GuessBid

Requirements Analysis

And

Specification Document

Authors:

Telemtaev Ruslan
Kazakov Sergey

Summary

1. Introduction

1.1 Description of the given problem

We will project GuessBid which implementing an inverse auction system. An inverse auction works like a regular auction. The difference is that in an inverse auction a user has to propose the lowest unique bid to win the auction.

The system that will be provide the registration for new use with virtual credit of 100. Create an auction for each good he/she has, defining an expiration date after which the auction expires. Every user can browse the existing auctions and he/she can bid for the goods of these auctions.

1.2 Goals

If we think about possible users, we think that GuessBidl has to provide this main features:

- Registration of a person to the system;
- Create auction;
- Browse the existing auction and place a bid ;
- Informed about the current status of user biddings for opened auctions.
- Notify all participants taking part in auction about relusts;
- Extend the system to keep track of the story of previous auctions.

1.3 Domain Properties

We suppose that these conditions hold in the analyzed world:

- A users can create auction
- A users can browse auction
- A users can place a bid in auction (it is forbidden for a user to bid for an auction created by himself)
- While auction have been going , system informed about the current status of user biddings for opened auctions.
- When auction have finished , system Notify all participants taking part in auction about relusts;
- A user can browse track of the story of previous auctions.

1.4 Glossary

First of all we have to define some words that will be used in our documents.

- **User:** for user we mean a person already registered in the system, so that has a profile and can use all the functionalities described below (see Functional Requirements).
- **Administrator:** the administrator of the event is the person allowed to create, delete and update events and send requests to other users
- **Profile:** A personal page created for individual use you add information about you
- **Auction:** This is an event that can be created an individual by user . It allows all other users take part in this event.
- **Add Auctiont:** Add Auction is a service that provides users create their own Auction.
- **Bid:** A prospective user indication or offer of a price that user will pay to purchase property at auction.

- **Notification:** that the way to infrom user about something that gives official information from Administrator(System). The act of notifying someone.
- **Previous auction:** track of the story of previous auctions.

1.5 Proposed system

We propose a web platform that will give to linked people the services described below.

Users will be able to create an auction for each good that user has, defining an expiration date after which the auction expires. Every user can browse the existing auctions and he/she can bid for the goods of these auctions. And the system will produce the current status of user biddings for opened auctions and when auction have finished , system will notify all participants taking part in auction about relusts.

1.6 Identifying Stakeholders

Our “financial” stakeholder is the professor who gave us this didactical project. Our professor needs are to have, within the end of the semester, a product that at least works and we think the main thing that interests the professor is to show that we have understood the development process in all its parts and that we can carry out a project from the beginning to the end passing from all its development phases.

So we want to show that we can identify requirements and specifications, design our web application, implement it and then test it, providing all the documents that developers use in developing real software.

It still remains that we have to focus on some functionalities that the professor gave us to have an application that works.

We think that program will be useful because this kind of program create to help people to sell their “goods” for the lowest price for other particiants of this application. Also users can create infinity numbers o auctions.

1.7 Other considerations about the system

Few words about GuessBid system. So we thinkGuessBid has to be:

- Easy to use: we want to create simple interface, that east to understand while you are working in system.
- Notification: notify all event participants about current status of user biddings for opened auctions, and when auction have finished , system will notify all participants taking part in auction about relusts.
- Have a nice look and feel: we will try to make a well designed application in the sense that we hope it will be nice to see and will fit to any user.

2. Actors Identifying

The actors of our informative system are basically two:

- **User:** a person already registered in the system, so that has a profile and can use all the functionalities described below (see Functional Requirements).

- **Administrator:** the administrator of the event is the person allowed to create, delete and update auction

3. Requirements

1. **Registration of a person to the system:**
   - The system has to provide a sign up functionality.
2. **Add Auction to create new auction in system :**
   - The system will made possible to create auction;
   - The system will allow users browse all auction that user wants
   - The system will allow users to place a bid in auction that user take part
   - The system will allow to see all information about previous auctions

3. **Contacting system developers:**
   - The system will made possible for users to have feedback with system developers.
   - The system will allow users to see system developers contact information.

3.1 Functional Requirements

Functional requirements of MeteoCal concerning each defined actor:

- **User**:
  - ■ Log in;
  - ■ Modify his profile information;
  - ■ Create auction
  - ■ Browse all auction that user need
  - ■ Place a bid in auction

- **Administrator (System developer)**:
  - ■ Log in;
  - ■ Answer on feedback notification

  - ■ Delete or block users and auctions in system

3.2 Non Functional Requirements

3.2.1 User Interface

The interface of our application is for via web users. It has to provide a unified log in page that allowing accessing personal user profile page and all functionalities of our application. We want to create simple interface that is why we develop one main page with all functionalities which contain auction creation form, browse auction and previous auction . All forms (create auction, browse auction list , feedback) will be pop up.

3.2.2 Documentation

We will draft these documents to well-organize our work in the way to do in a fewer time the best work as possible:

- **Project Plan**: to define tasks and to show our organization and the timings of this process.
- **RASD**: Requirement Analysis and Specification Document, to well-understand the given problem and to analyze in a detailed way which are our goals and how to reach them.
- **DD**: Design Document, to define the real structure of our web application and its tiers.
- **JavaDoc comments in the source code**: to make anyone that wants to develop the platform or do maintenance on it understand the code.

3.2.3 Architectural considerations

We will develop the system using the Java EE platform. In particular, we will use EJBs to develop the business logic.

What is this information will be clearer watching the Class Diagram given below, because it could be considered as a basis for our database, but the precise ER Diagram will be drawn in the DD.

An Internet connection is needed to use MeteoCal.

4. Specification

Here some specifications for GuessBid that will state clearer how we can reach the goals listed above:

- A users can create auctions from main page of the GuessBid system
- A users can place a bid in auction that he is interested
- A users can browse all auction
- System notify all event participants about current status of user biddings for opened auctions, and when auction have finished , system will notify all participants taking part in auction about relusts
- .A users can modify their own register data

5. Scenarios Identifying

Here are some possible scenarios of Meteocal:

- Giovanni wants to organize auction and sell somthing . So he remembers that he used GuessBid few days and his friends too. So he decided to sell something in GuessBid.
- Giovanni wants to buy somthing for the lowest price he have not ever bought somthing .So he use GuessBid system so he browses list off auctions and tryto find the most interesting auction for him and try to place the lowest price for item in auction.

6. UML Models

6.1 Use Case Diagram

We can derive some use cases from the scenarios identified in the previous paragraph:

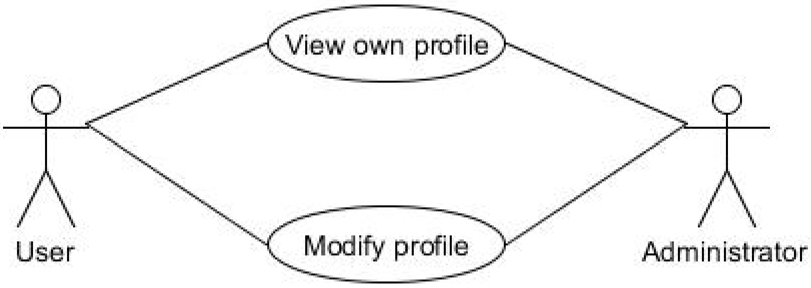
- Sign up;
- Log in;
- Add auction;
- Modify profile information;
- Place a bid
- Browse list of auctions
- Notify user about result of auction;
- Provide a feedback;
- Browes list of previous auctions
- Notify about current status of user biddings for opened auctions

6.2 Use Cases Description

We describe in a detailed way below the main use cases. Some use cases that extends other use cases and some others are omitted because they are really similar to others. It is important to understand that all references to "events", "buttons" or "input forms" are only hypothesis to make the situation clearer and to help the reader to draw a visual picture in his mind of what we are talking about, real pages and events, page structures will be well defined in the Design Document. We refine here the use case "Log in", because it is in all other sections:

Log in	**Name**
User or administrator.	**Actors**
The user has successfully signed up to the system.	**Entry Conditions**
• The user\administrator	**Flow of events**

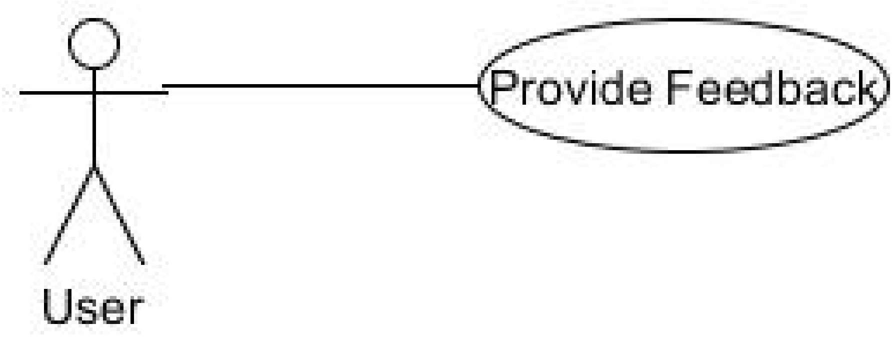
opens the home page of the platform; • The system shows him the page; • The user\administrator enters his e-mail address and password in the input form provided; • The user\administrator clicks the button “log in”. • The system shows the profile\administrator page.	
There are no exit conditions.	**Exit conditions**
The information inserted in the form is wrong, an error message is shown.	**Exceptions**



6.2.1 Provide a feedback

Provide a feedback	**Name**
User.	**Actors**
The user has to be logged in.	**Entry Conditions**

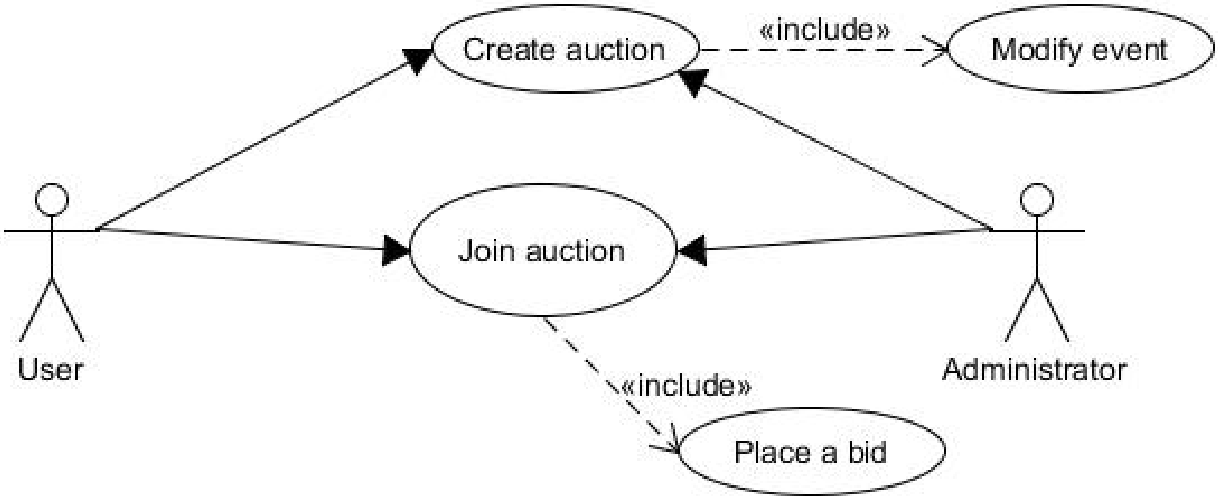
The user clicks on "Give a Feedback" on the side of the chosen reply; The system loads a page containing an input form which the user can fill with the mark of the performance and a description; The user fulfills the form and clicks on "Ok"; The system informs the user of the result of the operation; The system reloads the help request page.	**Flow of events**
The page is reloaded with the right information. The information is stored.	**Exit conditions**
There are no possible exceptions.	**Exceptions**



6.2.2 Auction Managing

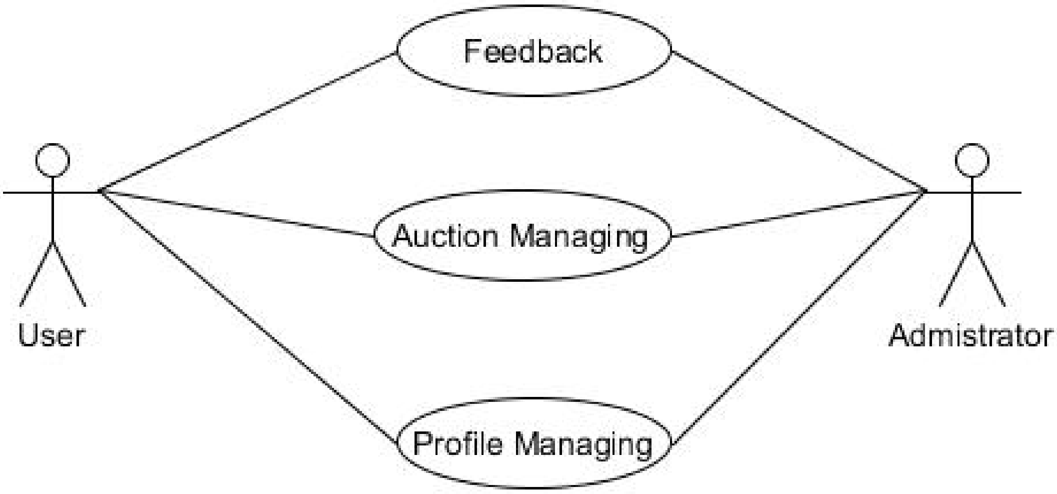
Add Auction	**Name**
User.	**Actors**
The user has to be logged and be into his profile page.	**Entry** **Conditions**

• The user can click on “Add auction ”; • The system loads a page with a form that allows to modify auction information ; • The user fill the form ; • The user clicks on “Submit ”; • The system reloads the auction page and shows the result of the operation.	**Flow of events**
The new auction is stored and accessible by users.And users can find this auction by browsing auction list . The auction is created.	**Exit conditions**
• The user doesn't click on “Submit” and loads another page. In this case data is lost; • The user fill an information which is not yet in the system; • The user has emptied some mandatory fields. In this case an error message is shown.	**Exceptions**



Browse list of auctions	**Name**
User.	**Actors**
The user has to be logged in.	**Entry** **Conditions**
• The user clicks on " list of auctions " in personal page	**Flow of events**
The user now in page with all auctions	**Exit conditions**
There is no exception.	**Exceptions**

Place a bid	**Name**
User.	**Actors**
The user has to be logged in.	**Entry** **Conditions**
• The user clicks on "Place a bid" in list of auctions, where for each auction matches button "Join Auction" ; • The user clicks on the button "Place a bid" and choose amount(value) of his bid ; • The user clicks on "Submit bid"; • The system shows a message like "Bid accepted "; .	**Flow of events**
The bid placed in auction	**Exit conditions**
There is no exception.	**Exceptions**



Browse list of previous auctions	Name
User.	Actors
The user has to be logged in.	Entry Conditions
• The user clicks on “ list of previous auctions ” in personal page	Flow of events
The user now in page with all previous auctions he have ever taken part .	Exit conditions
There is no exception.	Exceptions

Notify user current status of user biddings for opened auctions	**Name**
User, Administrator	**Actors**
The user has to be logged in. User have to place a bid in auction .	**Entry** **Conditions**
The system provides updates when such current position changes as a result of biddings from other users	**Flow of events**
The system provides updates about current position in auction	**Exit conditions**
There is no exception.	**Exceptions**

Modify auction	**Name**
User.	**Actors**
The user has to be logged and be into his profile page.	**Entry** **Conditions**
• The user can click on "Modify"; • The system loads a page with a form that allows to modify event information; • The user modifies what he wants; • The user clicks on "Save Changes"; • The system reloads the event page and shows the result of the operation.	**Flow of events**

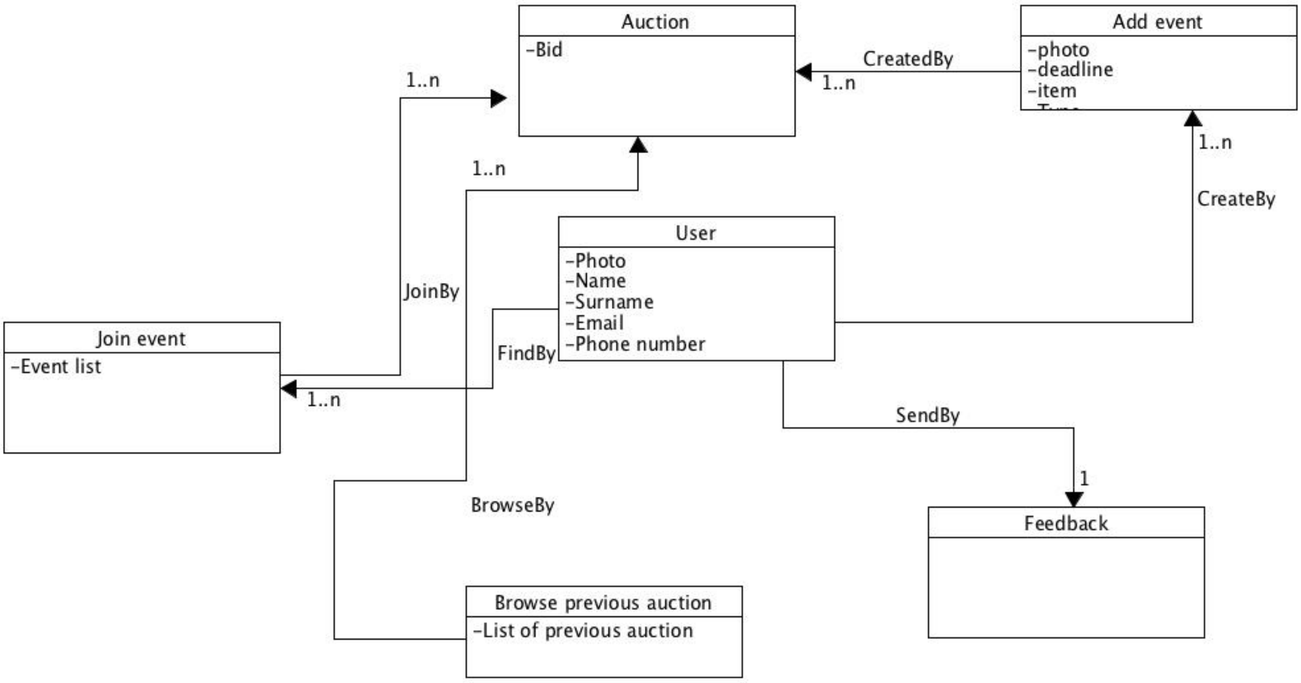
The new information is stored and accessible by users. The event page is reloaded and updated with the new information.	**Exit conditions**
• The user doesn't click on “Save Changes” and loads another page. In this case data is lost; • The user adds an information which is not yet in the system; • The user has emptied some mandatory fields. In this case an error message is shown.	**Exceptions**

Notify user about result of auction	**Name**
User, Administrator	**Actors**
The user has to be logged in. User takes part in auction.	**Entry Conditions**
• After auction have finished system notify participants about resulsts of this auction	**Flow of events**
Notification about resulsts of auction	**Exit conditions**
There is no exception.	**Exceptions**

6.2.3 Profile Managing

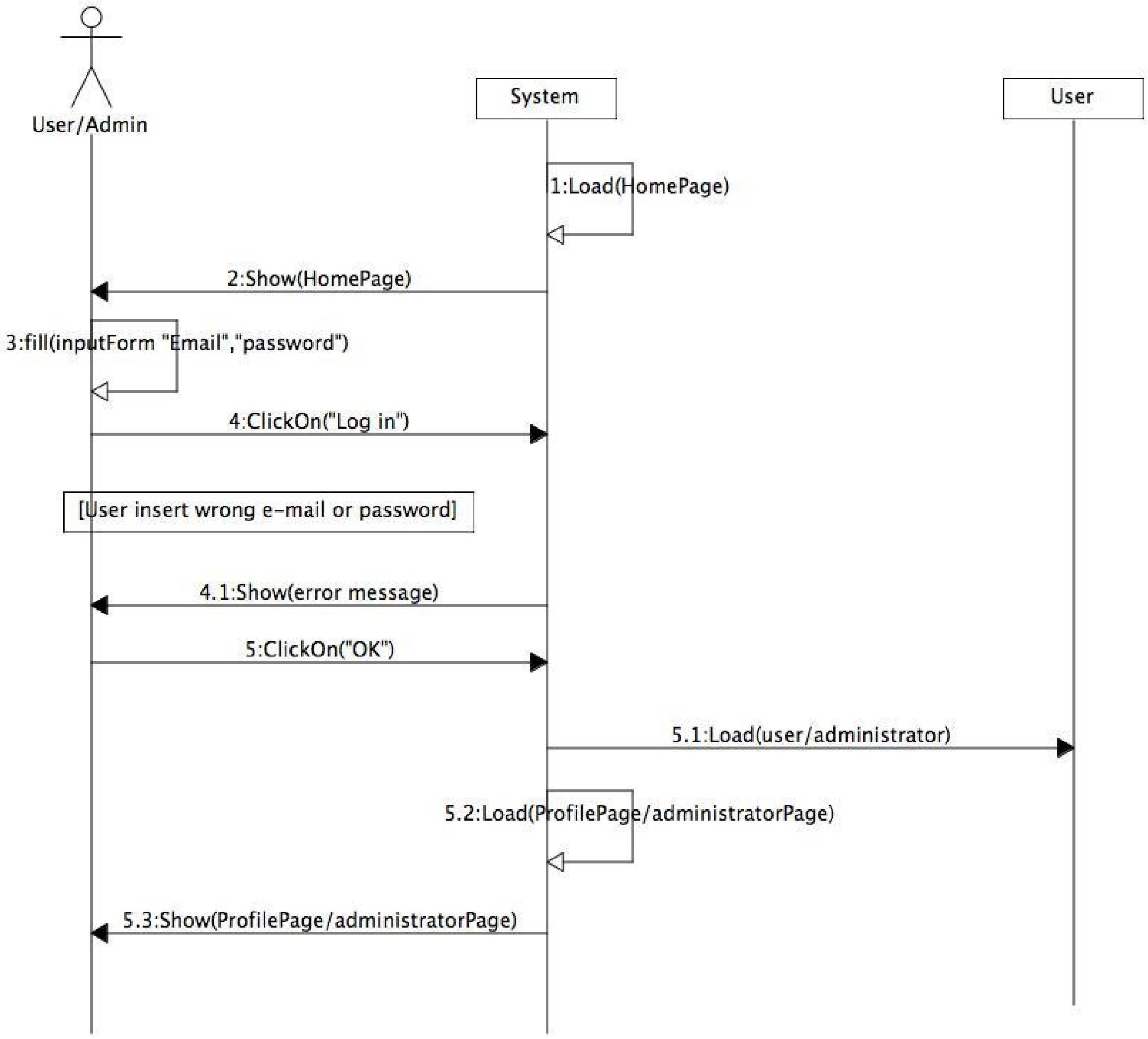
Modify profile information	**Name**
User.	**Actors**
The user has to be logged and be into his profile page.	**Entry** **Conditions**
• The user can click on "Modify"; • The system loads a page with a form that allows to modify profile information; • The user modifies what he wants; • The user clicks on "Save Changes"; • The system reloads the profile page and shows the result of the operation.	**Flow of events**
The new information is stored and accessible by users. The profile page is reloaded and updated with the new information.	**Exit conditions**
• The user doesn't click on "Save Changes" and loads another page. In this case data is lost; • The user adds an information which is not yet in the system; • The user has emptied some mandatory fields. In this case an error message is shown.	**Exceptions**

6.3 Class Diagram

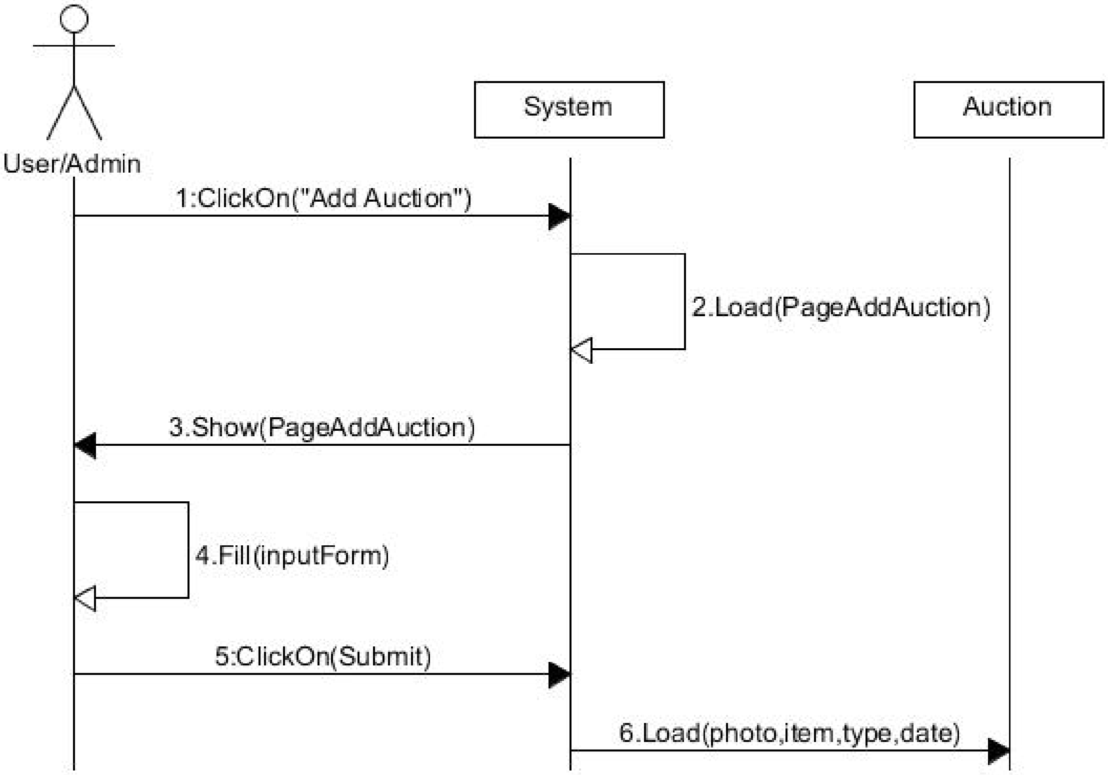


6.4. Sequence Diagrams

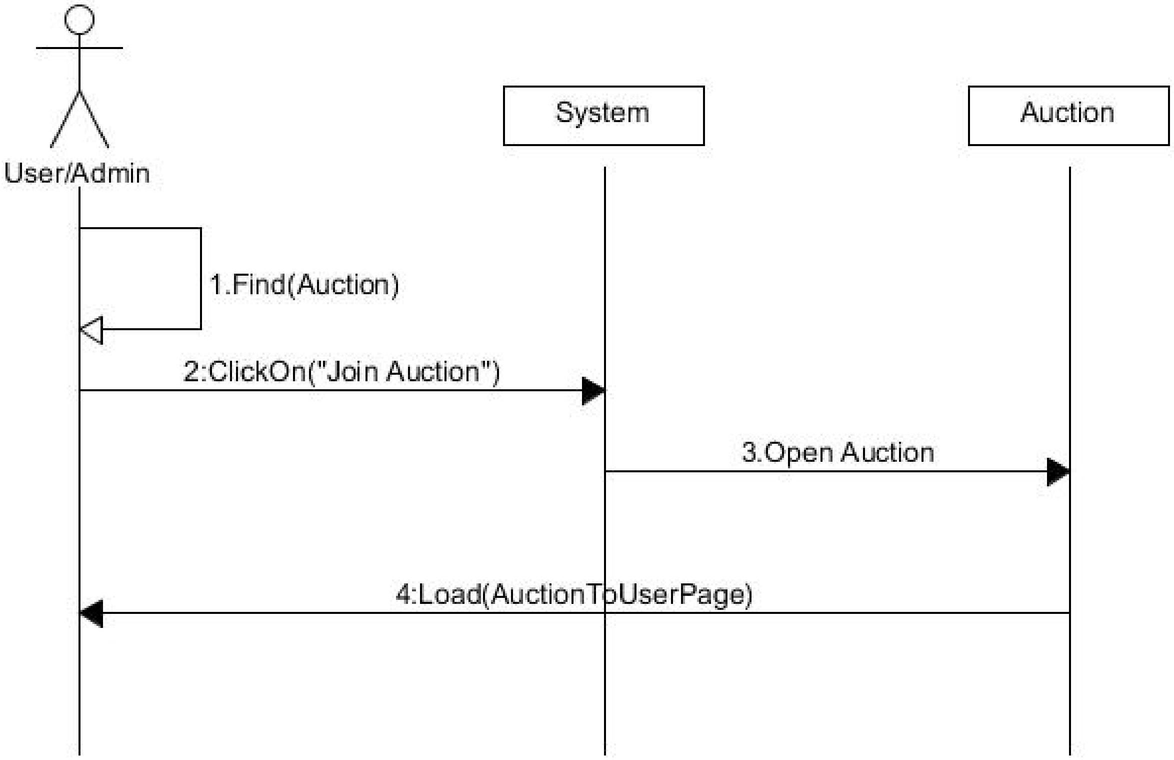
6.4.1. Log in


User/Admin
System
User
1:Load(HomePage)
2:Show(HomePage)
3:fill(inputForm "Email","password")
4:ClickOn("Log in")
[User insert wrong e-mail or password]
4.1:Show(error message)
5:ClickOn("OK")
5.1:Load(user/administrator)
5.2:Load(ProfilePage/administratorPage)
5.3:Show(ProfilePage/administratorPage)

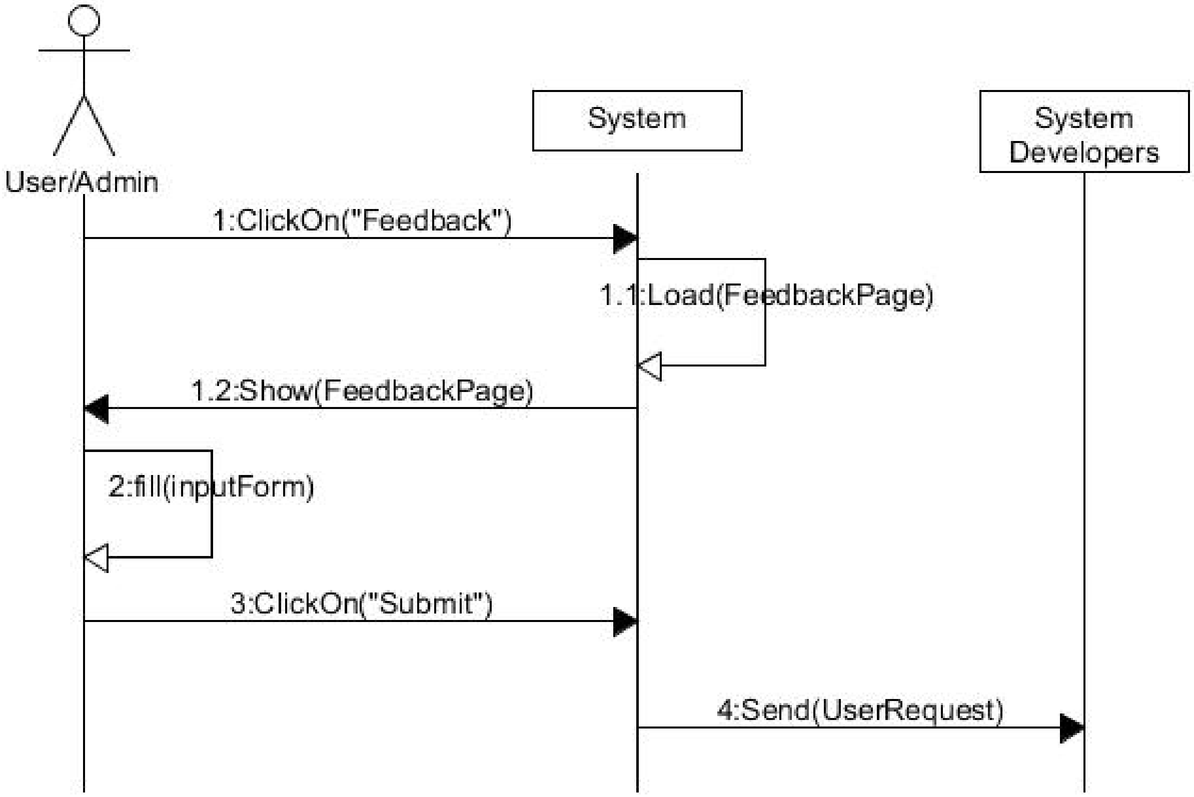
6.4.2 Add Auction



6.4.3 Join Auction



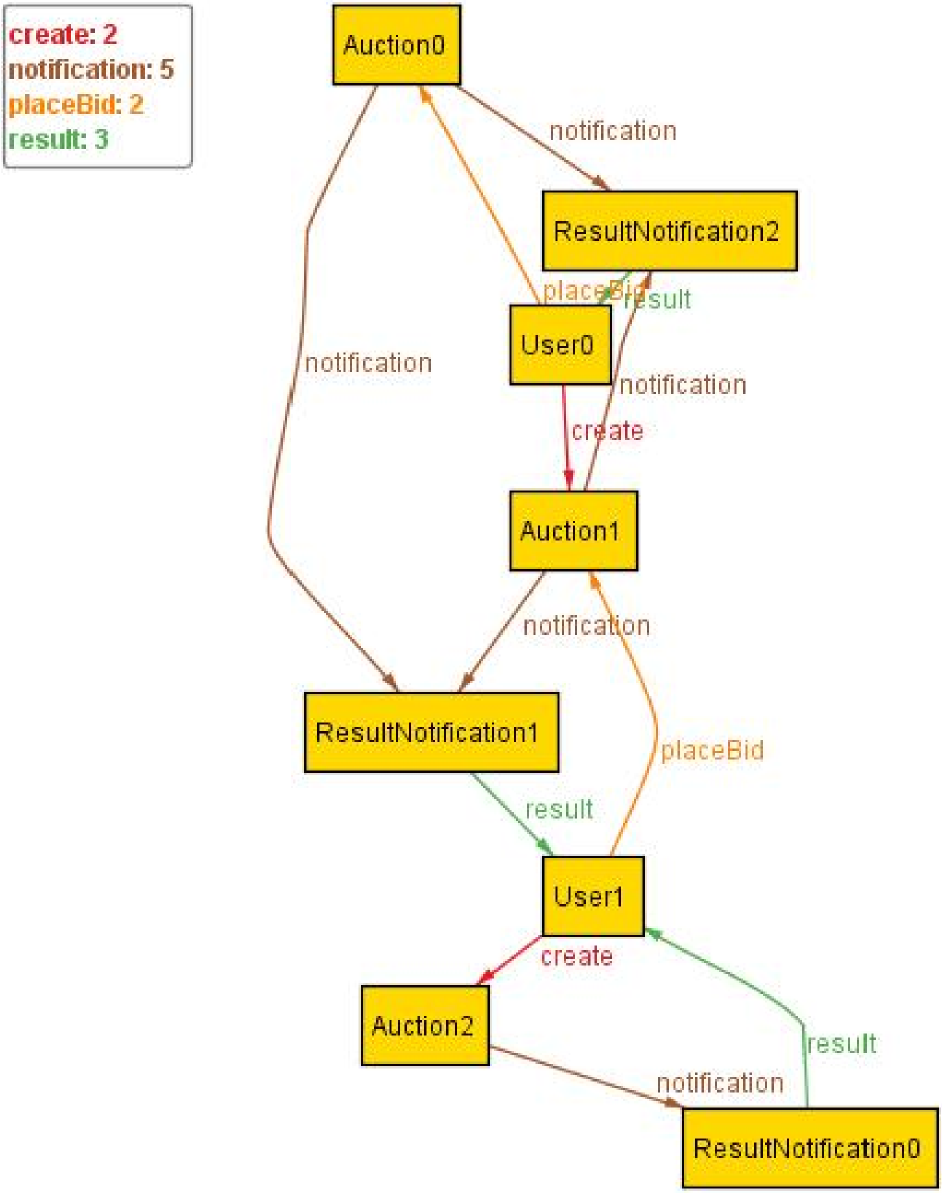
6.4.4 Provide a Feedback



7. Alloy Modeling

In this paragraph we try to program Class Diagram using Alloy Analyzer.

```
module GuessBid

sig User {
create: lone Auction,
placeBid: lone Auction
}{create!=placeBid}

sig Auction{
notification: some ResultNotification
}

sig ResultNotification {
result: some User
}

//FACTS
fact{
        (one a: Auction | one r: ResultNotification | r in a.notification)
}

pred show(){}
run show for 3
```


create: 2
notification: 5
placeBid: 2
result: 3
Auction0
notification
ResultNotification2
placeBid
result
User0
notification
notification
create
Auction1
notification
ResultNotification1
placeBid
result
User1
create
Auction2
result
notification
ResultNotification0

8. Used Tools

The tools we used to create this RASD document are:

- Microsoft Office Word 2007: to redact and to format this document;
- UMLet: to create the UML diagrams;
- Alloy Analyzer 4.2: to prove the consistency of our model;