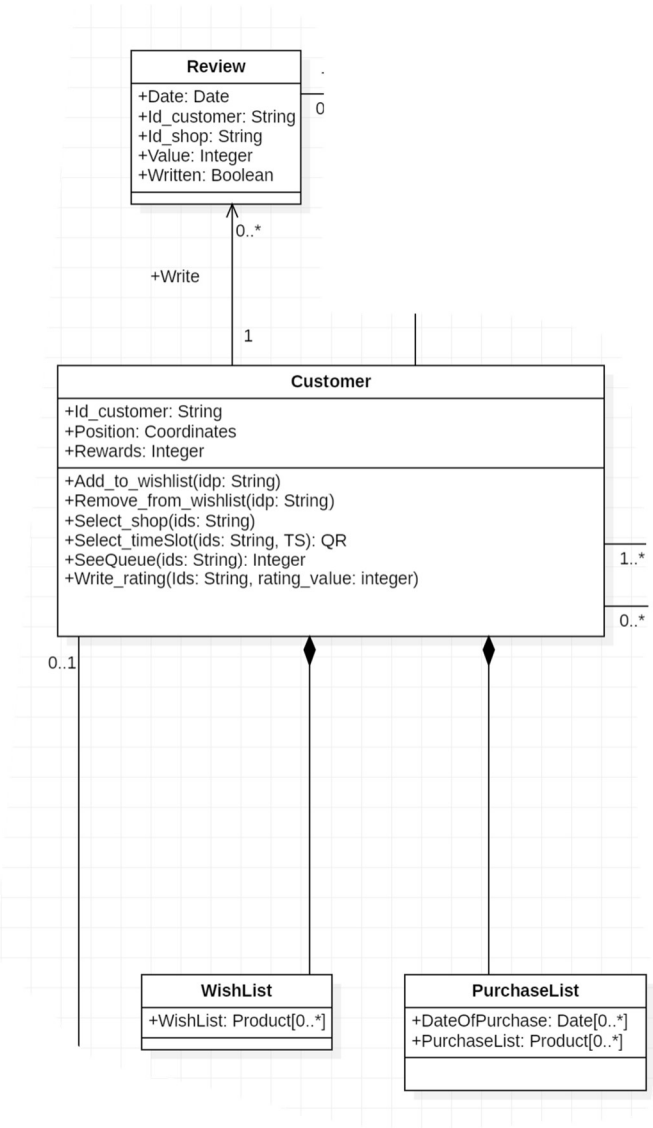

Figure 4: Customer class relationships

- Shop Owner

Similarly to the Customer class, the Shop Owner is connected to the User with a generalization and thus it will share the same features of this latter. The Shop Owner class represents the owner of a shop who is interested in downloading our application to register his shop and eventually his products and some special offers. In addition to that, he can also generate advertisements for his activity which can provide him a financial return. As for the Customer, he is characterized by its ID and some methods like QRcodeScan() and Add_cus_inqueue_NOAPP().

- Advertisement

A Shop Owner could also generate an Advertisement for his activity to have a financial gain from the use of the application. For this reason, the class is linked on one side to the Shop Owner and on the other side to the Shop. It has just attributes an no methods.

- QR

The class QR represents the QR code which will be generate by the Iqueue app as soon as a booking has been created and thus it is connected to the Iqueue class and to the Shop Owner class, which has the role of scan the QR code once the client with the booking enters in the Shop.

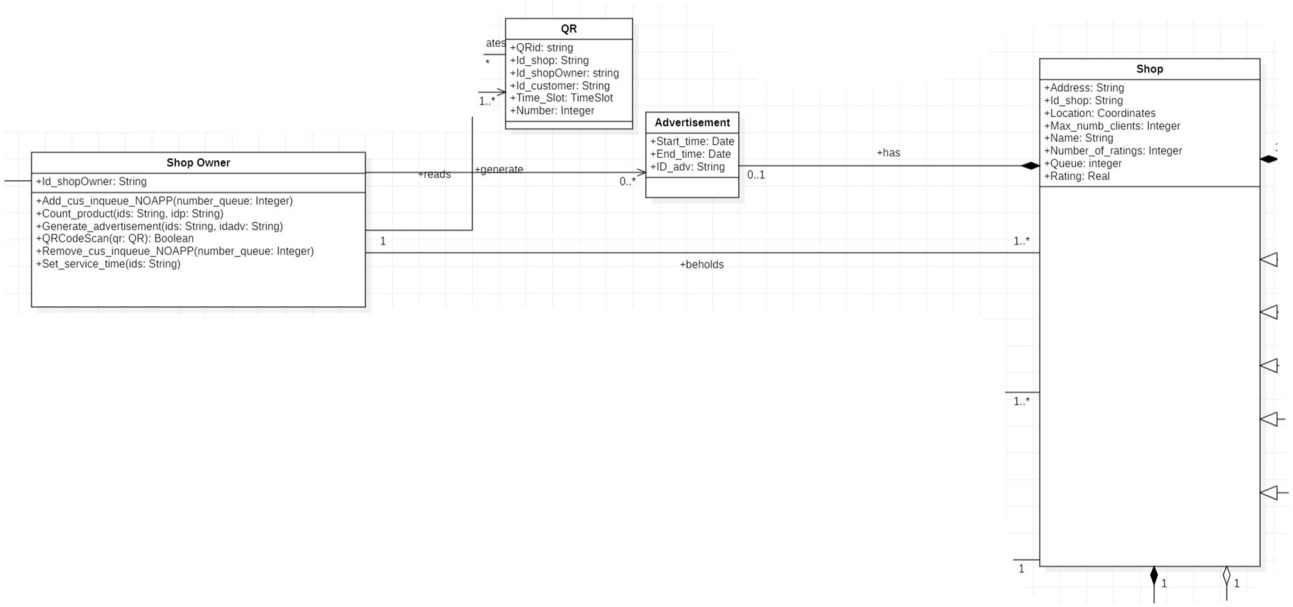

Figure 5: Shop Owner, Advertisement and QR class relationships

- Shop

The class Shop represents the role of a shop which has been inserted into the application by a Shop Owner, thus it is connected to it by a composition relationship because it cannot exist a Shop in our system without the corresponding Shop Owner. This class has attributes like Queue, which represents the number of people in queue at the shop, and of course the name, address and the location. Shop has no methods. An important feature of the Shop is that it has many subclasses like Bakery, Clothes, Perfumery, Hair saloon and Others which represent the possibly typologies of Shop which can be tackled by the application, since the target of Iqueue are small and medium activities. In addition to that, the class Shop is connected to the class Opening Counter whose role is to count the total number of customers which are present in the shop.

- Product

This class represents a possible Product inserted by a Shop Owner and part of a certain Shop, for this reason it is connected to the class Shop by a composition link, since if the Shop does not exist anymore the products cannot be sold as a consequence. The Product may be of two different typologies according to the Shop: Item or Service; this latter feature is represented by means of generalization link.

- Time slot

The Time Slot class represents the mean with which the Customer can make his reservation in the Shop. For this reason, the Time Slot is connected to the Shop (in fact each Shop can have a different Time Slot) and to the Customer which can reserve his preferable Time Slot. Of course, it is connected with a composition relation to the shop, since if the Shop does not exist anymore, the Time Slot has no longer meaning.

- Slot

The Slot class represents the available places in a Time Slot. Each Time Slot is characterized by a number of Slots equal to the number of maximum clients in the shop. A Slot is assigned internally by

Iqueue to each client that makes a reservation, and the number of the slot indicates the position inside the queue in the Time Slot.

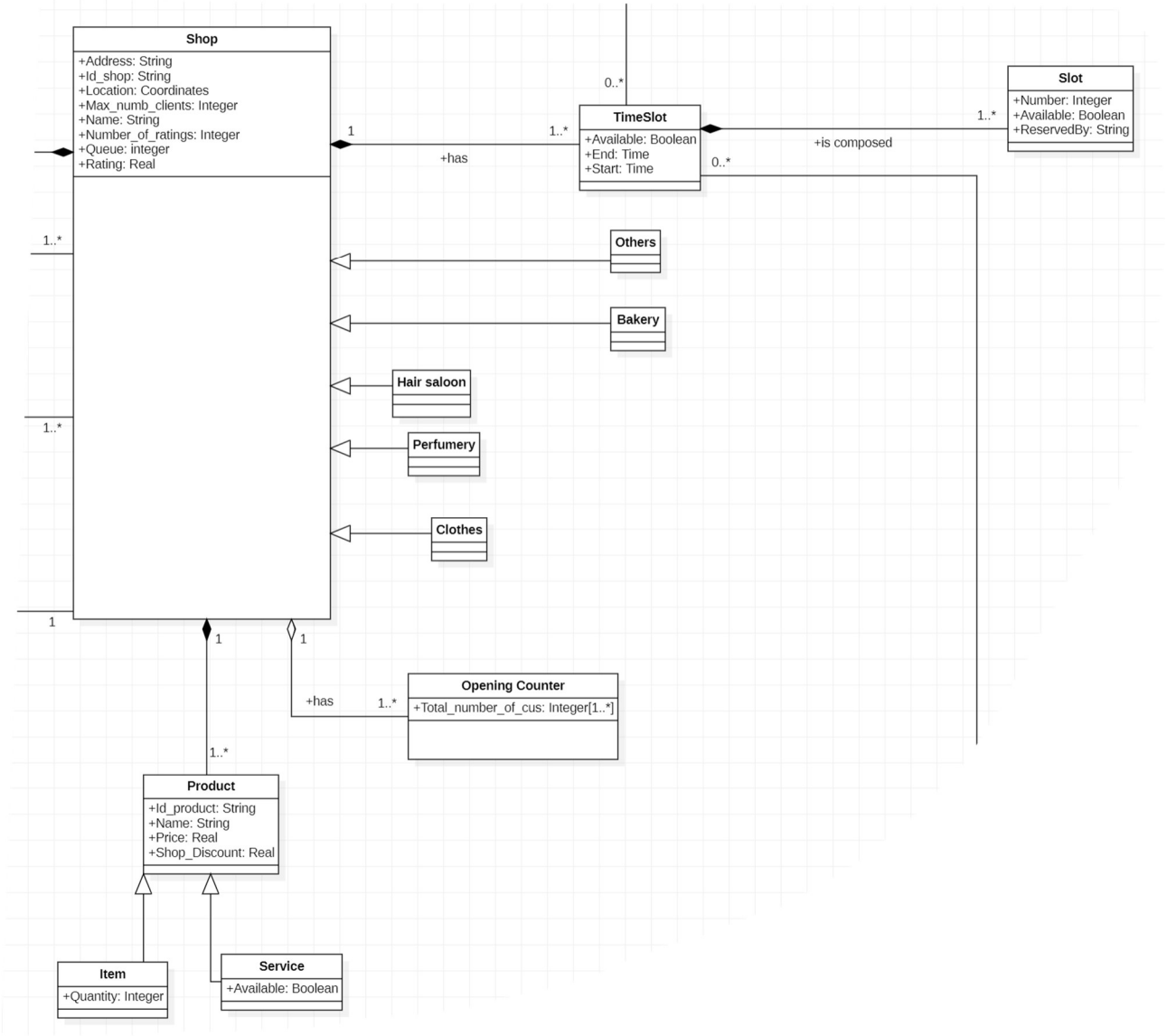

Figure 6: Shop, Product, Time slot and Slot classes relationships

- GPS system

This class represents a third party GPS system which will be included in the Iqueue framework. The GPS system has the role of showing the Shop position to the Customer and to guide the client from its location to the Shop. This class has a method named Guide_to_shop() which describes the above considerations and for this reason it is connected both to the Customer and to the Shop

2. Methods

In this section a detailed description of the methods performed by the different actors is shown:

- **Iqueue methods:**

  - User_registration(in Name:String, in Surname:String, in Birthday:Date, in Email:String, in Password:String)

This method takes as input the information of the User like Name, Surname, Birthday, Email and Password and it registers the new account created by the user.

  - User_login(in email:string, in pwd:string)

This method generates the user login, given the email and the password.

  - User_logout()

It allows the user to logout from the application.

  - Shop_Registration(in idso:String, in Shop_name:String, in Shop_location:Coordinates, in Time_opening:time, in Time_closing:time, in TimeSlotDuration:Integer, in Day_open:Date[1..*])

Creates the form for the shop registration, given the ID of the Shop owner (idso), the shop name, location and opening information.

  - Delete_shop(in Idso:String)

It allows to the delete a shop from Iqueue.

  - QRcode_generation(in idc:String, in idso:String, in ids:String, in ts:TimeSlot): QR

Automatically generates a QR code once the time slot has been booked, in fact the inputs are the id of Customer (idc), Shop Owner (idso), Shop (ids) and the TimeSlot.

  - Add_cust_in_queue(in qr:QR)

This method automatically add a customer in the queue, once the QR code has been created.

  - Remove_cust_from_queue(in QRScan_ok:Boolean)

This method automatically remove a customer from the queue, once the QR code has been scanned.

  - Sent_notification(in idso:String, in idqr:String)

It sent a notification to the Shop Owner when a booking is generated.

  - Show_shop(in Category:String, in Location:Coordinates)

It allows to show the Shops that are present in the Iqueue system, given the interested category and location.

  - Show_shop_max_dist(in Location:Coordinates, in ids:String)

It allows to show the maximum distance from the Shop given the Customer location and the ID of the Shop(ids).

- Generate_rewards(in QRScan_ok:Boolean, in idc:String)

This operation generates a reward for the Customer that has used the Iqueue app and in fact the inputs are the check that the QR code has been scanned and the id of the customer.

- Compute_waiting_time(in idc:String, in ids:String, in Queue_number:integer): Time

Computes the waiting time in queue, given the customer and the shop id (idc and ids) and the number of people in queue.

- Delete_Booking(in qr:QR)

Allows the customer to delete a booking.

- Insert_Product(in ids:String, in idp:String, in ProductName:String, in ProductPrice:Real, in quantity:Integer)

Inserts the products of a shop and in fact the inputs are the id of the shop (ids), of the product (idp) and the product name, price and quantity.

- Delete_Product(in ids:String, in idp:String, in Product_name:String)

Allows to delete a product from the shop.

- Insert_Shop_product_discount(in idso:String, in idp:String, in v:Real)

Allows to insert product discounts given the id of the shop owner (idso) and of the product (idp) and the value v of the discout

- Delete_Shop_product_discount(in idso:String, in idp:String, in v:Real)

Remove the product discount.

- Compute_avarage_rating(in act_rating:Real, in new_rating:integer, in number_of_ratings:Integer): Real

This method automatically generates the average rating for the shop, by updating the actual rating value, the new value and the number of ratings.

- AddPurchase(in idp:String, in idc:String)

Add the product idp to the purchase list of the customer idc.

- Show_PurchaseList()

It allows the Customer to see its PurchaseList.

- Generate_Review(in ids :String, in idc:String, in date:Date): Review

Automatically generates a review once the shop owner has scanned the customer QR, in fact the inputs are the id of Customer (idc), Shop (ids), and the date of the scanning.

- **User methods**:

  - Select_category()

This method allows the User to select which category he wants to be, either Customer or Shop Owner.

- **Customer methods**

- Select_shop(in ids:String)

The Customer can choose the Shop he wants to visit.

- Select_timeSlot(in ids:String, in TS:TimeSlot): QR

The Customer should select the TimeSlot in which he wants to go to the Shop.

- SeeQueue(in ids:String): Integer

It allows the Customer to see the queue in front of the Shop, given the id of the Shop (ids).

- Add_to_wishlist(in idp:String)

Given a certain Product with its id (idp), the Customer can add it to its wishlist.

- Remove_from_wishlist(in idp:String)

Remove a Product from the wishlist.

- Write_rating(in Ids:String, in rating_value:integer)

It allows to provide a rating of the Shop (given id of the shop ids) and the value of rating the Customer wants to give to the Shop.

- **Shop Owner methods:**

  - QRCodeScan(in qr:QR): Boolean

With this method the Shop Owner can scan the QR code of the Customer in the Shop.

- Add_cus_inqueue_NOAPP(in number_queue:Integer)

It allows the Shop Owner to manually increment the queue counter in case a client without the booking with the Iqueue app enters in the Shop.

- Remove_cus_inqueue_NOAPP(in number_queue:Integer)

It allows the Shop Owner to manually decrement the queue counter in case a client without the booking with the Iqueue app leaves in the Shop.

- Set_service_time(in ids:String)

The Shop Owner can decide which is the service time for each Customer in the Shop.

- Generate_advertisement(in ids:String, in idadv:String)

The Shop Owner can generate an advertisement for his Shop, given the id of the Shop ids and of the Advertisement idadv.

- Count_product(in ids:String, in idp:String)

It allows the Shop Owner to count the product in a Shop.

- **GPS system methods:**

  - Guide_to_shop(in cust_pos:Coordinates, in shop_pos:Coordinates)

Operation provided by the GPS system which guides the Customer from its location to the selected Shop.

3. Sequence diagrams

In this section, in order to describe better the functionalities of the UML requirements, some sequence diagrams of relevant methods are shown. The diagrams below are referred not only to the operations described in the UML, but also to graphical features of the Iqueue application. Complementary to this part, in the RASD document other sequence diagrams linked to the use cases are presented. The sequence diagrams can be found in the file UML REQUIREMENT.mdj, accessing them inside the UML Class Diagram through the Model Explorer.

- AddProductToWishList

The following sequence diagram explains the steps to be followed to insert a product in the wishlist. First of all, from the HomePage of the Iqueue app the customer is able to load its wishlist (from the GUI). After that, with the Add_to_wishlist(idp) method he can add a product defined by its idp to the wishlist.

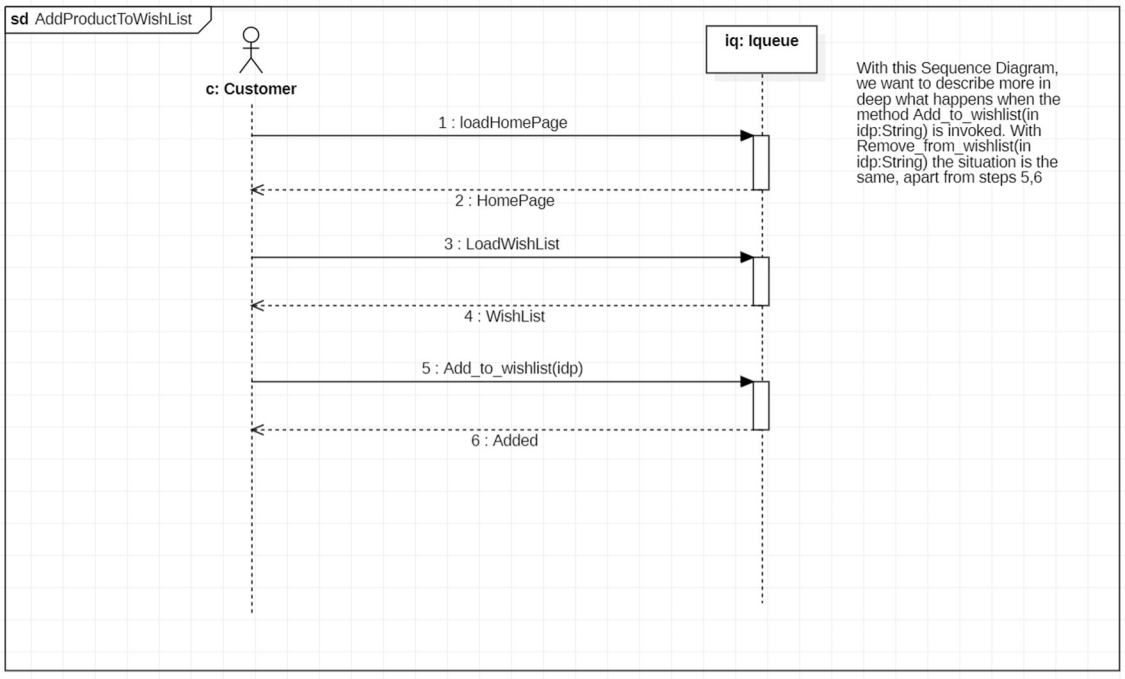

Figure 7: AddProductToWishList Sequence diagram

- Advertisement

The following scheme represents the passages for the generation of an advertisement for the shop. Once the Shop Owner load the advertisement design, he could use the operation Generate_advertisement(ids, idadv) to link the advertisement with the corresponding shop

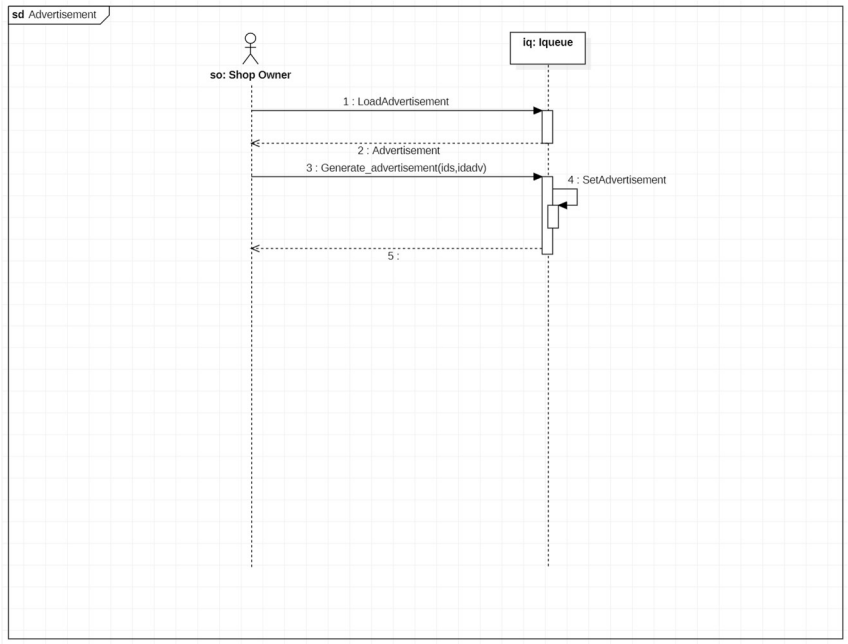

Figure 8: Advertisement sequence diagram

- Customer Delete Booking

This sequence diagram explains the passages that the Customer should follow to delete its booking. In the HompePage interface he can see its QR codes (MyQRList), each one corresponding to a booking. From that list, he can select the QR code to be deleted (selectQR_to_delete) and invoking the method Delete_Booking(QR) with the QR id as input he will delete its booking and hence he will exit from the queue in that Shop.

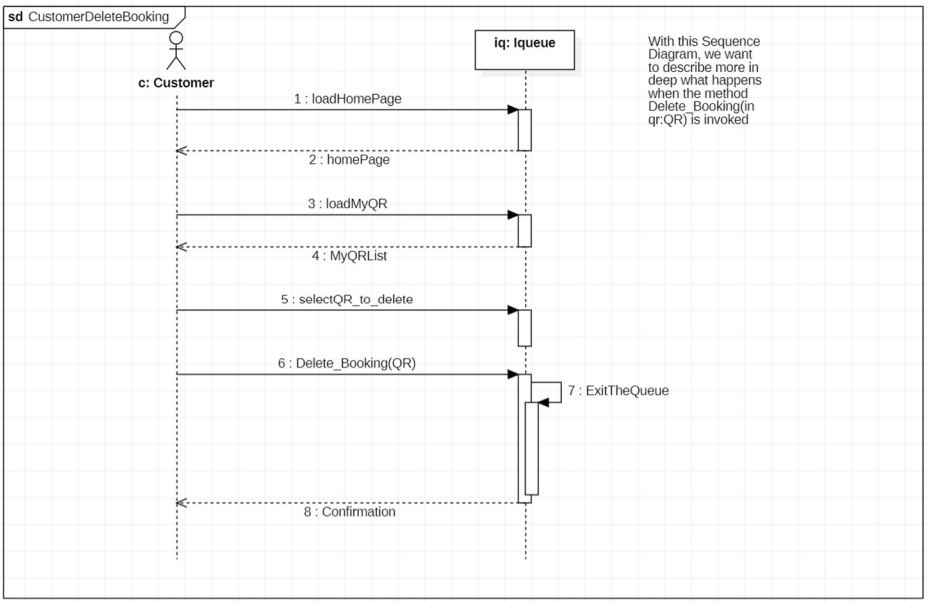

Figure 9: Customer Delete Booking sequence diagram

- Remove Product

This sequence diagram highlights the steps of the Delete_Product() method: the Shop Owner arrives in the graphical interface where he can manage the products of the shop, and by applying the Delete_Product(ids, idp) he is able to remove the product idp from the corresponding shop ids

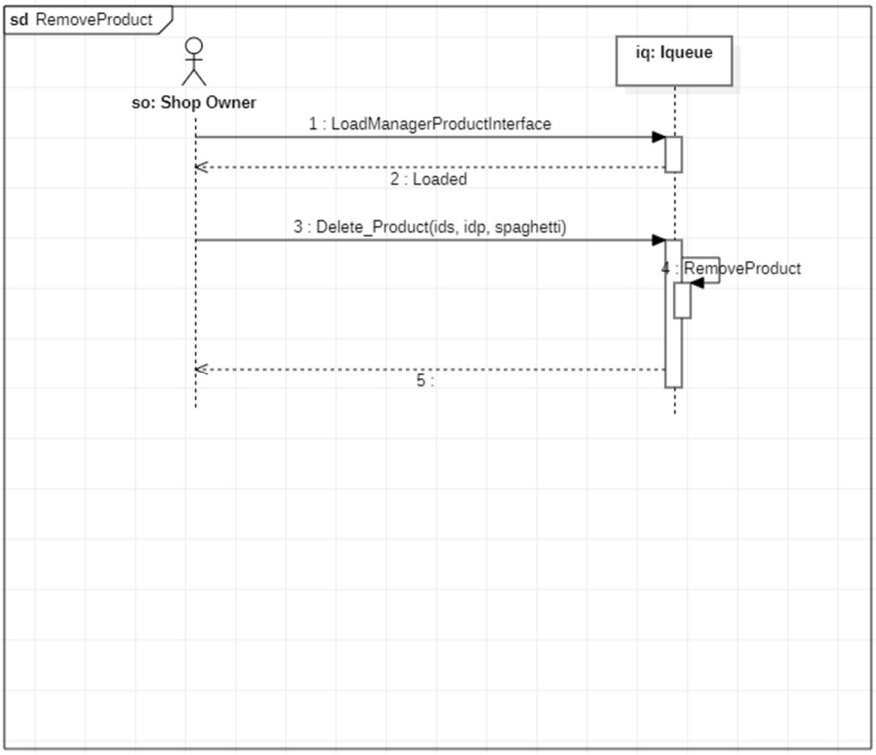

Figure 10: Remove product sequence diagram

- User Login

The following sequence diagram shows the steps of the User Login phase.

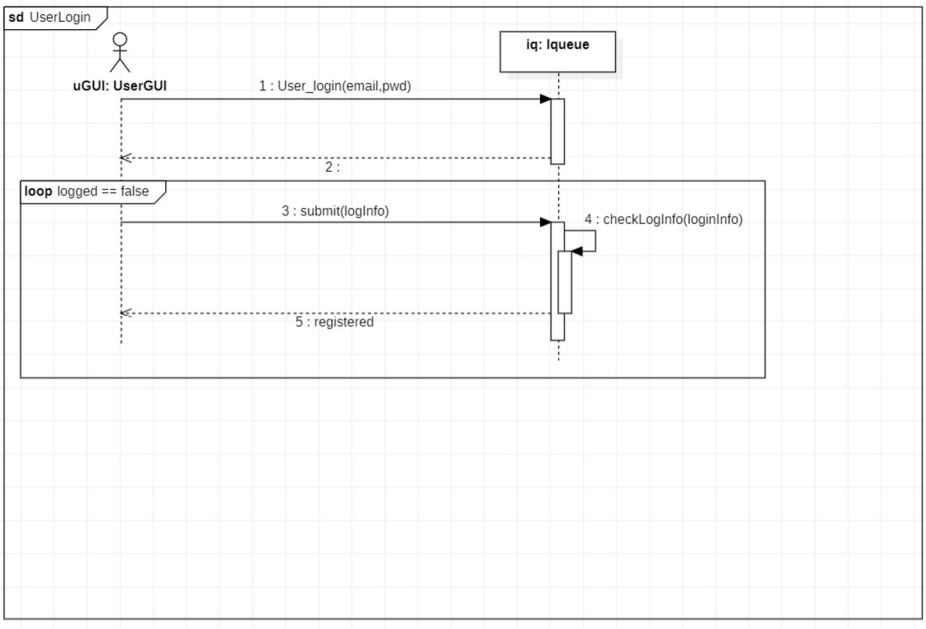

Figure 11: User login sequence diagram

- User Registration

Sequence diagram related to the User Registration

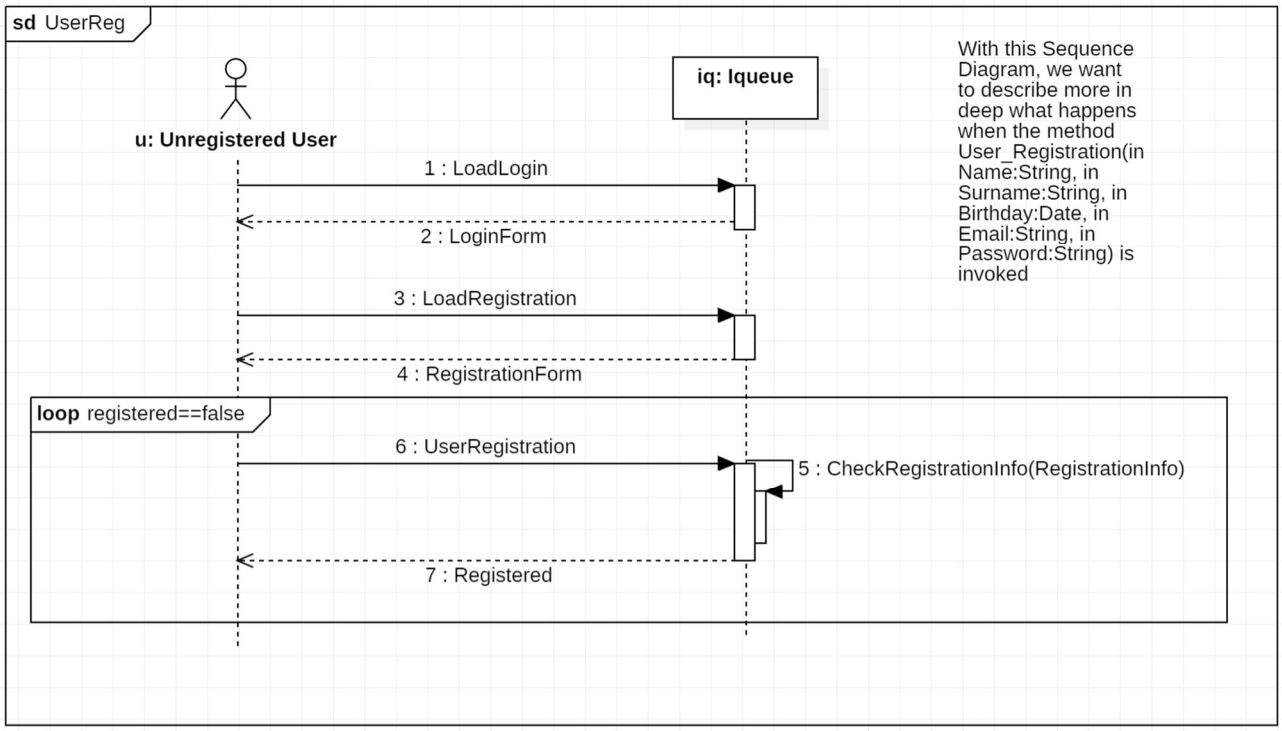

Figure 12: User registration sequence diagram