
POLITECNICO
MILANO

SOFTWARE ENGINEERING 2

TEACHER:
PROF. RAFFAELA MIRANDOLA

GuessBid

DESIGN DOCUMENT

Authors:

Eliseo Sartori
Alessandro Picca

Sommario

1. Introduction

This paper aims to describe in detail the architectural and functional characteristics of the system, with particular attention to the description of the choices related to the structure of the database and the interaction between users and GuessBid.
This document complies with the specifications described in the RASD and its draft is totally based on it, although the RASD has undergone some changes, as will be seen later in this document.

1.1 Sources and references

- IEEE Standard for Information Technology-Systems Design and Software Design
- Requirements analysis and specification document (Rasd), already available in this folder

1.2 Changes to RASD document

In RASD it was talked of the system compliance to some standards (including HTML5) without taking into consideration some aspects (and constraints) implementative emerged along the way; for this reason, any implementation choices described in this document and disagreeing to those mentioned in the RASD should be considered as a priority.

The section *"Settings"*, where is possible to modify personal profile is directly accessible from the page *"Logged User Profile*" ;
at the moment, this section contains only the possibility to modify your own personal information, while in the future we may add new features such as

customizing your profile page with colored backgrounds, but for now this is not possible.

1.3 Overview

The remainder of this paper is organized as follows:

- Section 2 – Architecture description: brief description of Java EE's architecture;
- Section 3 – Persistent data management: detailed explanation of the choices that affect the database's structure;
- Section 4 – User Experience: some words about the user experience (UX) given by GuessBid to its users;
- Section 5 – BCE diagrams: a further diagram of the system, built according to the paradigm BCE (boundary – control – entity);
- Section 6 – Sequence diagrams: these diagrams are designed to better understand the BCE Diagram, through examples of user interaction with the system, and its "path" between Boundary, Control and Entity classes;
- Section 7 - Appendix: general considerations about the document and the system.

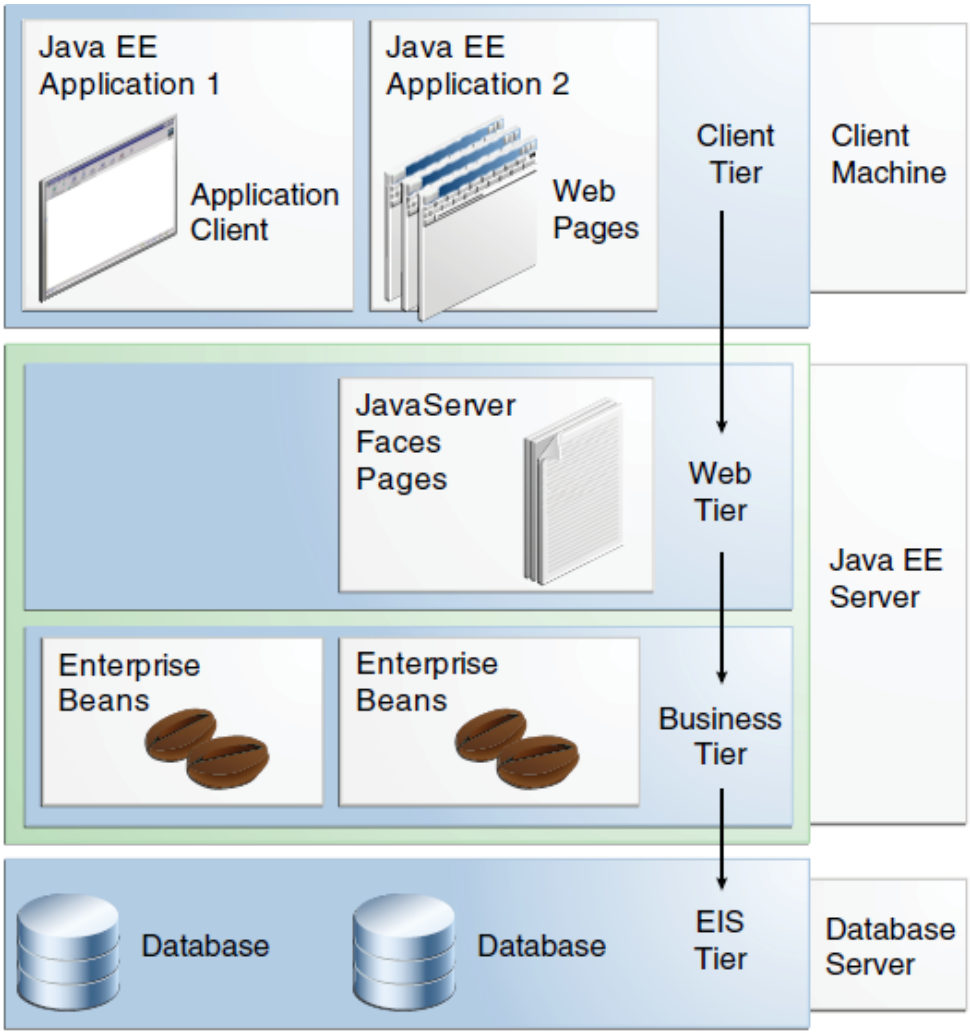
2. Architecture description

2.1 Java enterprise architecture overview

As already mentioned in various sections of the RASD, the architecture used is a client-server architecture: a client device (via browser) queries via HTTP the web server, which processes the request and provides dynamic web pages. For simplicity, database and application server will be considered on the same physical machine, which will later be generically defined as "server".

In this specific case the Java Enterprise overall architecture of the system is composed of four basic tier:

- *Client Tier:* it is the portion of the system that hosts the user interface, which allows the interaction between the user and the platform. The user only needs a browser and an Internet connection to access the system.
- *Web Tier:* portion located on the server and that has to permit the dialogue with the interface located on the client to generate the required contents each time according to user interactions.
- *Business Tier:* like Web Tier, also this portion is located on the server and deals with the interaction with the database via EJB (Enterprise Java Beans). The application server chosen to develop GuessBid is Glassfish version 4.1
- *EIS Tier:* it is the tier that contains all relevant data, stored in a database and managed by a MySQL server.



2.2 Identifying Sub-systems

We decided to adopt a top-down approach at least at this point of the project. Maybe, once defined the sub-systems, we will adopt a bottom-up approach in order to create more reusable components.

We also think that it is necessary to split our system into some sub-systems, in order to make our team-work easier with a clear division of jobs and to separate groups of functionality to define their interactions.

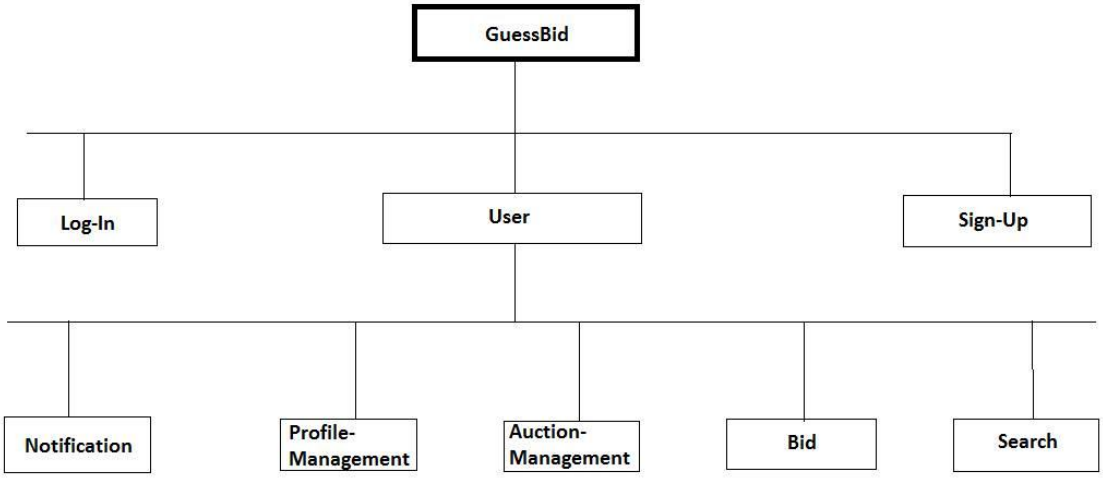
This is the result of what we have just explained:

Sub-systems identified:

- Sign up sub-system;
- Log in sub-system;
- User sub-system:
    - Notification sub-system;
    - Profile managing sub-system
    - Auction Management (Only Creation)
    - Bid sub-system;
    - Search sub-system;

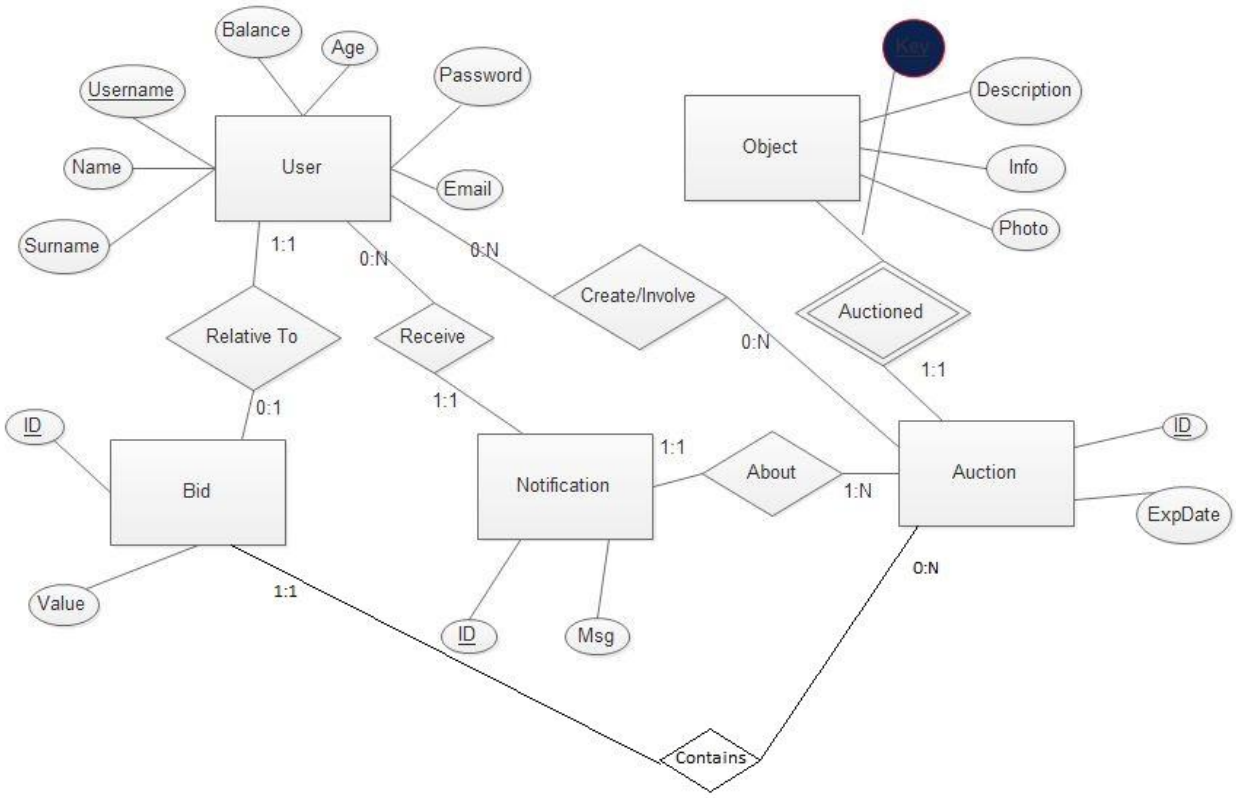
3. Persistent data management

We decided to upload data about the system in a database: below we show the conceptual model, in terms of entity-relationship diagram (ER) and logical model, obtained starting from the ER itself.



3.1 Conceptual model

the conceptual model allows you to have a first idea about the system, in terms of entities and relationships between these entities themselves



Notes to the previous diagram:
-There are some modification with respect to what we said in RASD; in general the valid version of a document is the most updated, which overwrite all the others
-Key attributes are underlined
-Weak relationship are marked with double border

In our system, there are two different types of person: the user that is not registered and logged user; since only the latter entity contains information relevant to the ER diagram we decided not to split the generic entity "person"

in two sub-entities that inherit from it, but to keep the only entity "user". Moreover, in the diagram, we have created the entity "object", defined as a weak entity referred to “auction” entity, to whom is linked with the relation "auctioned", in fact “auction” presents as primary key, in addition to its attributes "description", “info” and "photo ", also the primary key of the entity "auction".
Another entity is “bid”, characterized by the primary key attribute “BidID”, but also by the attribute “value”; “bid” is linked both with “user”, with the relation “relative to”, and with “Auction”, through the relation “contains”, in GuessBid’s ER diagram.
The system also contains the entity "notification", connected to the user only by the relation "receive", supposing that all notifications are arbitrarily sent by system, and that has among others, the "message" attribute, containing the message that the system delivers to the user with the specific notification to which it refers.

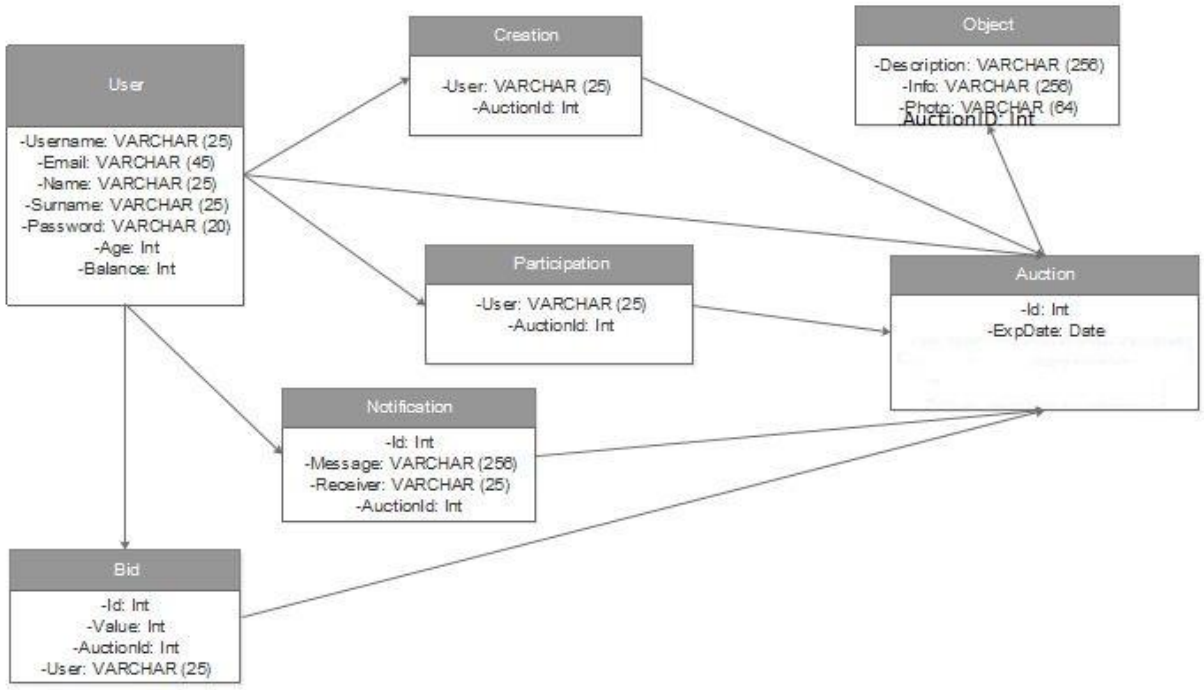
3.2. Translation to logical model

Starting from the ER we can derive the logical model of the database, using these tips:

- Relation “Receive” between “User” and “Notification” has been translated inserting a foreign key into the table “Notification” named “Receiver”;
- The weak entity "Object" , with respect to "Auction", has been translated adding the primary key of the latter entity into the “Object” entity;
- “Create” and “Involved” relations, being many to many relationships, have been translated into two new table called “Creation” and “Participation”, both containing primary keys of “User” and “Auction”

- The relation “Relative to” , between “User” and “Bid”, has been translated adding to the entity “Bid” a foreign key “User”, which corresponds to the username of the creator;
- The entity “Notification” will have one more foreign key, which is “AuctionId”, in order to resolve the relation “About”;
- The relation “Contains”, between entities “Auction” and “Bid” has been translated adding the primary key of “Auction” (its ID), as foreign keys of “Bid”.

After that we can draw the logical model:



3.3. Physical structure of the logical model

Our model has the following physical structure:

- **User** (Username, Email, Name, Surname, Password, Age, Balance)
- **Creation** (*Username, AuctionID*)
- **Object** (Description, Info, Photo, *AuctionID*)
- **Bid** (ID, *Username, AuctionID*, Value)
- **Auction** (ID, ExpDate)
- **Participation** (*Username, AuctionID*)
- **Notification** (ID, Message, *AuctionID, Receiver*)

Legend:

-Primary keys are underlined

-Foreign keys are in Italic (pay attention to the conceptual model and paragraph 3.2 to understand which entities are referred to)

4. User Experience

The aim of this paragraph is to describe the user experience (UE) that GuessBid gives to its users; in order to do that we drafted some Class Diagrams (only one Class Diagram would have been chaotic and more difficult to understand, so we decided to divide it in few smaller class diagrams) with appropriate stereotypes <<*screen*>>, <<*screen compartment*>> and <<*input form*>>.
There is a correspondence between stereotypes and real component of the system, in particular <<*screen*>> represents GuessBid pages, <<*screen compartment*>> represents parts of the page that can be shared with others person and <<*input form*>>, where there is, represents some input fields that can be fulfilled by a user and eventually the user input will be submitted to the system clicking on a button.
The following UX Diagrams, as we said previously, are part of a big generic UX Diagram, and the division was made on the grounds that every UX Diagram

contains similar functionality; furthermore, in each UX diagram are not shown all the methods relating to each screen, but only those relevant to the UX Diagram in question.
Each of the paragraphs below is titled with the name of the functionality (or the set of functionalities) represented by the UX Diagram drawn.

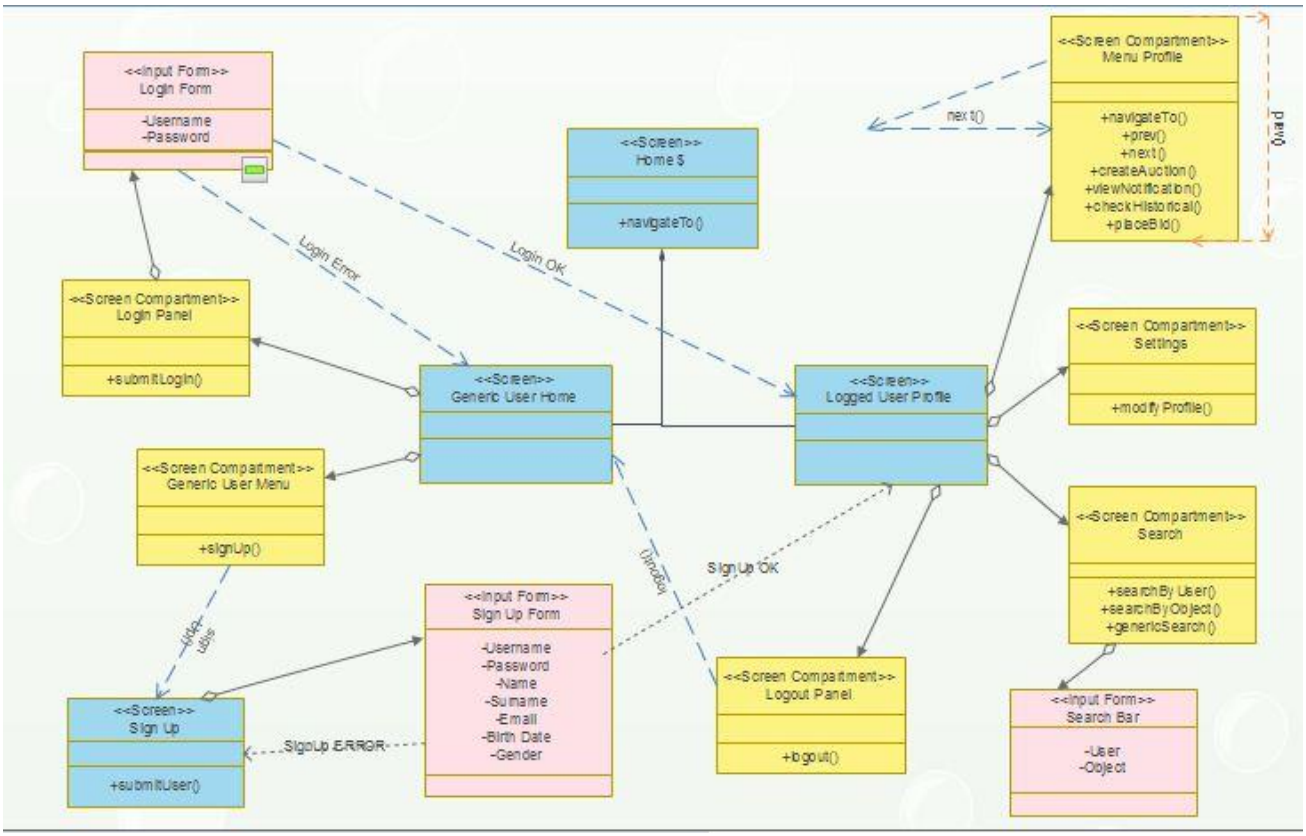
4.1. Homes, sign up/ login and search operations

We can see below the Diagram that represents the different home pages based on the type of user (logged or not) that is interacting with the system. Both pages inherit from a Home screen that is also marked with the landmark *$*, that indicates its reachability from every page of the system. In order to have an easier interpretation of the schema we decided to draw some components that are contained in all the screens, but we decided to draw them only in this UX Diagram, if not, it would have been very difficult to understand other diagrams. Here is a list of these elements:

- **Login Panel** and **Generic User Menu**:
  They are in all generic user's pages;

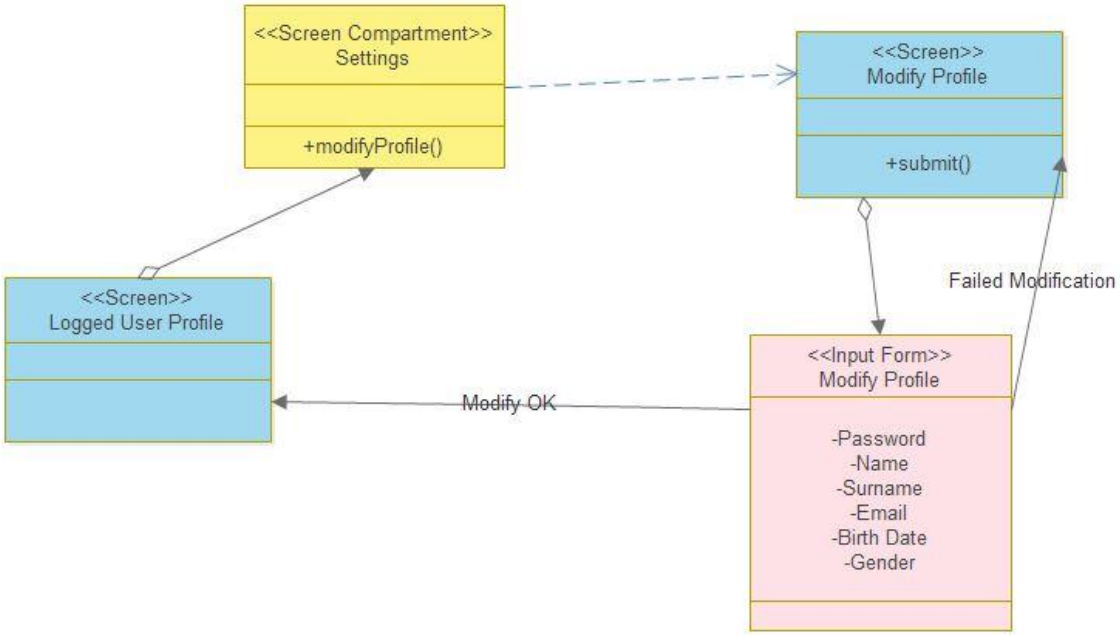
- **Settings, Search Panel** and **Log Out Panel:**
  They are in all logged user's pages;

Moreover, from a *Generic User Home,* we can log in through the *Log In Panel* and, if there is no error, we will be redirected to our profile page, in case of error in log in operations, we will be redirected to the *Guest Home* again. If we've already logged in we can perform a logout through the *Log Out Panel*. *Logged User Profile* contains the principal functionalities for a logged user, like create an auction, bid for an existing auction, view notification and check historical.
The *Settings* screen allows users to modify settings about their profile.
The last functionality expressed by this UX Diagram is the possibility of signing up reachable from the *Generic User Menu*: clicking on the related link the *Sign Up* screen is shown; this screen contains a form to be fulfilled with all the data of the new user; after that we will submit the data to the system and it redirects us to our home page if there aren't errors in sign up operations, otherwise it redirects us to the *Sign Up* screen again.


<<Input Form>>
Login Form
-Username
-Password
<<Screen Compartment>>
Login Panel
+submitLogin()
Login Error
Login OK
<<Screen>>
Home $
+navigateTo()
<<Screen Compartment>>
Menu Profile
+navigateTo()
+prev()
+next()
+createAuction()
+viewNotification()
+checkHistorical()
+placeBid()
next()
prev()
<<Screen Compartment>>
Settings
+modifyProfile()
<<Screen>>
Generic User Home
<<Screen>>
Logged User Profile
<<Screen Compartment>>
Generic User Menu
+signUp()
<<Screen Compartment>>
Search
+searchByUser()
+searchByObject()
+genericSearch()
SignUp
Logout
SignUp OK
<<Input Form>>
Sign Up Form
-Username
-Password
-Name
-Surname
-Email
-Birth Date
-Gender
<<Screen>>
Sign Up
+submitUser()
SignUp ERROR
<<Screen Compartment>>
Logout Panel
+logout()
<<Input Form>>
Search Bar
-User
-Object

4.2. Profile Managing Operations

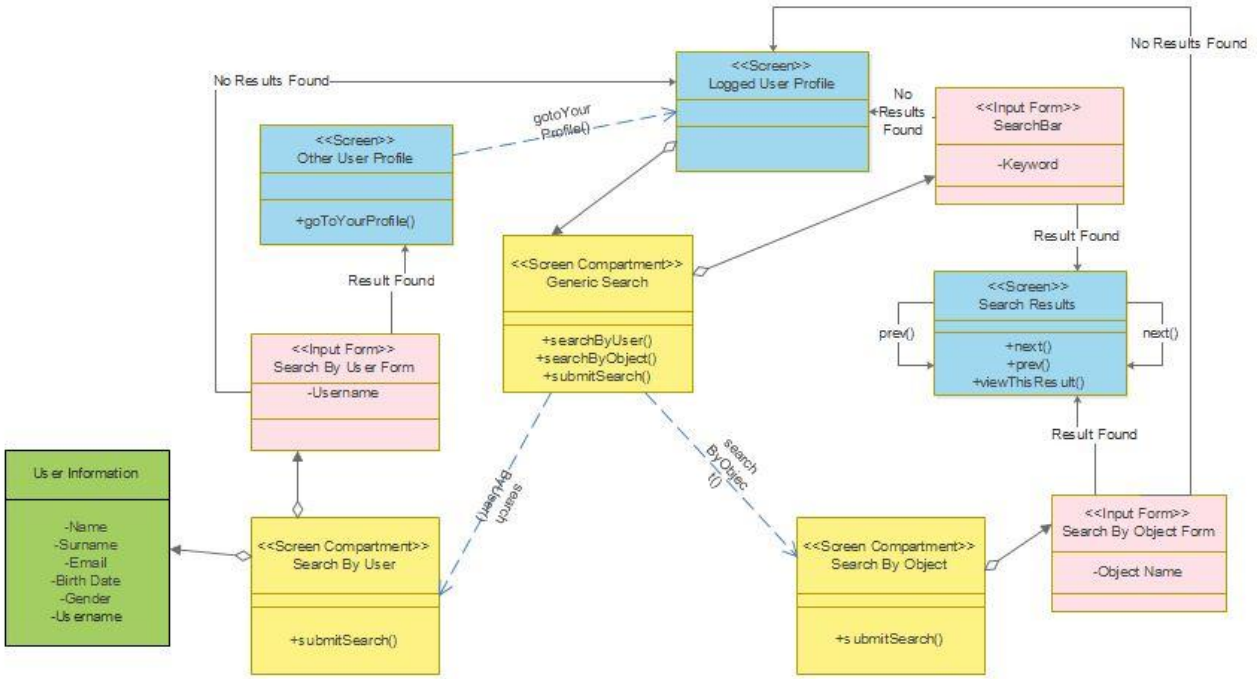
This part of the UX diagram aims to better understand the reader how happens in MeteoCal the operations of profile editing.



Starting from *logged user profile*, we can have access to the screen compartment *settings,* where it's possible to access the section in which you edit your profile, through the screen *modify profile* and filling out the form *modify profile form*, connected to it.

4.3. Search operations

This UX diagram aims to better explain various types of research that a user can perform using GuessBid: the system allows users to perform a research by username, by auction object or simply by a keyword.



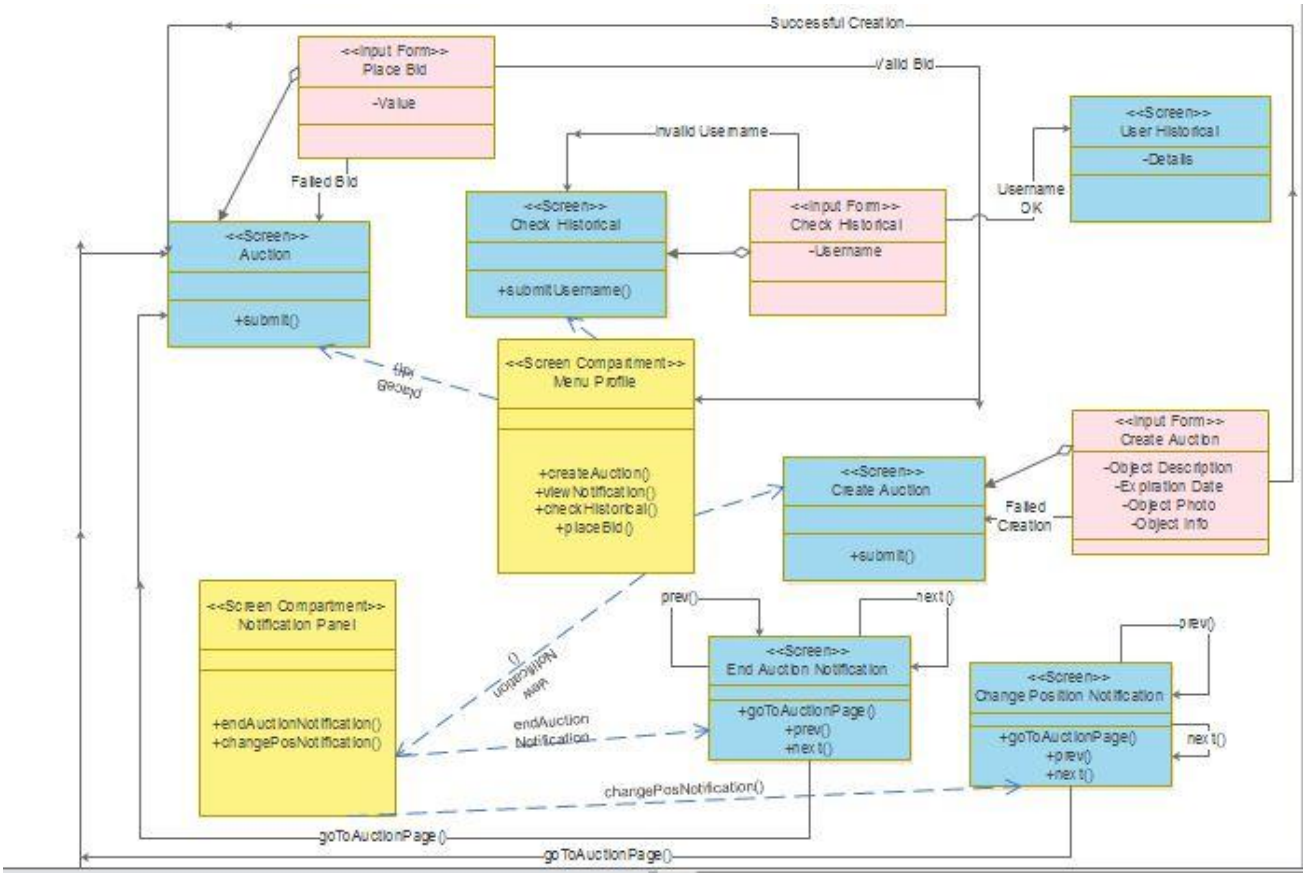
From the personal profile, a user can choose between three different types of research: passing through *generic search* the user can reach the other two screen compartment: *search by username* and *search by object*, and from there, filling the respective form, user can see the results and navigate through them.

The user can also search directly from *generic search*, inserting in the relative form the keyword and, if there are, being redirected to the page that contains related results.

If the research by username gives a valid result, the user will be redirected on *profile*, which represent the profile of the person associated to the username found.

4.4. Personal Profile Subsystem

In this paragraph we will describe how the user can enjoy GuessBid, starting from his personal page:



Starting from *Menu Profile*, the user can create a new auction, clicking on *Create Auction* , and filling the relative form; if the operation fails the user returns on *Create Auction* screen, otherwise he will be redirected on the *Auction*'s page.
Another available feature in this section is the consultation of the historical of a user, reachable from the *Check Historical* screen, filling its relative form with a valid username.
A user can also bid for an existing auction, starting from *Auction* page and inserting a valid amount in the related form; if the operation will be succesfull, the user will be redirect to his profile's page with a confirmation message.
The last feature available in this section of GuessBid is the *Notification Panel*, where a user can see both *End Auction Notification* and *Change Position Notification*; the user can also move through these notifications and reach the *Auction*'s page to whom notifications are associated.

5. Sequence diagrams

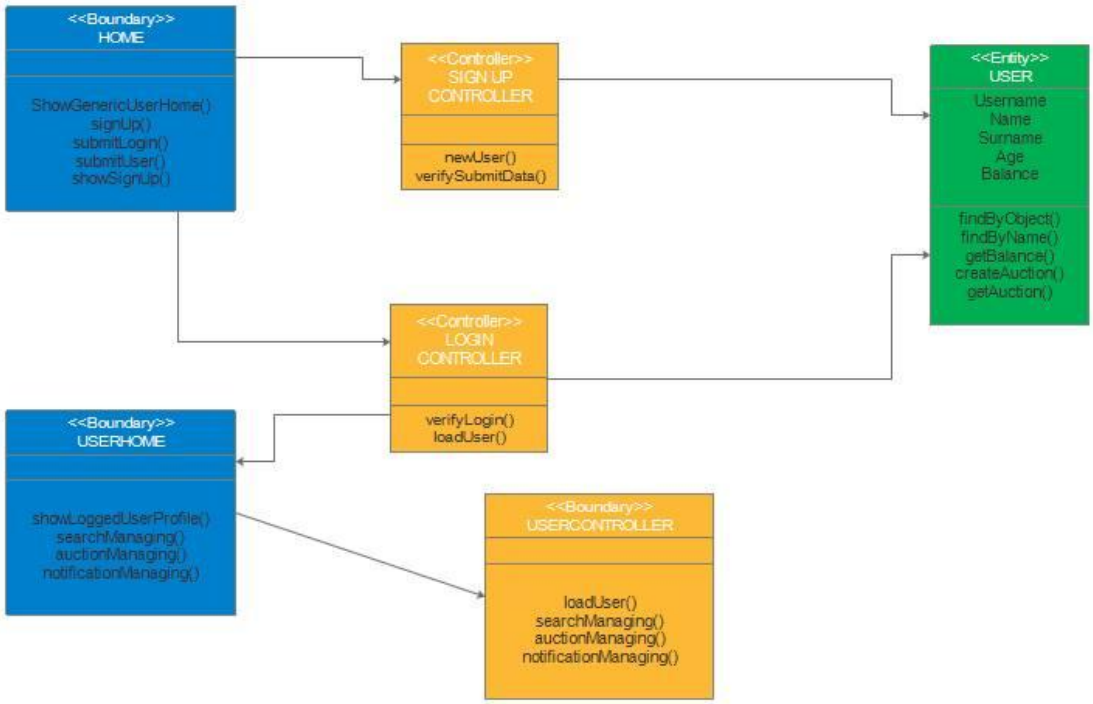
This subsection aims to present the various components of the system through the use of some diagrams ECB (boundary - control - entity).
The choice of including in this design document diagrams ECB is due to the 1: 1 correspondence of elements of this type of diagram with those of the MVC pattern, which will be used in the implementative part, in particular:

-boundaries correspond to the user interface, and consequently to the view;

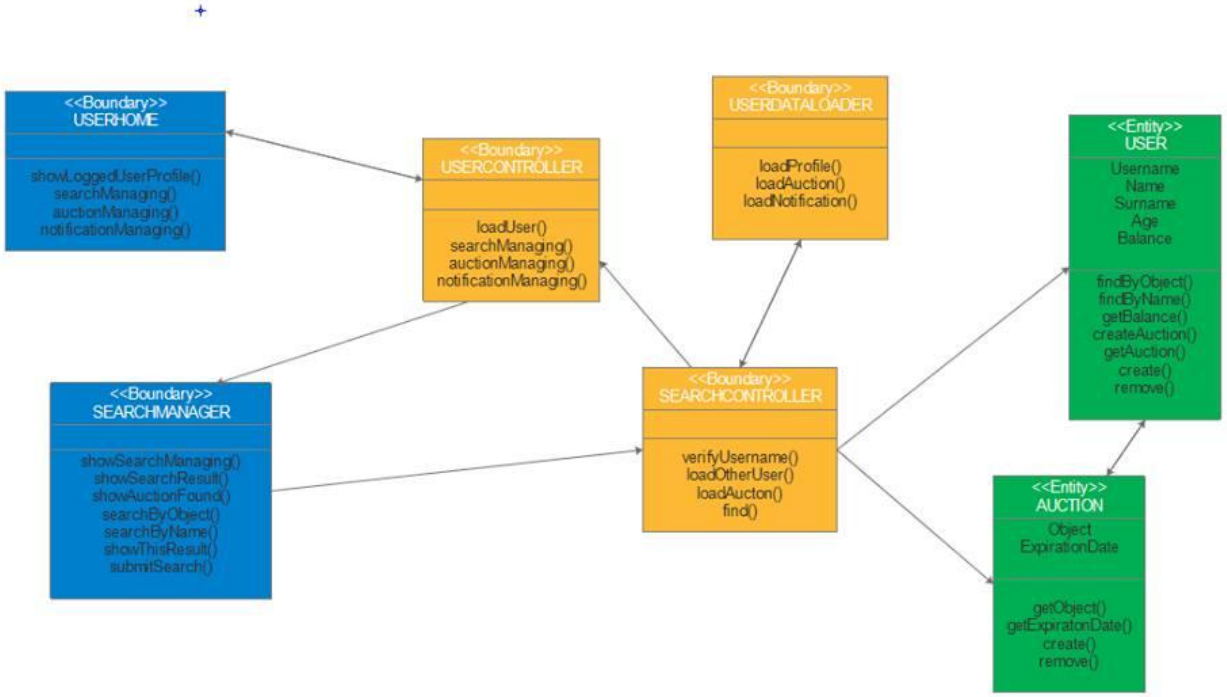
-controls correspond to the controller;

-entities correspond to the model;

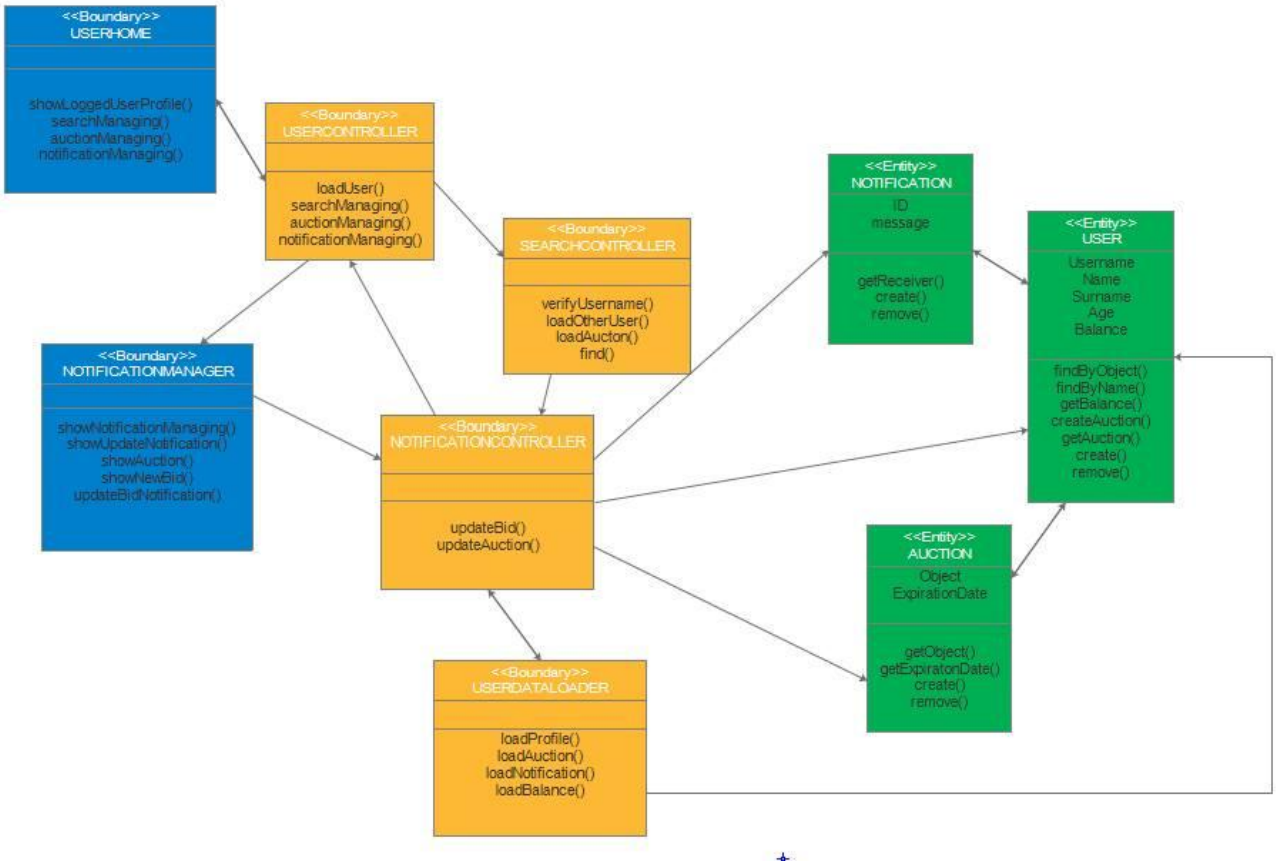
5.1. Log in / Sign up



5.2. *Auction and Search Managing*



5.3 *Notification Managing*



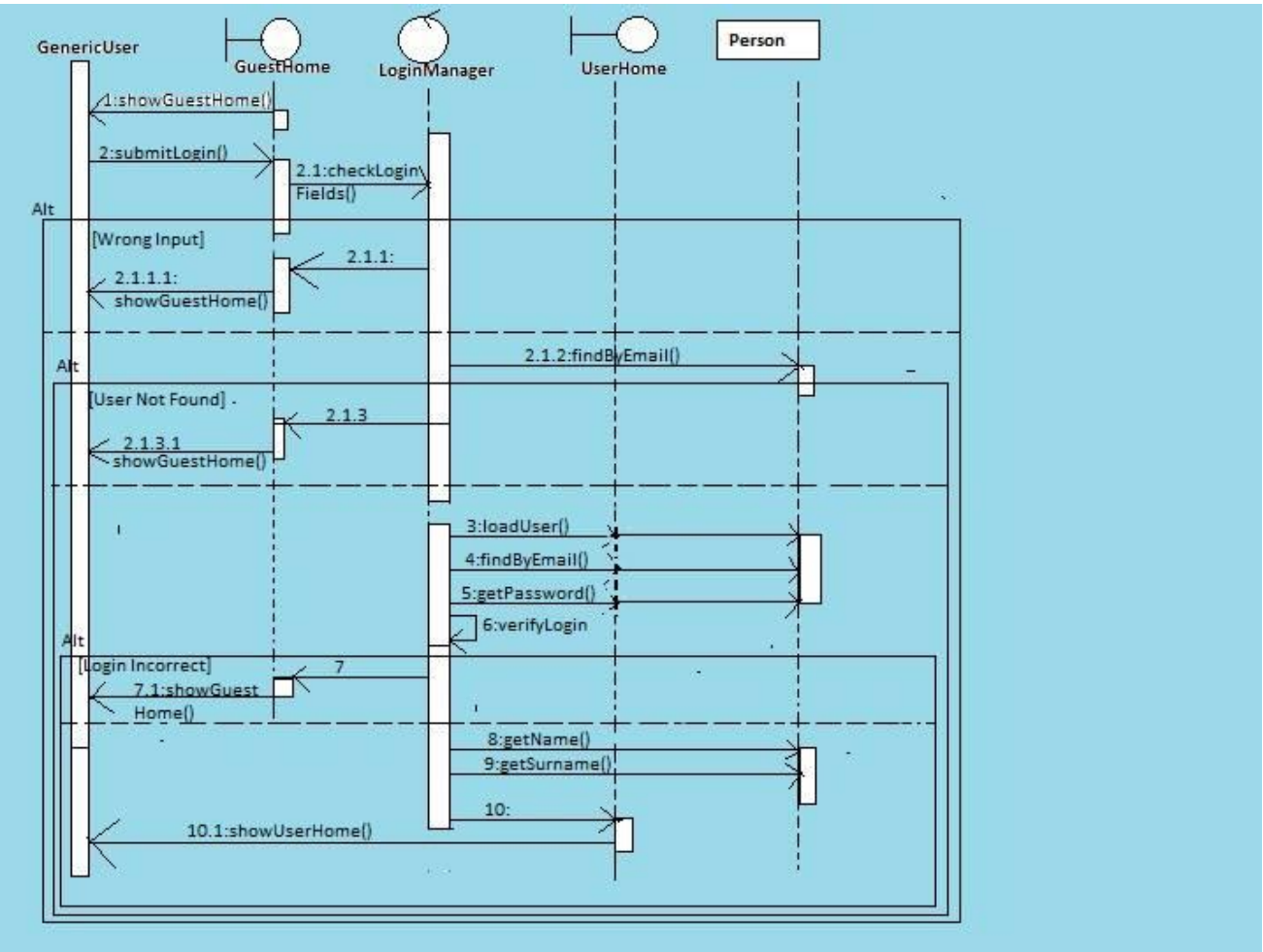
6. Sequence diagrams

We drafted some sequence diagrams to allow the reader to understand in a detailed and accurate way the system GuessBid and diagrams BCE described in the previous paragraph.

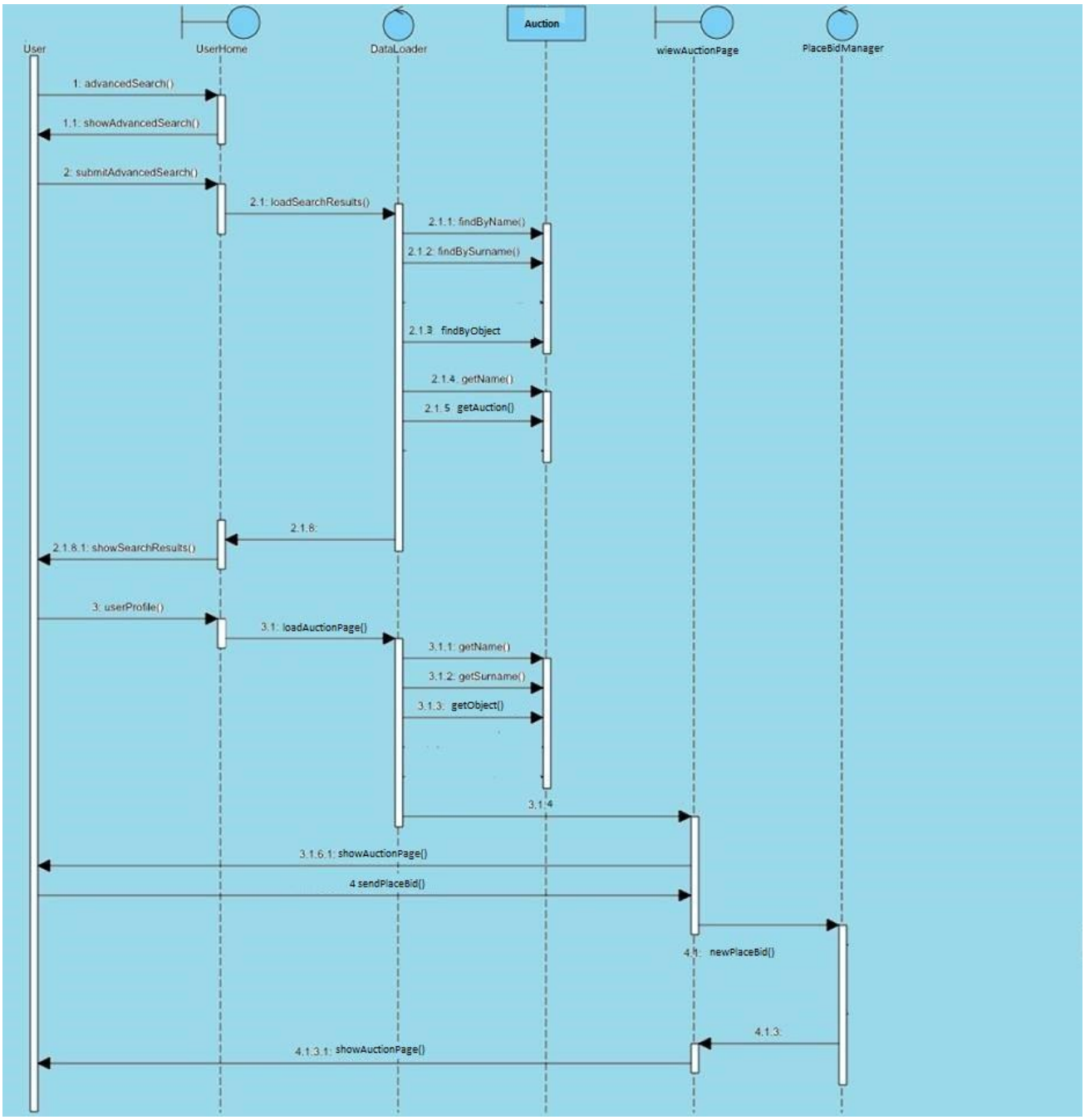
6.1. Log in sequence diagram

Activity diagram that describe the interactions between user and system during login operation

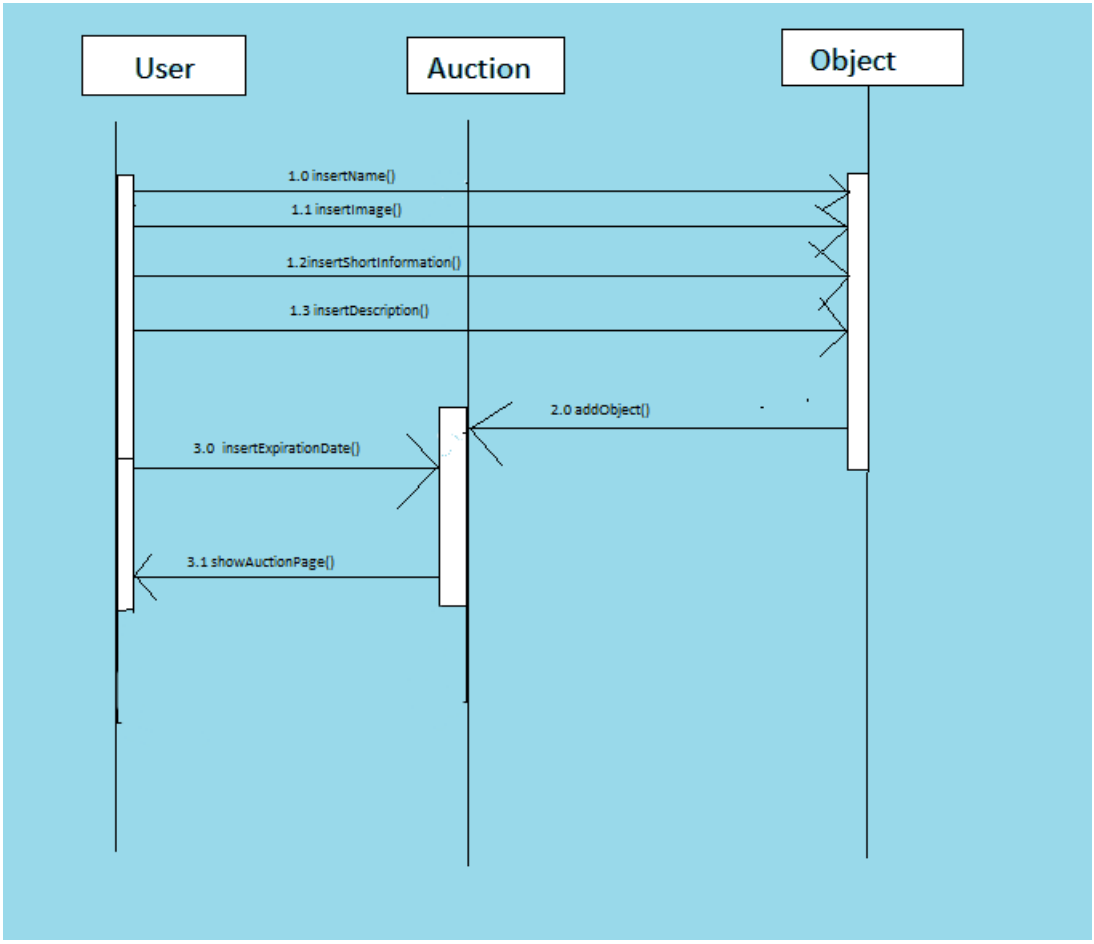
NOTE: in this diagram we used the word “guest” with the meaning of “not registered user”, unlike what we said in the RASD



6.2. Place Bid activity diagram



6.3. Create Auction sequence diagram



7. Final Considerations

Preparing this document, we have focused our attention more on the drafting of the three types of diagrams contained in the previous paragraphs: UX diagrams, BCE diagrams and sequence diagrams; this because in our opinion the drafting and good understanding of these diagrams is enough to understand in an acceptable manner the project in its pre- implementative phase and clarify our choices regarding application design.

Furthermore, we believe that the drafting of other more detailed diagrams would not have made more understandable GuessBid, for as it is currently structured, so we drafted only what we have just described previously.