Dario LORENZONI

Raffele NOCERINO

Marco PETTORALI

Riccardo XEFRAJ

PC BOOKING

Task 1

Index

[Description of the system 2](#_Toc22224042)

[System requirements 2](#_Toc22224043)

[Main actors 2](#_Toc22224044)

[Use Cases 3](#_Toc22224045)

[ER diagram 5](#_Toc22224046)

[Entities: 5](#_Toc22224047)

[Relations: 5](#_Toc22224048)

[Tables 6](#_Toc22224049)

[UML Diagram 7](#_Toc22224050)

[Classes 8](#_Toc22224051)

[User manual 9](#_Toc22224052)

[Login 9](#_Toc22224053)

[Woking with the application 9](#_Toc22224054)

[Removing a reservation 11](#_Toc22224055)

## Description of the system

The main purpose of this project is to provide to the students of a university a booking system for using computers in informatics rooms of the campus in the spare time. After the registration to the service, students can monitor the available computers. If a user decides to book a computer, the selected machine will provide a login screen in which only that student can put his credentials; otherwise, the system will reject the session. A computer can only be reserved for an hour at a time; that is in order to prevent users to reserve a PC for too many consecutive hours, not letting other users to use the service. The system keeps track of the total number of reservations made for each PC, in order to analyze which computer are used the most and most likely to be worn out.

## System requirements

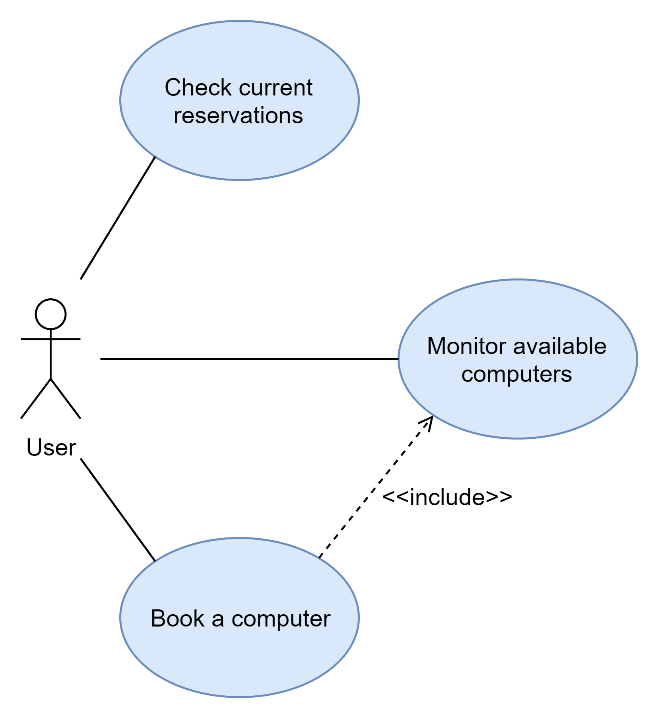
### Functional requirements:

* Users can sign up to the system by putting their personal information in a sign-up service
* A user can sign in using a login service, in order to book computers using his name.
* A user can see in the main page of the application all his reservations, except for the expired reservations.
* A user, from the reservations list, can delete one or more future reservations just selecting them.
* A user can insert his desired date and hour in order to see all the available PCs for each room in that time. If no PCs are available in the specified date and hour, the system displays a message.
* After the insertion of date and hour, a user can see all the available rooms. Choosing a room from the displayed list, a user can reserve a PC in that room. If something occurs during the reservation, the system shows an error message.
* After the reservation, a user can see the map of the room and in particular, it can see the position of the booked PC in the room, which is indicated by the yellow color.

### Non-functional requirements:

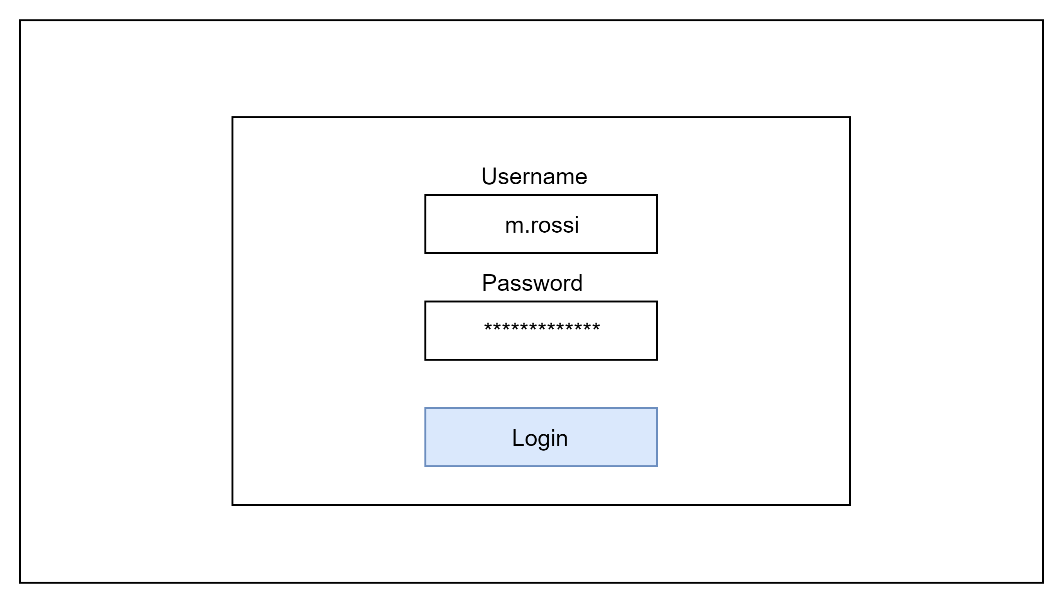
## Main actors

* User (a student)
* Computer (each one in the campus)

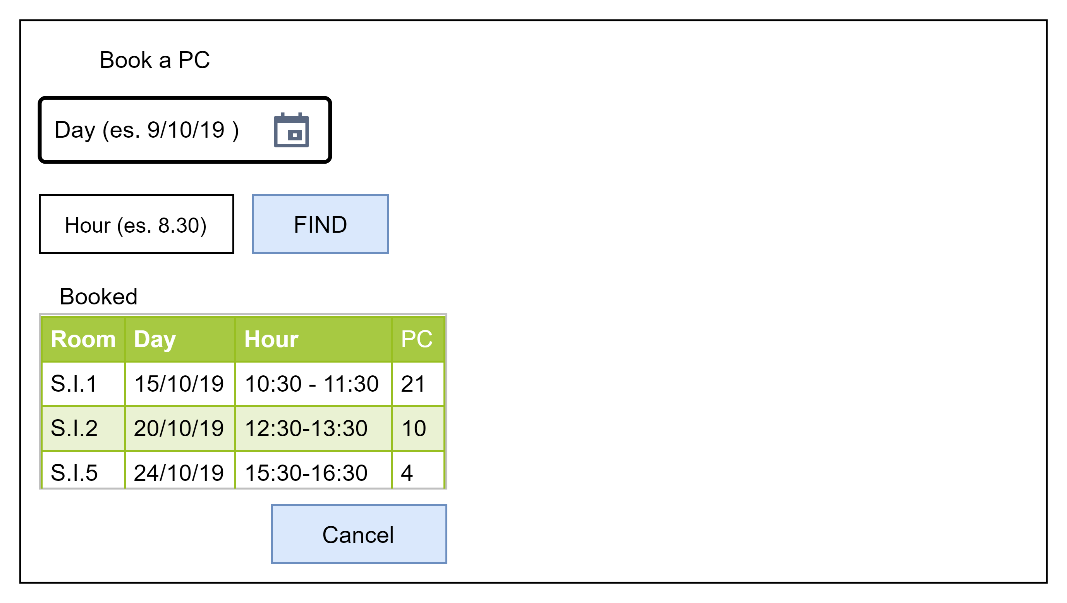


## Use Cases

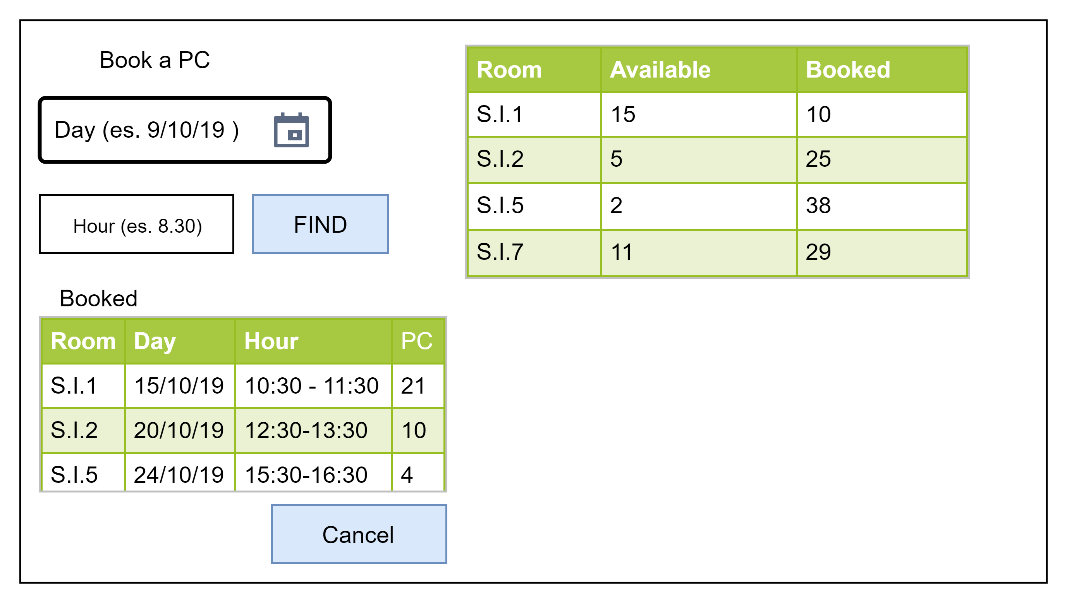
1. The System shows to the user a login screen
2. The User inserts its university credentials



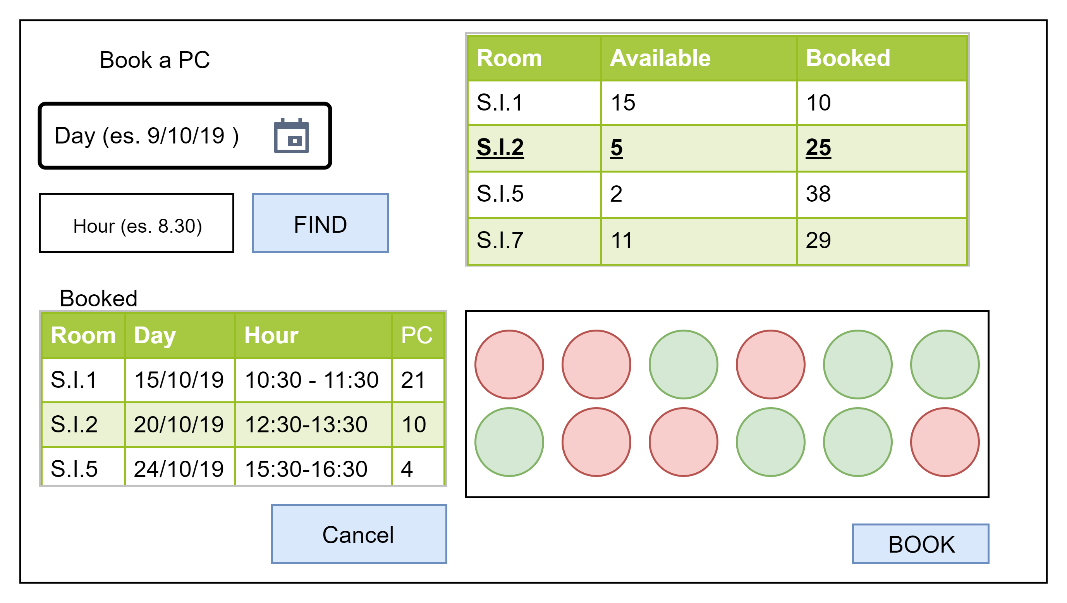
1. The System shows to the User the search form with the Date and Hour boxes
2. The System shows to the User its active reservations
3. The User inserts a Date and a Hour to book a computer



1. IF the User clicks on FIND button
   1. FOREACH available informatics rooms in the selected hour
      1. The System shows the name of the room
      2. The System shows how many computers are available in the room
      3. The System shows how many computer are already booked in the room

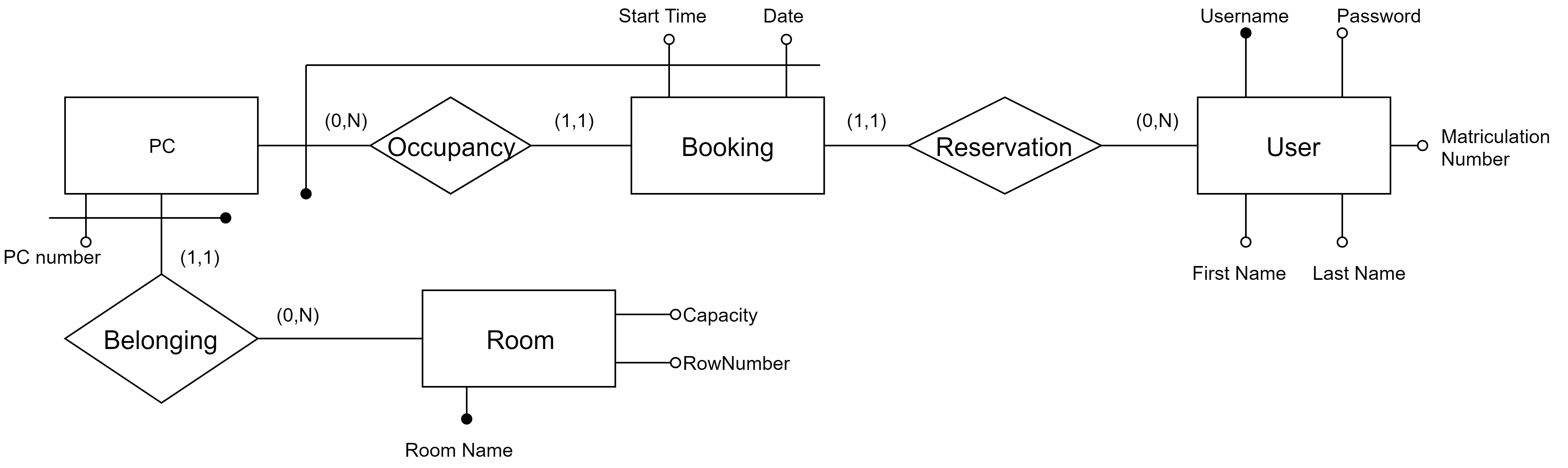


1. IF the Users clicks on a room
   1. The System shows a map of the available and not available computers



1. IF the User clicks on BOOK button
   1. The System reserves a computer for the user

## ER diagram



### Entities:

* PC: it collects the information about the name and the location of the computer in the college;
* Booking: it gathers all the reservations made by the students, by storing the users’ id, the selected machine’s id, and the date and time of the booking;
* User: it collects the users’ personal data, including their name, credentials and matriculation number;
* Room: it indicates name, shape and capacity of each informatics room in the campus.

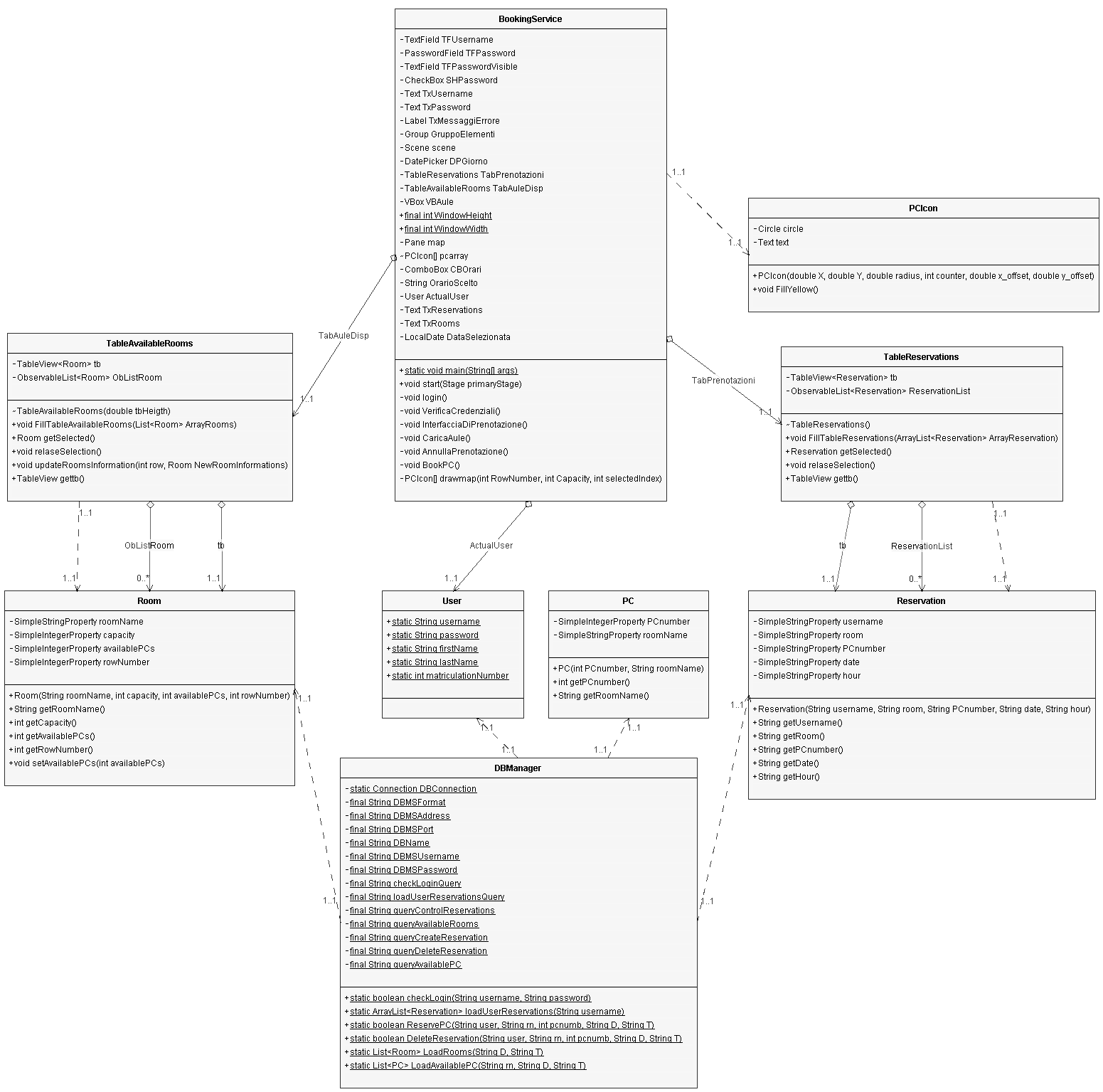
### Relations:

* Occupancy: it connects a specific booking to the reserved pc
* Reservation: it connects a specific booking to the user who made it
* Belonging: it connects each PC to the room in which is collocated

## Tables

* **USER(Username, Password, First Name, Last Name, Matriculation Number)**
  + Username (primary key, string): it’s the university credentials’ username;
  + Password (string): it’s the university credentials’ username;
  + First Name (string): it’s the student’s name;
  + Last Name (string): it’s the student’s surname;
  + Matriculation Number (integer): it’s the student’s university id.
* **BOOKING(Start Time, Date, PC Number, Room Name, Username)**
  + Start Time (primary key, time): it’s the time at which the PC was reserved;
  + Date (primary key, date): it’s the date on which the PC was reserved;
  + Pc Number (primary key, integer): it’s the reserved PC’s id;
  + Room Name (primary key, string): it’s the room in which the reserved PC is located.
  + Username (string): it’s the student’s id;
* **PC(PC Number, Room)**
  + PC Number (primary key, integer): it’s the PC identifier in a certain room;
  + Room (primary key, string): it’s the name of the room in which the PC is located.
* **ROOM(Room Name, Capacity, Rows Number)**
  + Room Name (primary key, string): it’s the name of the room;
  + Capacity (integer): it represents how many PC are located in the room;
  + Rows Number (integer): it represents how many rows the room is divided in.

## UML Diagram



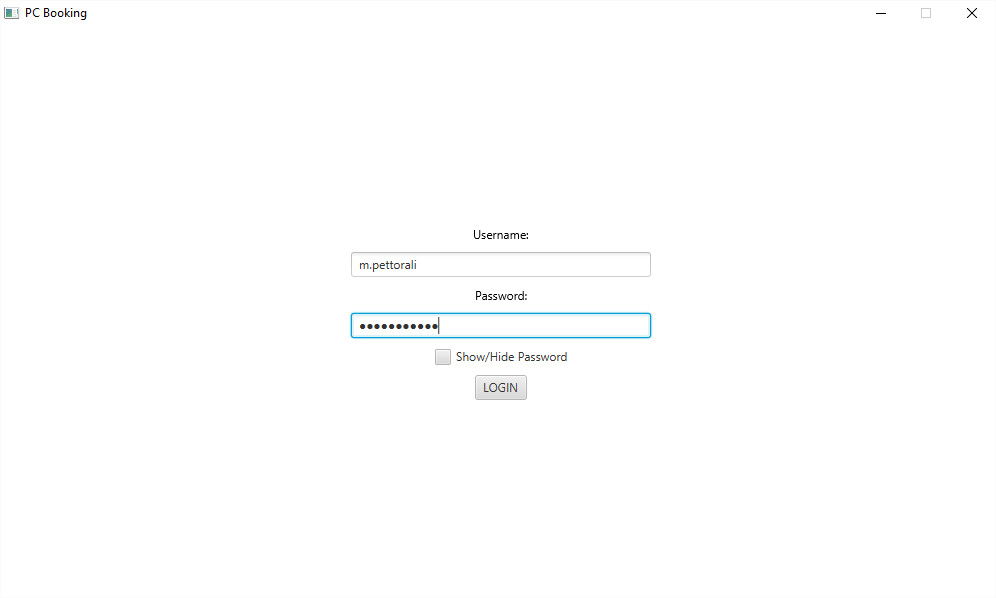
## Classes

|  |  |  |
| --- | --- | --- |
| CLASS NAME | | DESCRIPTION |
| FRONT-END CLASSES | **BookingService** | It is the main class of the application. It displays the GUI and contains the methods for the event handlers |
| **PCIcon** | It represents each pc position in the map of a room |
| **TableAvailableRooms** | It contain the methods to show to the user the status of each room in the campus |
| **TableResevations** | It contain the methods to show each future reservation made by the user |
| BACK-END CLASSES | **Room** | It’s the class that collects all the information about a specific room in the campus |
| **User** | It collects the personal data of a specific user |
| **PC** | It collects the information about a PC in a certain room |
| **Reservation** | It collects the data of a reservation made by a specific user |
| **DBManager** | It’s the class containing the needed methods for making the application relate with the DB |

## User manual

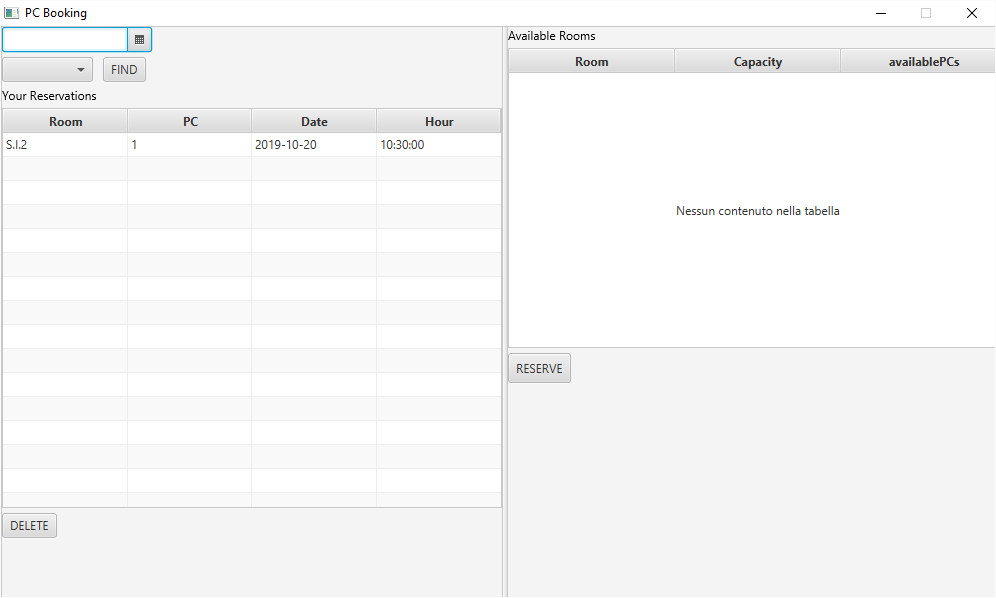
### LOGIN

When the application is launched, the system will display a login form. The user has to put in its username and password and then to click on “LOGIN” button. If desired, the user can also see its password as plain text by clicking on the “Show/Hide Password” radio button.



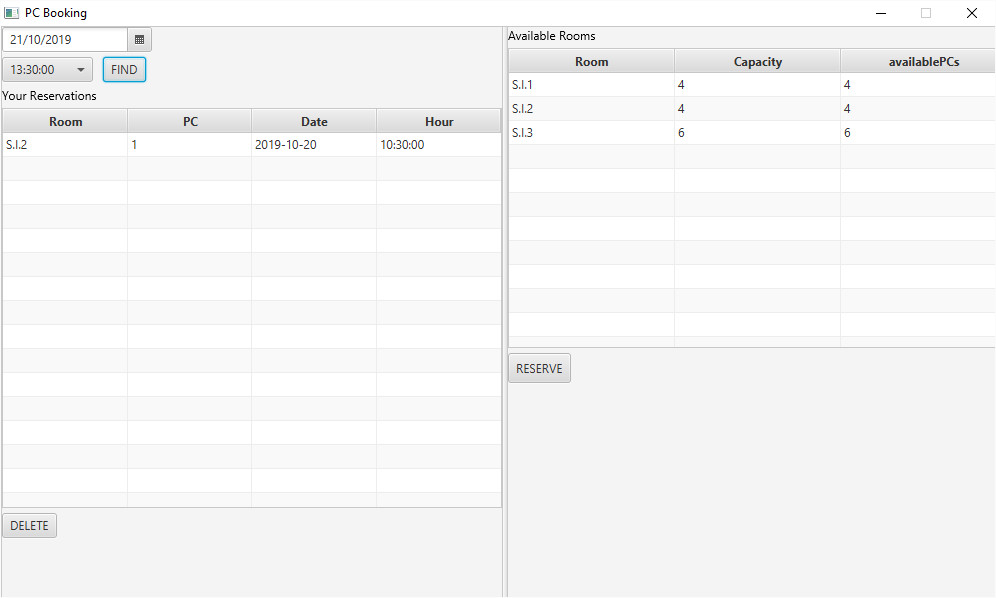
### WORKING WITH THE APPLICATION

After the login, the user is given a list of all its reservation made for future days. The data is displayed in a table, showing the room, the pc number the date and time of the reservation.

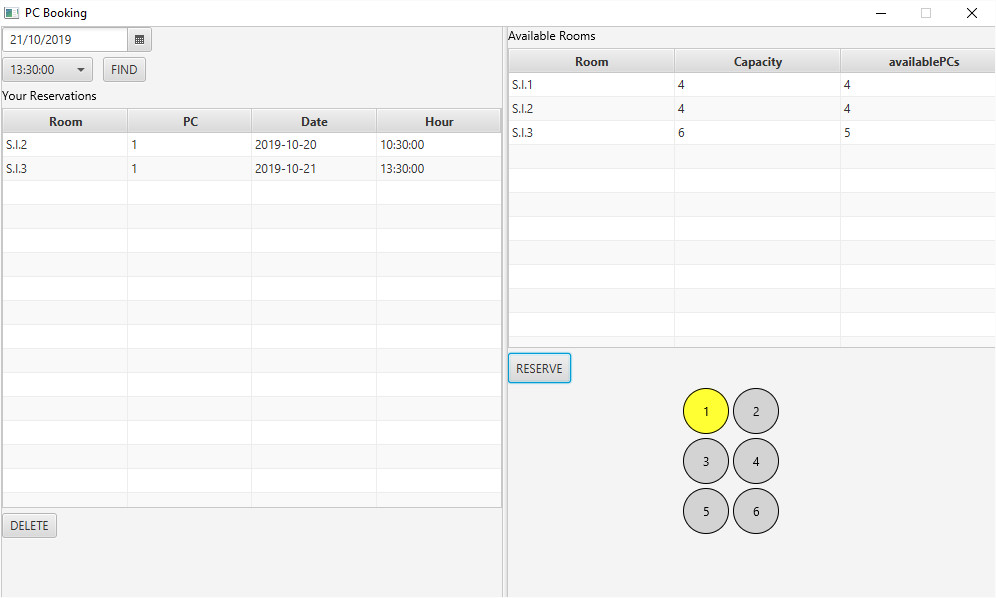


The user has to choose a date and time in the form in the left upper corner and then to click the “FIND” button to watch the status of the rooms in the campus at the specified moment.

The table “Available Rooms” will be filled with a row for each room in the db, indicating some information about the room itself, like the capacity and the number of the available PCs in the room



The user has now to click on a room from the table and click on the “RESERVE” button, to create a new reservation in his name. A map of the room will be displayed, highlighting the position of the PC assigned to the user. Moreover, the tables will be updated: “Your Reservations” table will display another row including the information about the reservation, while “Available Rooms” table will show the updated value of the available PCs in the rooms.



### REMOVING A RESERVATION

Any time the user wants to delete a reservation, it simply has to click on the reservation that it wants to delete from “Your Reservations” table, then just to click on the “DELETE” button; the tables are updated accordingly.

