



## Table of contents

[Table of contents 1](#_heading=h.gjdgxs)

[A.](#_heading=h.30j0zll) Presentation: 2

[1.](#_heading=h.1fob9te) Introduction : 2

[2.](#_heading=h.3znysh7) TaskDistribution : 2

[B.](#_heading=h.2et92p0) Diagrams : 2

[1.](#_heading=h.tyjcwt) Entity-Relationship Diagram: 3



[2.](#_heading=h.3dy6vkm) Use Case Diagram: 8

[C.](#_heading=h.1t3h5sf) Presentation Of The Interfaces: 9

[1.](#_heading=h.4d34og8) Login: 9

[2.](#_heading=h.2s8eyo1) Sign-up: 10

[3.](#_heading=h.17dp8vu) Forgot Password: 12

[4.](#_heading=h.3rdcrjn) Validation of the employees: 13

[5.](#_heading=h.26in1rg) Booking: 16

[6.](#_heading=h.lnxbz9) Room info: 17

[7.](#_heading=h.35nkun2) Customer info: 21

[8.](#_heading=h.1ksv4uv) Statistics: 23

[i.](#_heading=h.44sinio) Occupancy Status: 23

[ii.](#_heading=h.2jxsxqh) The most reserved rooms: 24

[iii.](#_heading=h.z337ya) Top 3 Clients: 25

[iv.](#_heading=h.3j2qqm3) Monthly Revenue: 26

[9.](#_heading=h.1y810tw) Details: 29

[D.](#_heading=h.4i7ojhp) Conclusion: 29

# Presentation:

## Introduction :

This Hotel Management System is an app that consist in helping the employee apply different operations on the data concerning the hotel. The minimalist design makes easy and efficient to work with. The different interfaces give the possibility to the employee to manage the reservations, check-ins, and check-outs while navigating smoothly. It can also keep track of the rooms, customers, and employee’s info. The system’s ergonomics boosts the performance of the user’s work.

## Task Distribution :

In Collaboration:

* The conception of this project was made in collaboration from the two parts.

Mariam Ben Abdallah:

* The report including: A PowerPoint file, the diagrams, the design.
* The MySQL queries and the stored procedure.

Yasmine Ben Slimane:

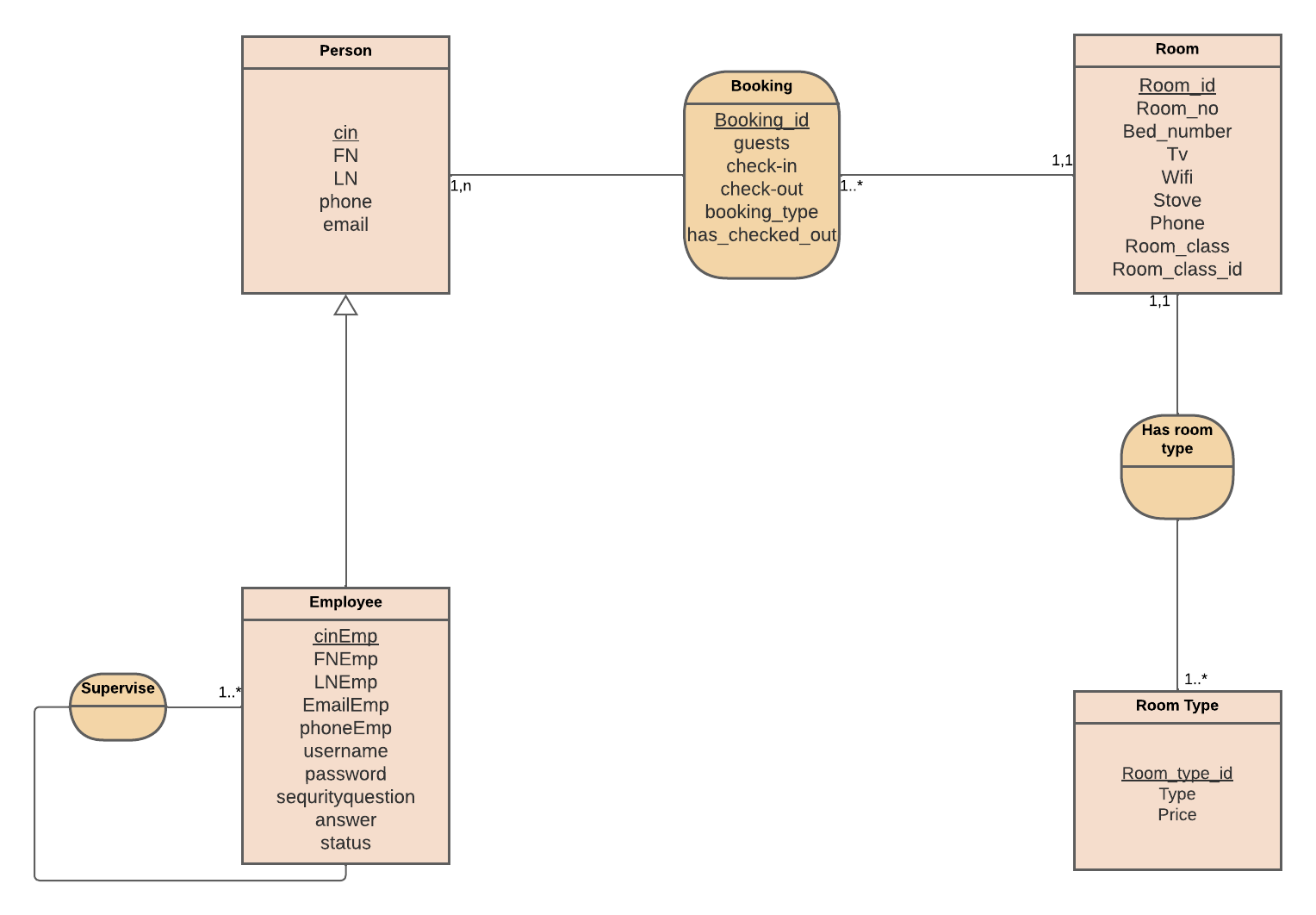
* The back-end coding (Classes, DB\_Classes) and the design of the interfaces in Java (UI) and the input and error control
* The connection between the database and the app

# Diagrams:

## Entity-Relationship Diagram:



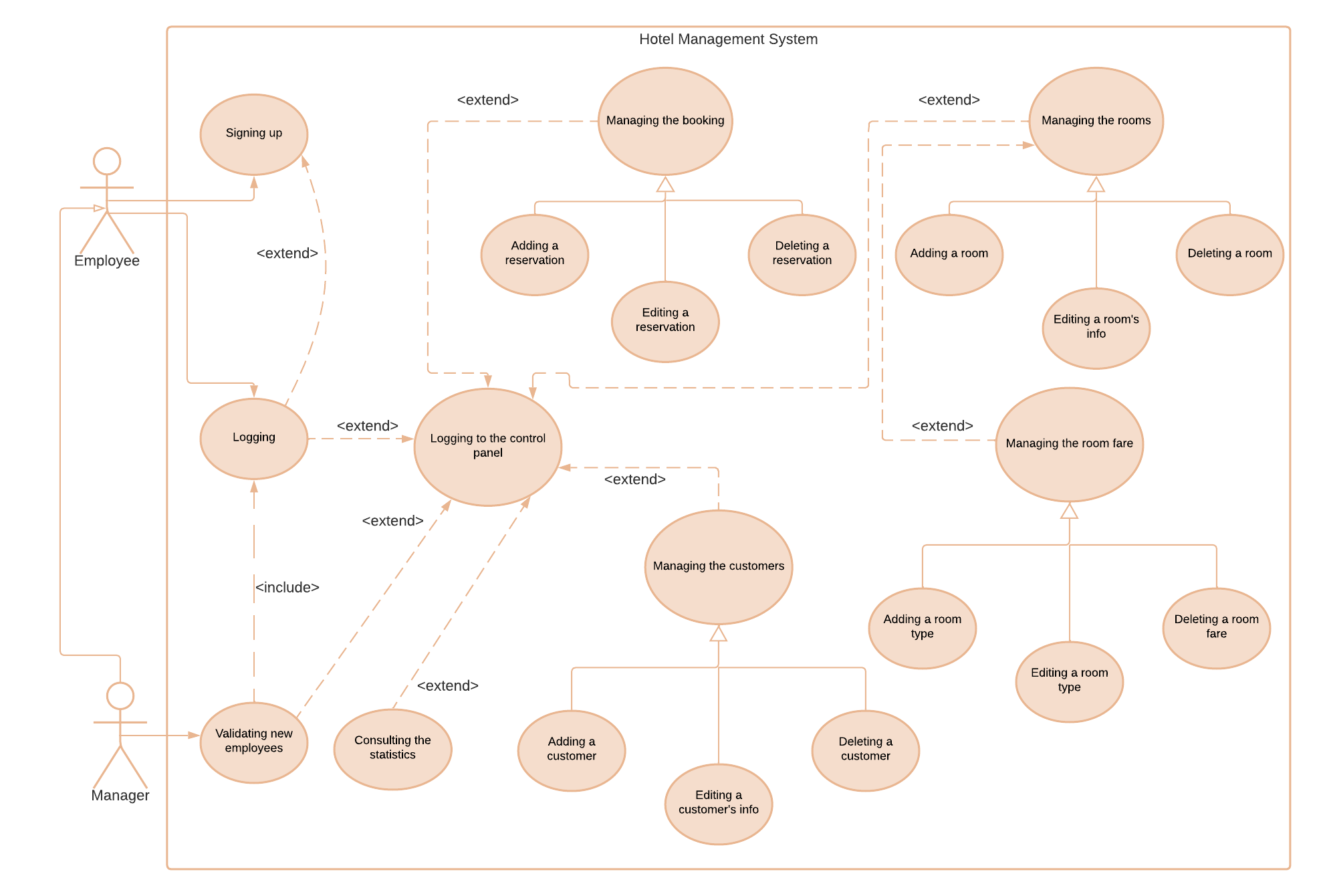




The ER diagram represents a hotel management system, which includes several entities.

* The first entity is "Person," which has attributes such as CIN, first name, last name, phone number, and email. The second entity is "Employee," which has similar attributes as the "Person" entity, along with additional attributes such as username, password, security question, answer, and status. It is worth noting that **one employee is also a person.**
  + This is translated in the java classes by the fact that the Employee class inherits (extends) from the Person class.
* **one employee (manager) supervises all the other employees**. This is indicated by the "Employee" entity's relationship with the "Employee" entity, where many employees are supervised by one employee.
  + This is translated in the java classes by the fact that the Employee class inherits has a static attribute called manager (of type Employee)
* The third entity is "Room," which has attributes such as room ID, room number, bed number, TV, Wi-Fi, stove, phone, and room class. The fourth entity is "Room Type," which has attributes such as room type ID, type, and price. **Each room has one room type, and one room type can belong to many rooms.** This relationship is indicated by the "Room" entity's relationship with the "Room Type" entity, where one room is associated with one room type.
  + This is translated in the java classes by the fact that the Room class has an attribute room \_class (of type RoomType)
* Finally, there is an association called "Booking" that represents the booking information for each room. This association has attributes such as booking ID, guests, check-in date, check-out date, booking type, and checked-out status**. One person can book many rooms, and each booked room belongs to one booking**. The "Booking" association is represented as a table in MySQL.
  + This is translated in the java classes by the fact that the Booking class has an attribute called client (of type Person) and an attribute called rooms (of type ArrayList<Room>)
* In summary, the ER diagram represents a hotel management system that includes several entities and their relationships. The entities include "Person," "Employee," "Room," and "Room Type," and the associations include "Booking." This ER diagram can be used to design a database schema for a hotel management system.

## Use Case Diagram:

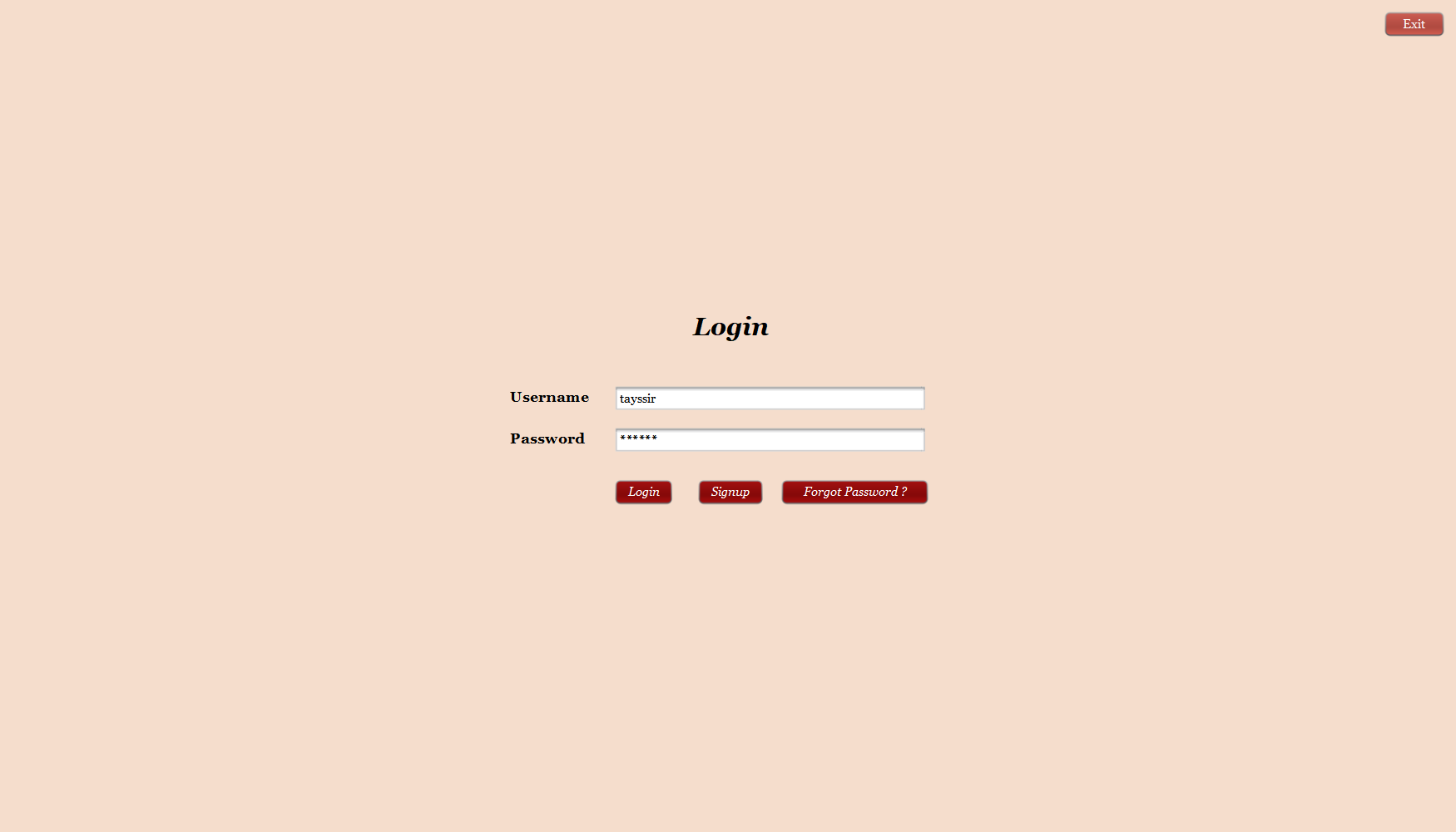


Our system contains two actors that are employee and manager, as you can see the manager is also an employee and can accomplish all the user's case, add to that that he is the only one that can validate the new employees. For that he will need to login and validate the employees, at that point he will have the choice to log to the control panel. The normal employee can sign up and optionally login after the validation of the admin. Like the manager he has the possible to navigate through the control panel. This system allows the employee to manage the customers, the reservations, the rooms, and the room fare. By managing he can add, edit, or delete objects.

Obviously prior to logging to the Control panel the employee is obliged to login and that explains the include between the two use cases.

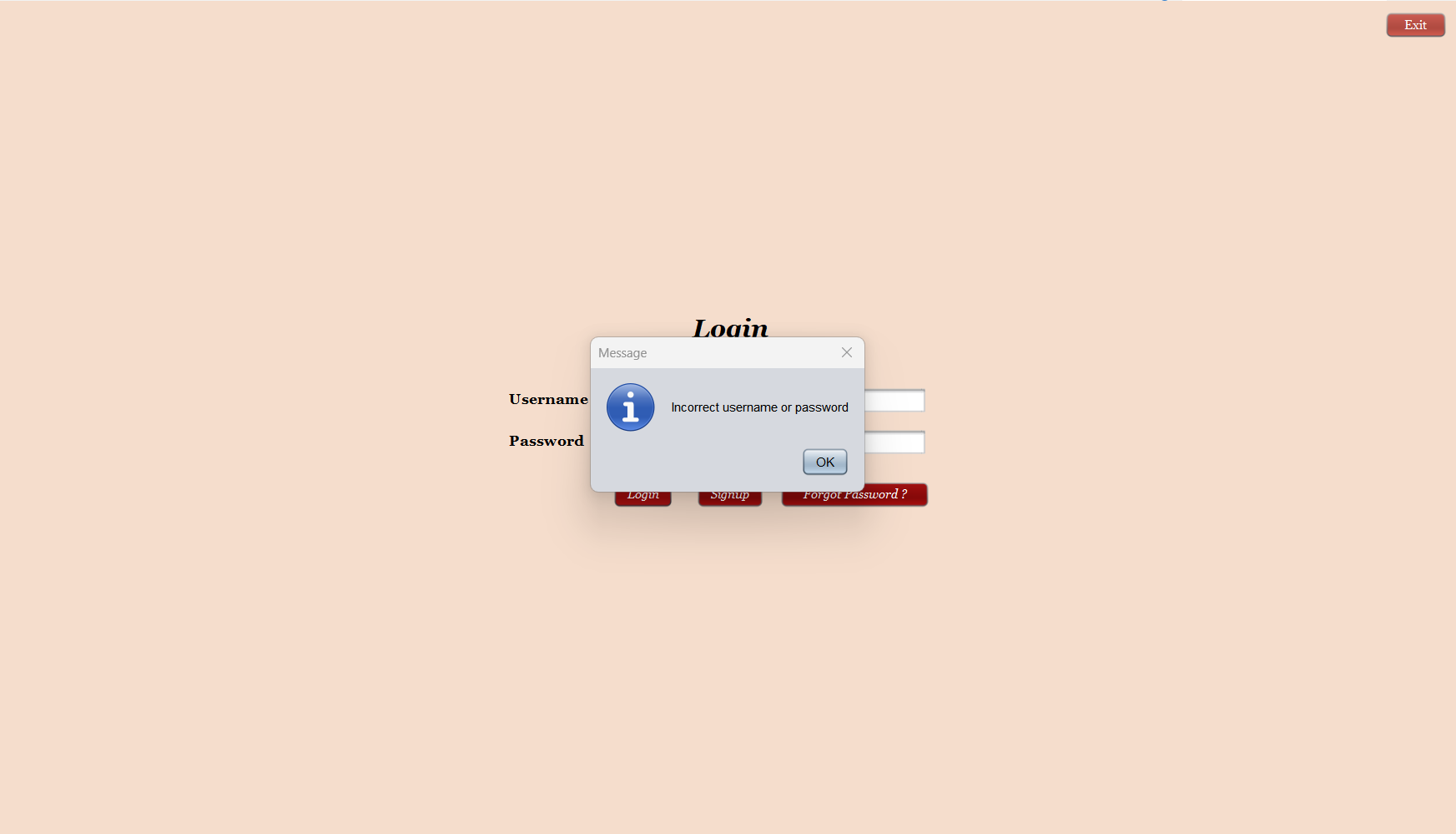
# Presentation Of The Interfaces:

## Login:



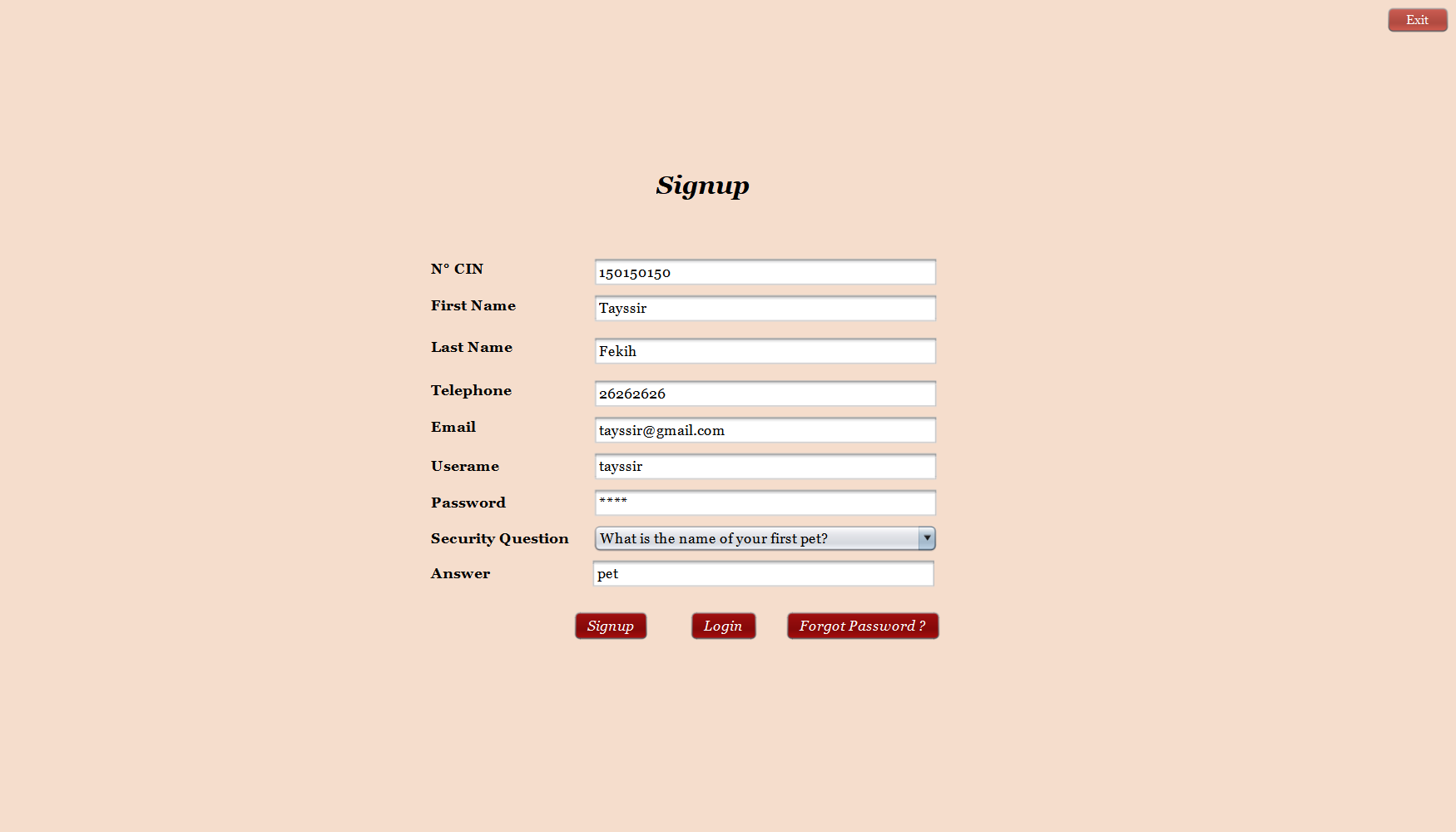
This is the first step of using the Hotel Management System, the authentication interface requires a username and password of an employee to be able to reach the control panel.

If you enter information that does not exist in the database and try to login (the previous query returns nothing) a message will pop out with an error:



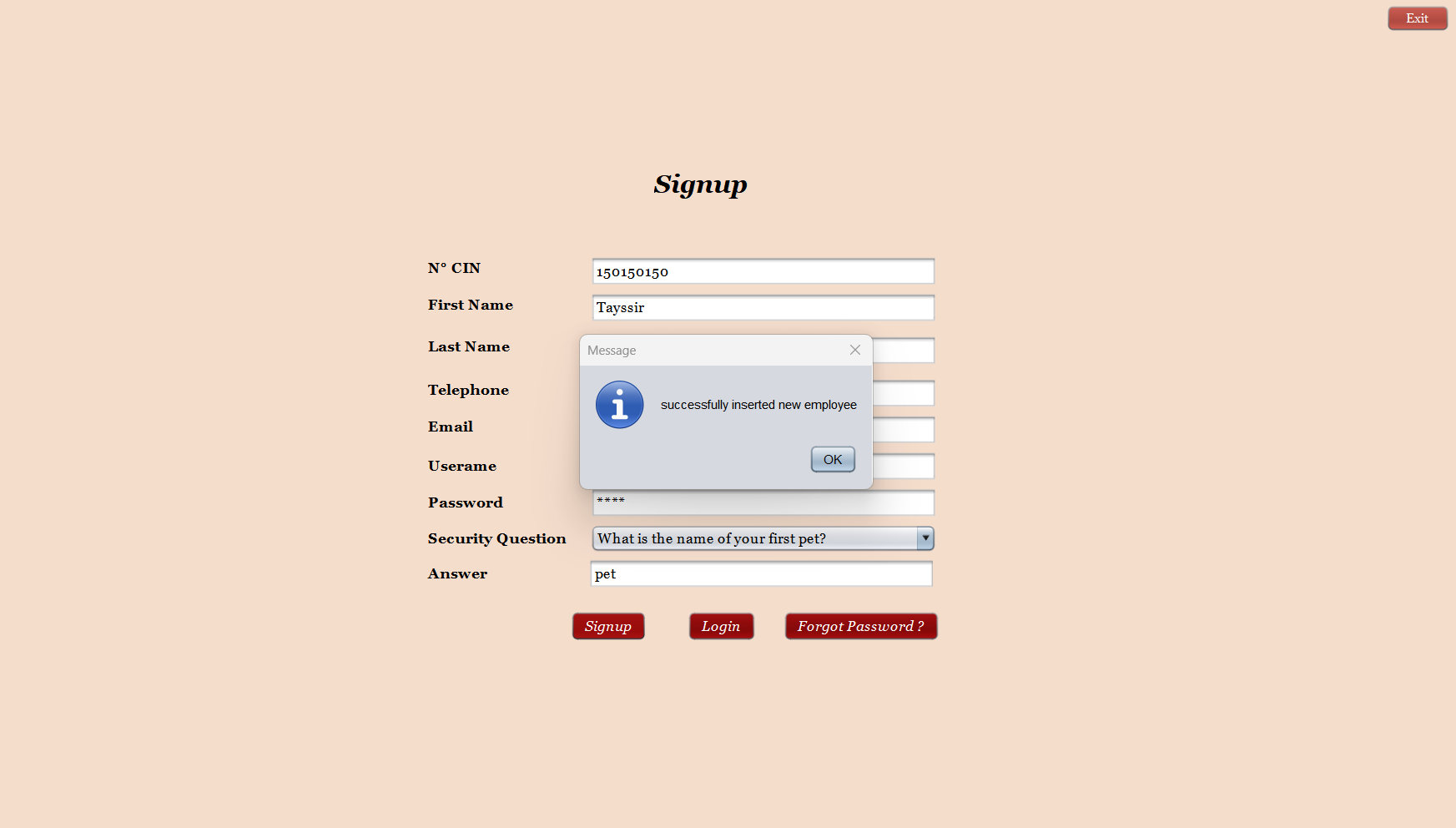
## Sign-up:

If you click on the sign-up button another interface will open and you can create a new employee account.



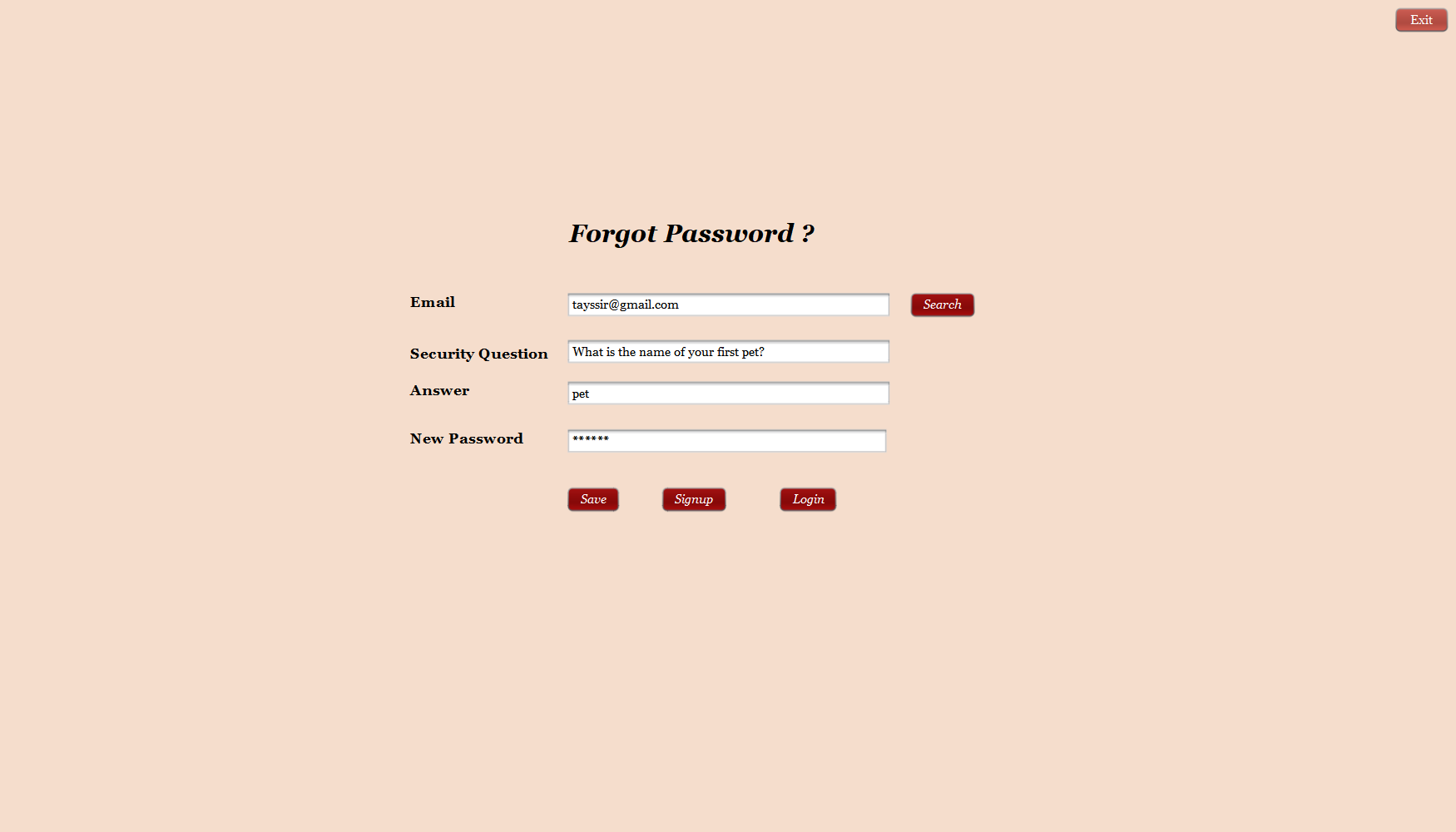
While you enter the information needed for the sign-up, you will notice that the numeric text fields accept digits only, as well as the email text field which requires a specific verification for the existence of the ‘@’, ‘.com’ and the domain of the address.

This form also requires a security question in case the employee forgets their password.



As soon as you hit the signup button a message will pop up with the confirmation of the added employee.

## Forgot Password:



When the employee tries to login and forgets his password, he gets the chance to change it through this interface:

First, he will be asked to enter his email address that will be verified in two steps:

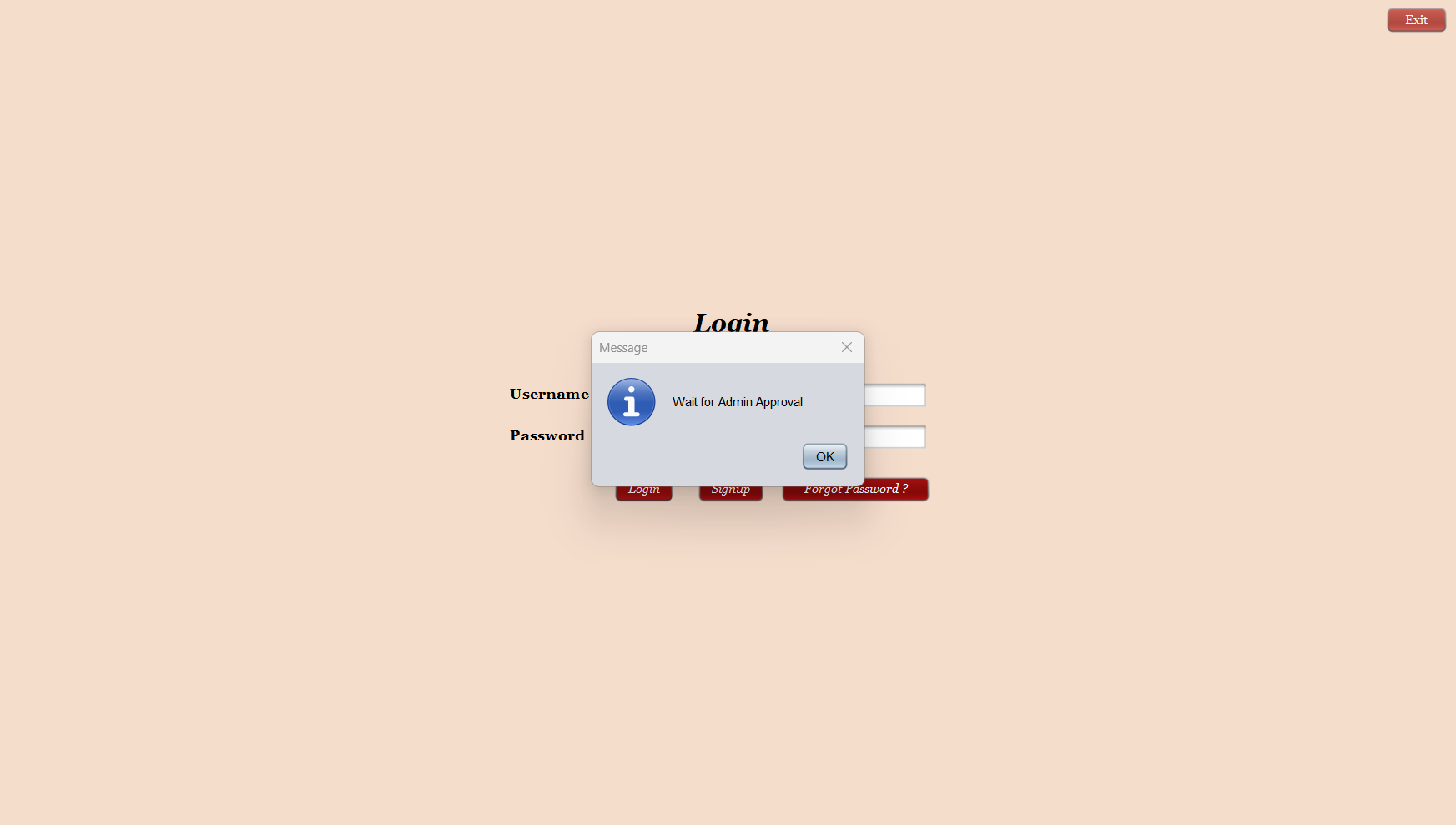
* the typed text in the email field would be verified whether it conforms to a regular expression that checks the existence of the ‘@’, ‘.com’ and the domain of the address, otherwise, an error message will be displayed as soon the focus is lost from the email text field.
  + If the previous condition is verified, the existence of the email in the database is checked (if not, an error message will be displayed)

Then the security question field is auto filled from the database (this field is not editable)

If the employee answers the security question correctly, he will be given a chance to enter a new password, if his answer doesn’t conform to the one in the database, he will get an error message.

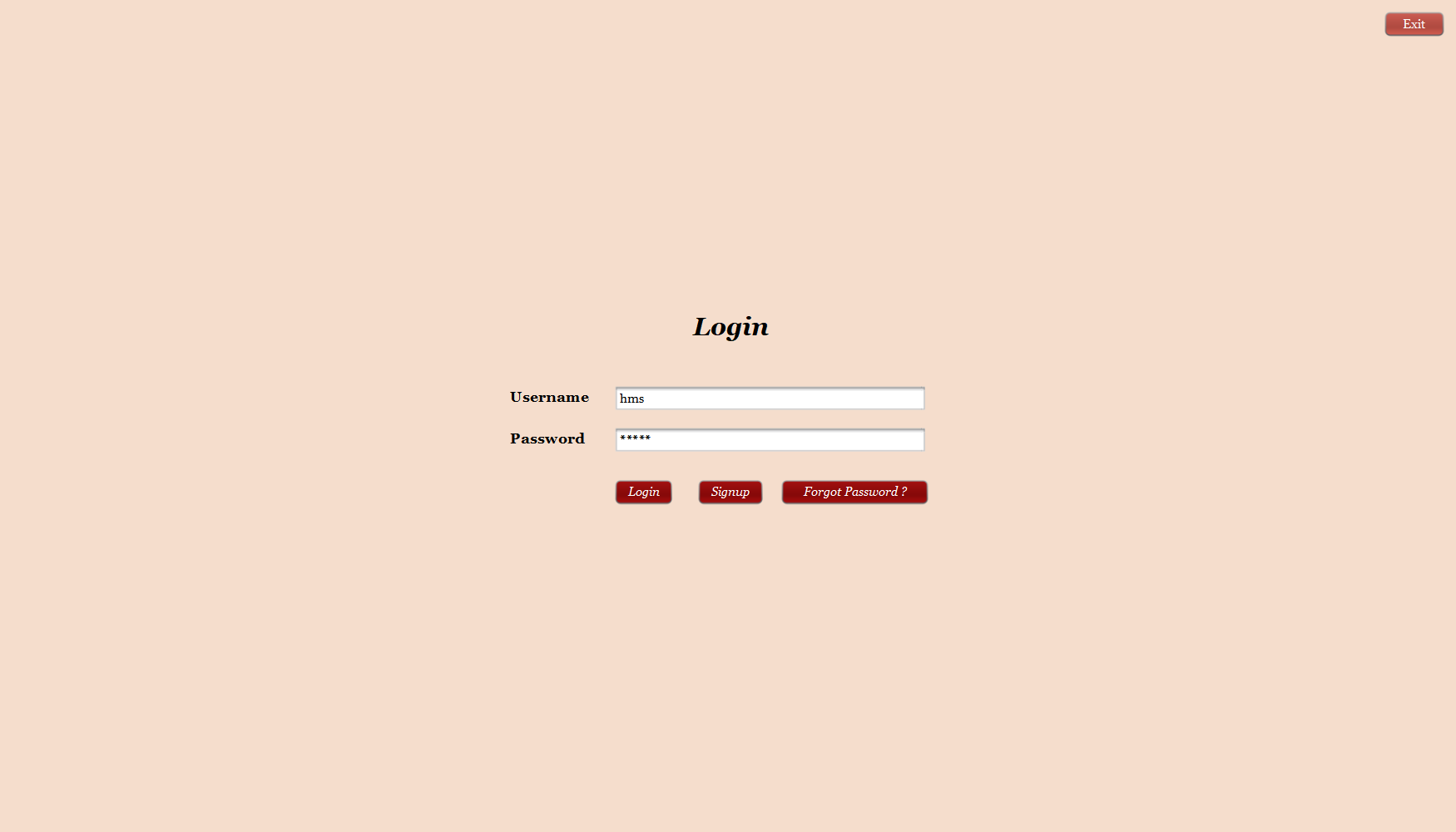
## Validation of the employees:

Let’s say a new employee made an account and tries to login. Even if he enters his credentials correctly, his status is still false this message will pop up:

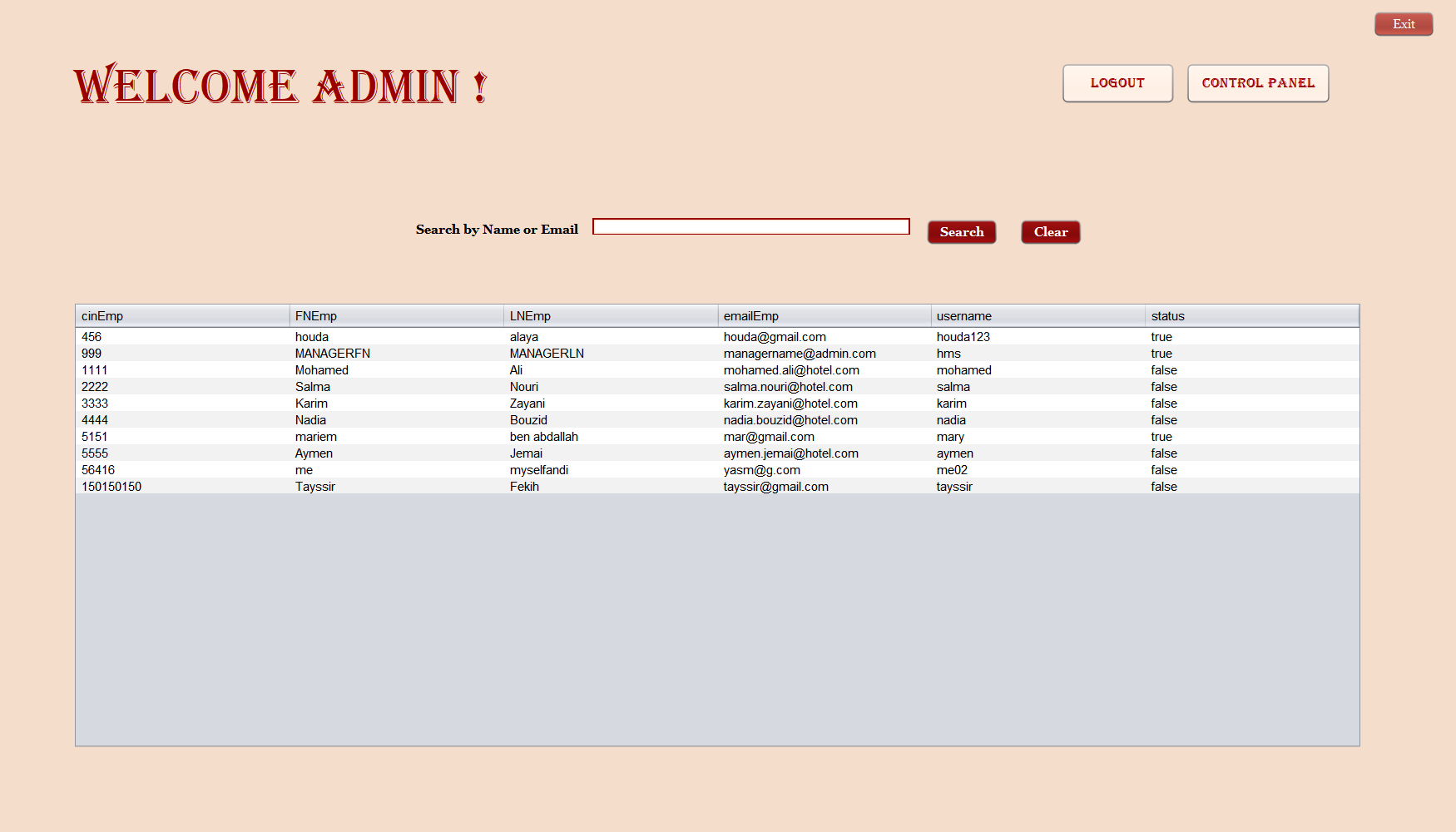


Each employee needs to get validation from the admin who is the manager of the hotel to be able to access the control panel (his status must be true).

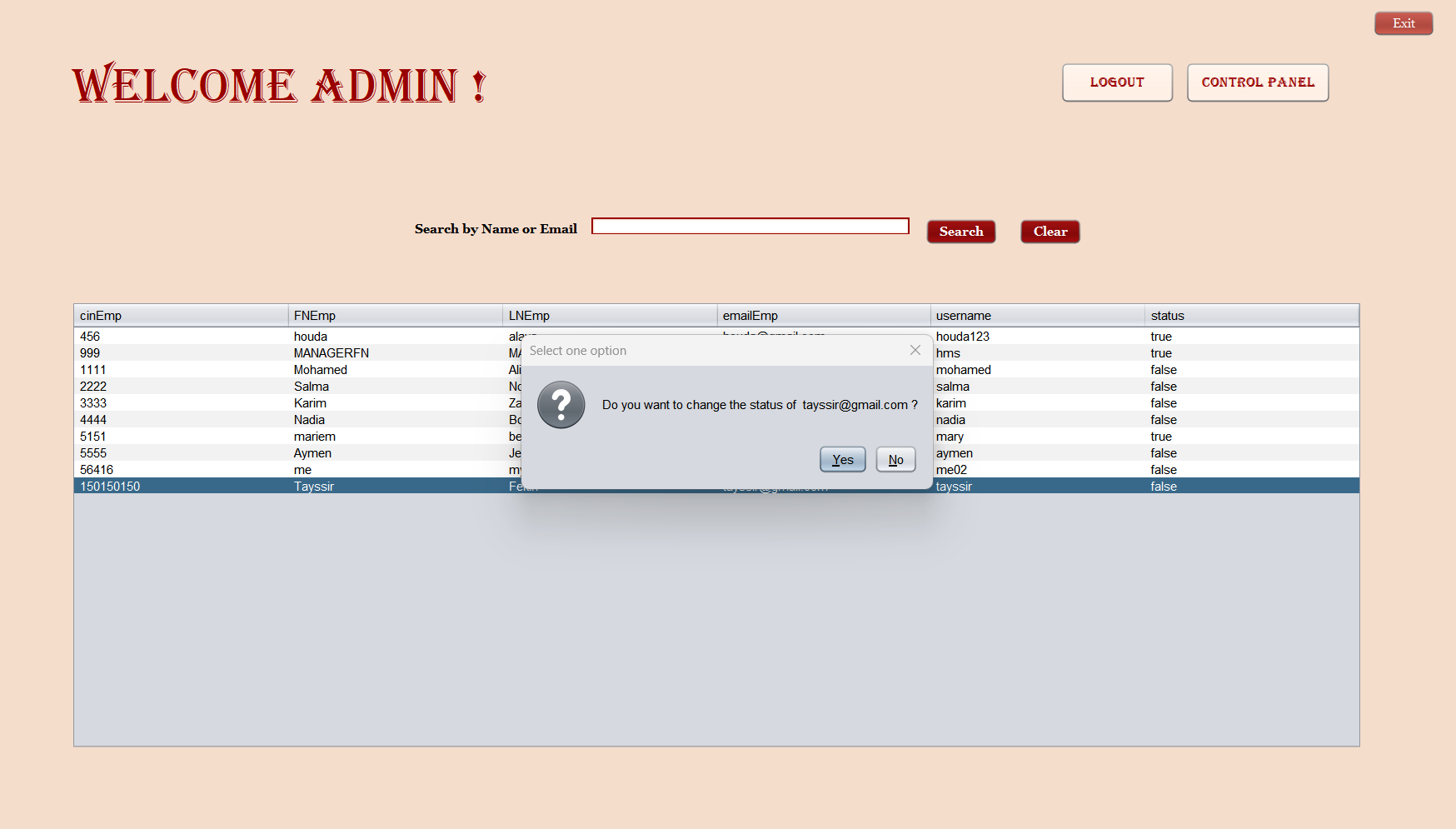
The manager needs to login with his username ***hms*** and password ***admin***:



He will then reach a special interface only accessible for him where he will be able to change the status of a new employee either with selecting one from the list before him (a JTable that extracts all the lines from the employee table in the database) or by searching an employee with a username or an email



When the manager selects the row of the employee that he wants to change his status (the status of all the new employees is “false” by default), this message pops up

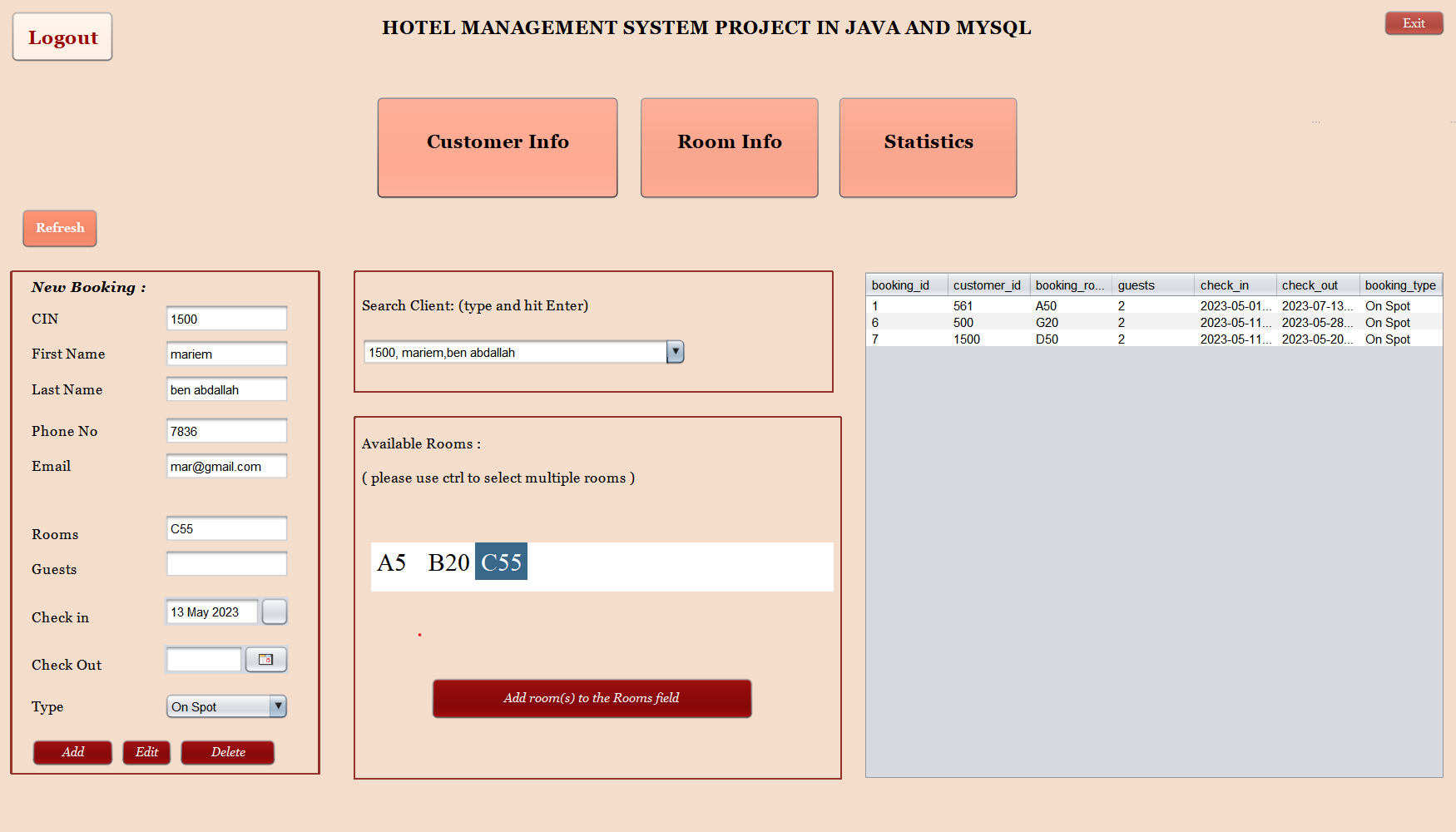


A confirmation will be needed to make sure to accept the employee.

The change in status means that if it is false, it becomes true, and vice versa.

## Booking:





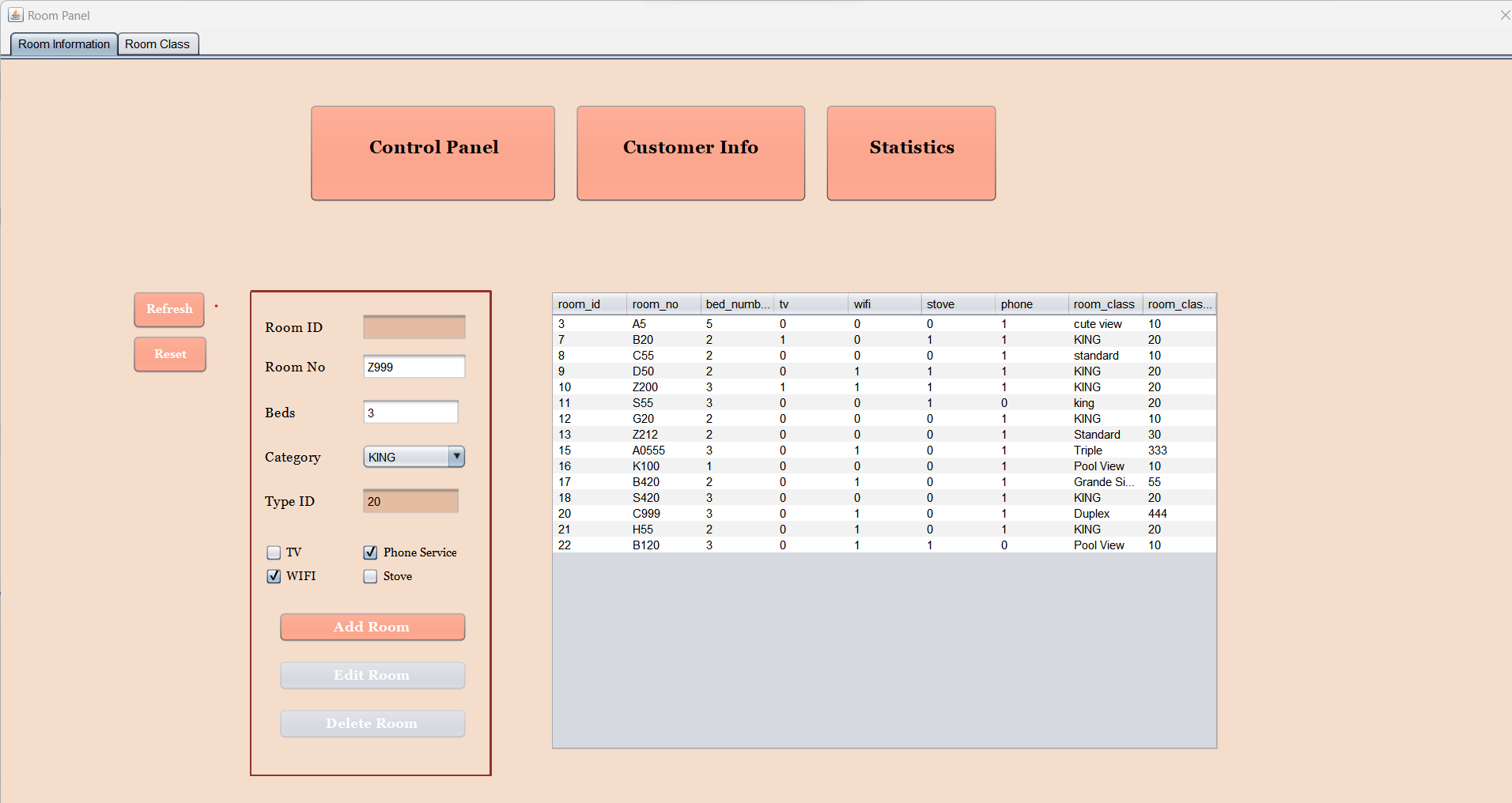
* The employee who can access the control panel will be directed to the booking interface which will provide a form with the required information, a possibility to search a client from a search bar and a JCombo Box and a display of the available rooms in a JList to help with filling the fields of a new booking (multiple rooms can be added at once (using ctrl key) but multiple bookings will be created, each one has one room)
* All the numeric fields only accept digits, as well as the verification of email field.
* The check out field is controlled in case the employee enters a date prior to the check in date.
* This interface also provides a shortcut to an employee to add a new customer in the database through the fields of the customer info, in the booking form: if the typed CIN does not exist in the person table in the database, an add customer query will be executed in the background.

A screenshot of a computer

Description automatically generated

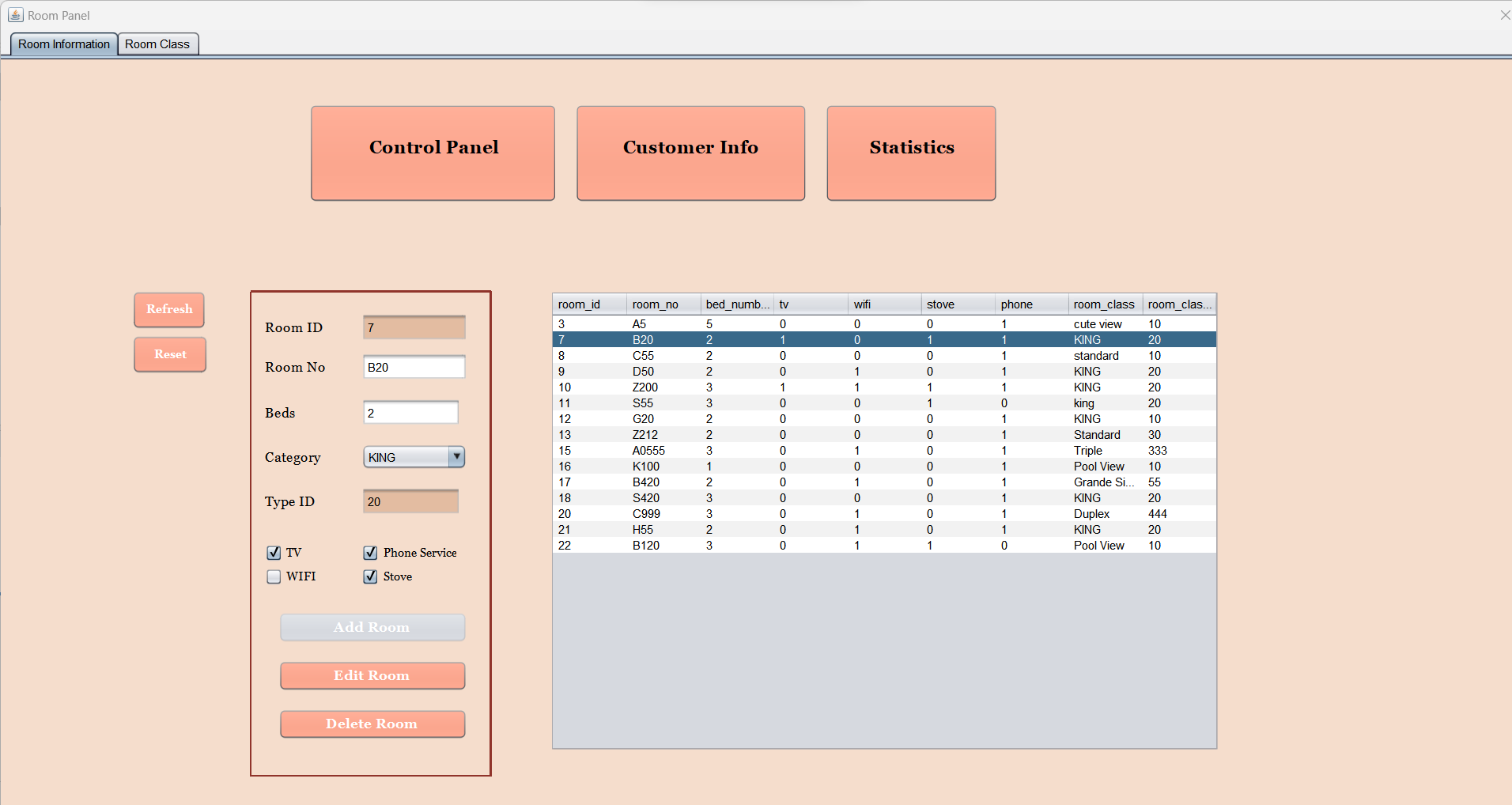
As soon as the employee clicks on a booking from the list its information will populate the text fields, the JDate chooser, and the possibility to edit and delete a booking’s info, will be available to the employee.

## Room info:



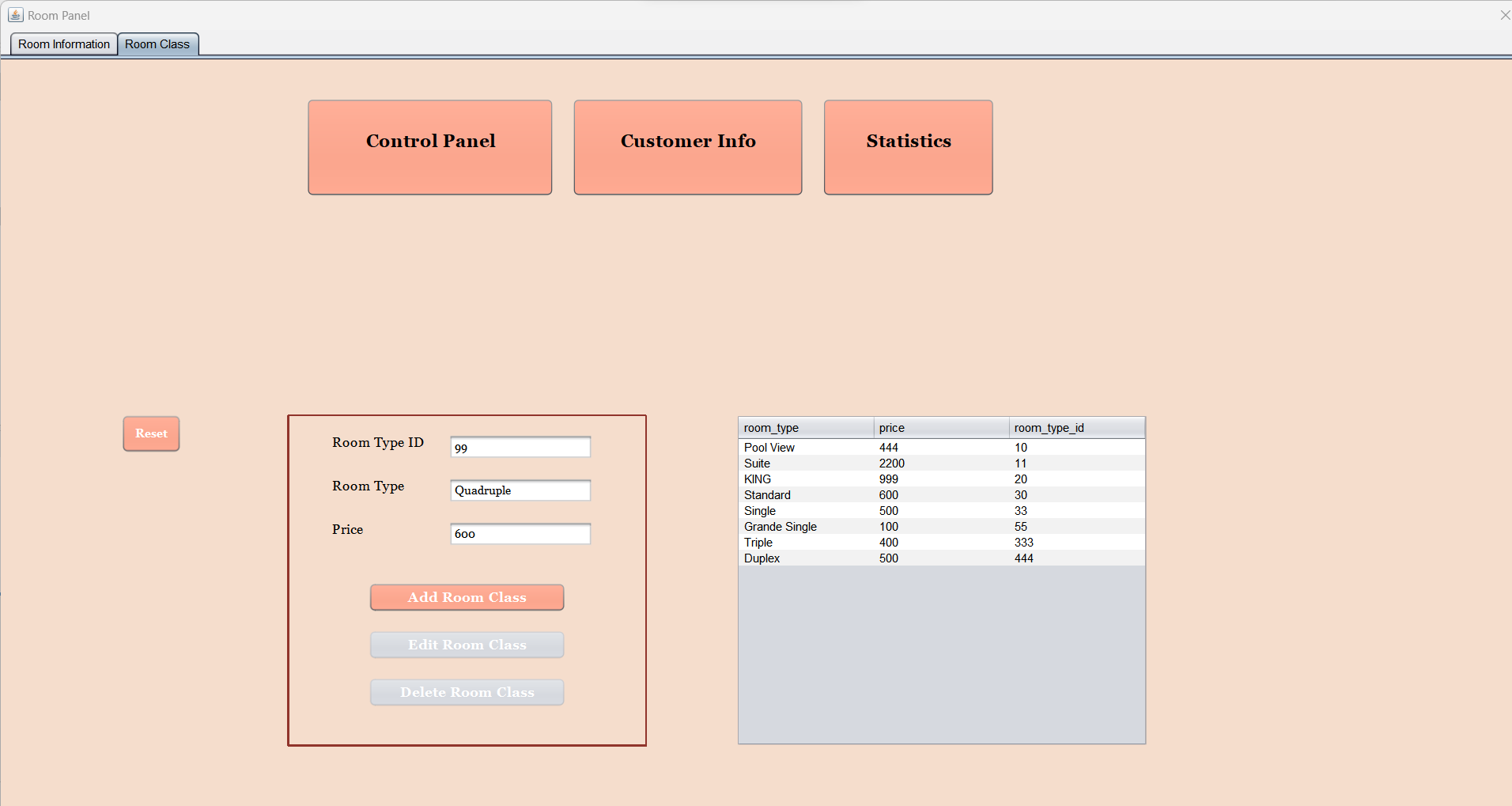
This interface will help the employee adding a room and making sure the information was inserted in the database with a display of the rooms of the hotel.

* Is it worth noting that, in this interface, the numeric text fields accept digits only, the Room ID field is non-editable because its autogenerated (auto-increment feature of MySQL in table creation queries)
* And also, the category (room type) JCombo Box is generated with a list that takes the result of the query to select all the room types from the table roomType
* And to improve the user experience and make our application more ergonomic: the type ID field is auto filled when it gets clicked on
* And finally, in the interface the JCheck Boxes ‘states (checked or unchecked) are translated to Boolean in order to create a new room object that will be used in the insert query of a new room

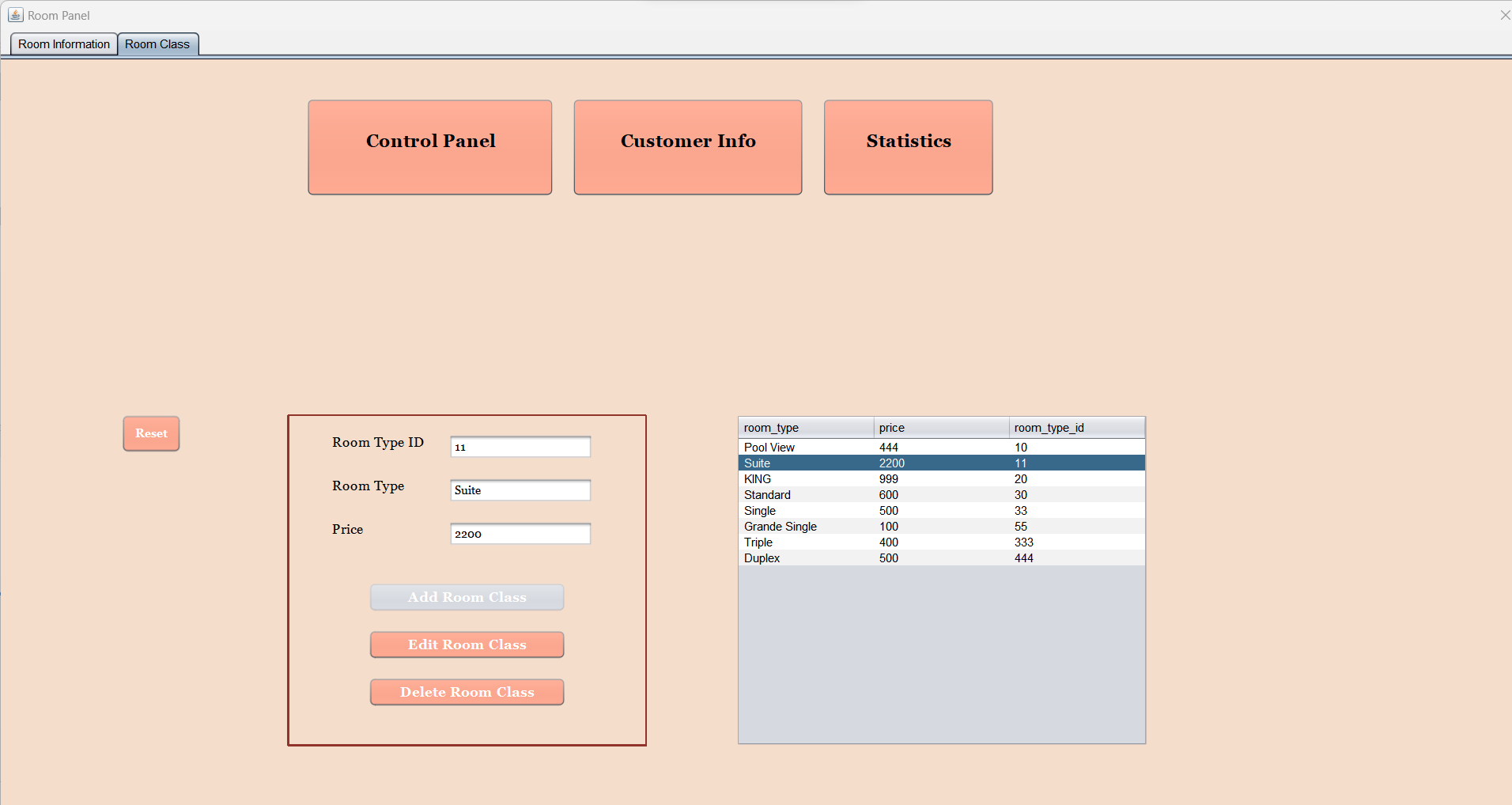


As soon as the employee clicks on a room from the list its information will populate the texts fields, the JCombo Box and the JCheck Boxes, and the possibility to edit and delete a room info, will be available to the employee.

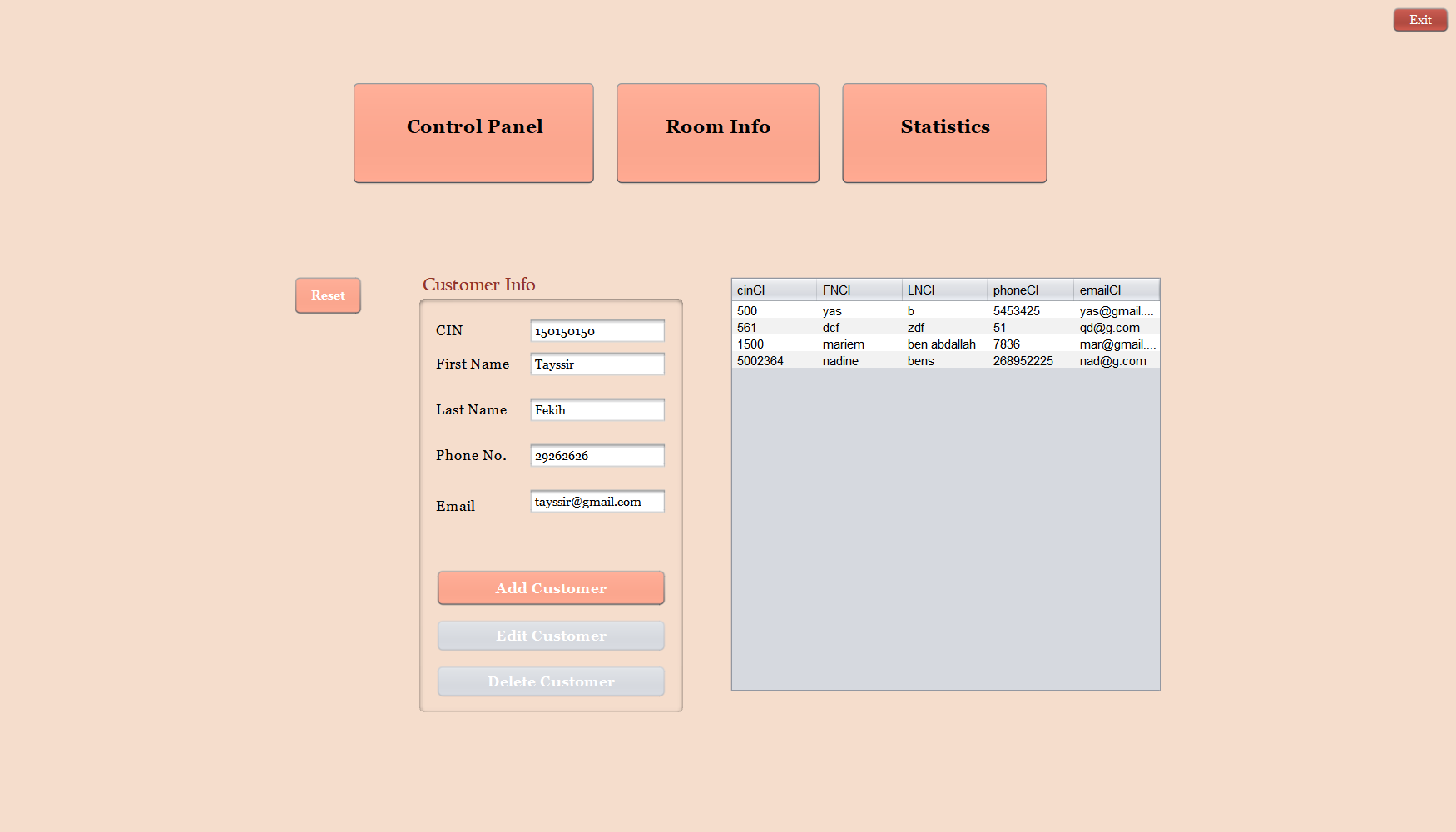
**Room Fare:**



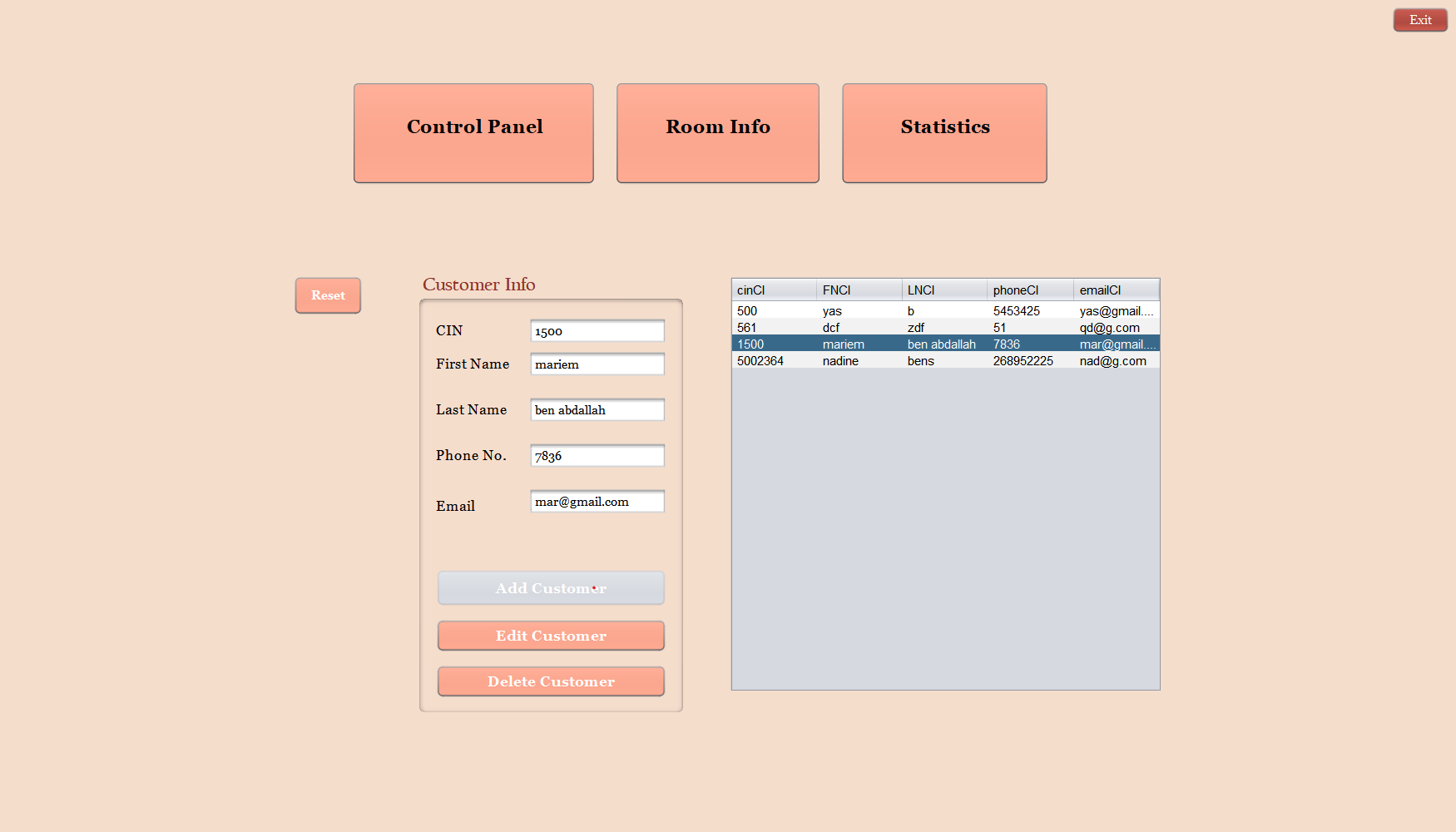
As well as inserting room information the system allows you to insert, edit and delete a selected type of room.



## Customer info:



From the control panel we can access the customer info interface which will insert a customer the same way we could insert a room. When you enter the information needed for a client, you will notice that the numeric text fields accept digits only, as well as the email text field which requires a specific verification for the existence of the ‘@’, ‘.com’ and the domain of the address.



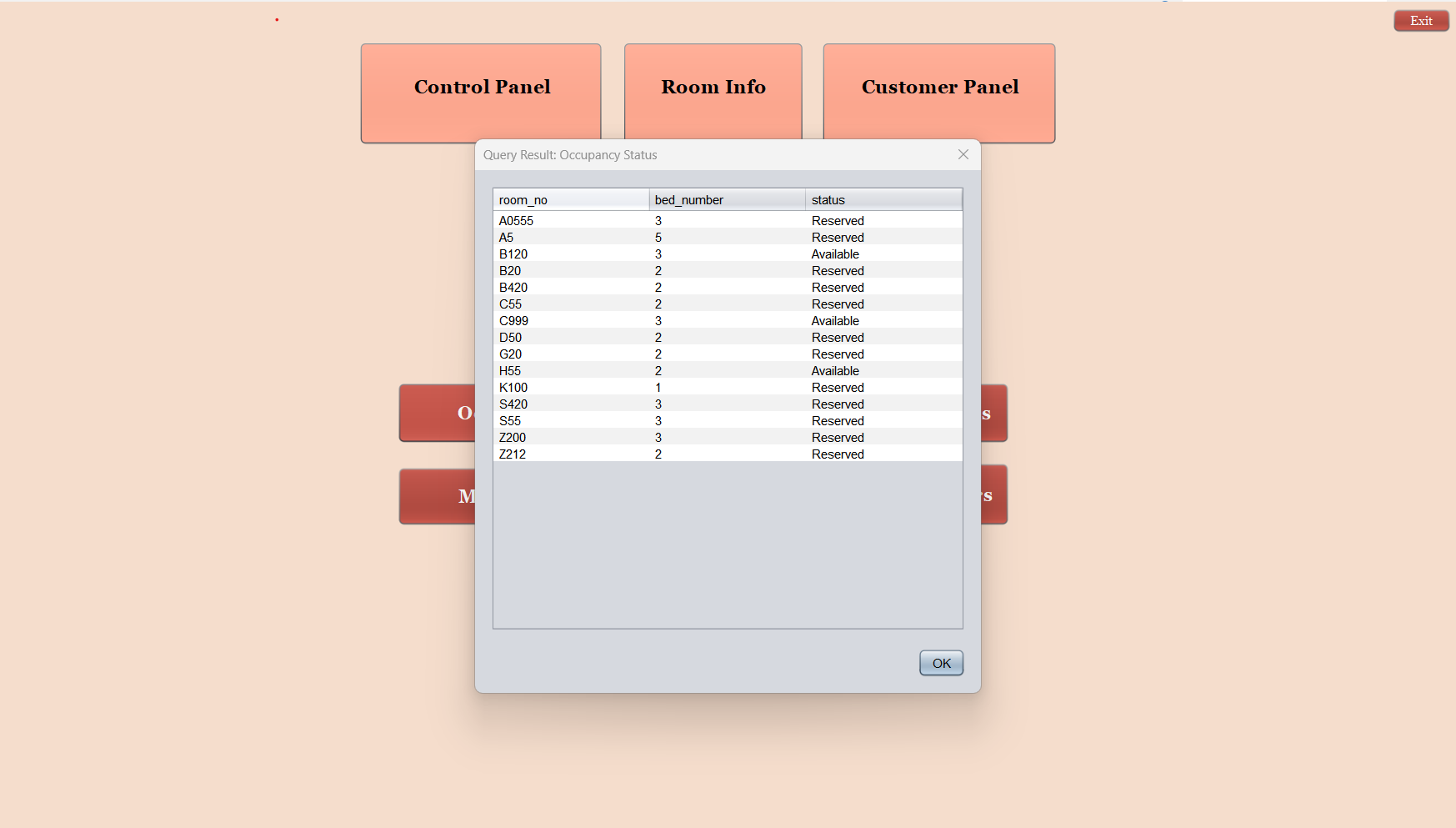
The edit and delete buttons will be available after clicking on a customer in the list.

## Statistics:



This panel will show us the statistics of the hotel that were implemented by MySQL queries and stored procedures.

### Occupancy Status:



For the ergonomics of the app, the result of the query will show up in a dialog showing the available rooms as well as the reserved ones.

* This is the query used to show these results:

**"SELECT DISTINCT room.room\_no, room.bed\_number," +**

**" IF(booking.booking\_id IS NOT NULL" +**

**" OR (booking.check\_out > '"+timestamp+"' AND booking.check\_in < '"+timestamp+"' )," +**

**" 'Reserved', 'Available') AS status" +**

**" FROM room " +**

**" LEFT JOIN booking ON room.room\_no = booking.booking\_room" +**

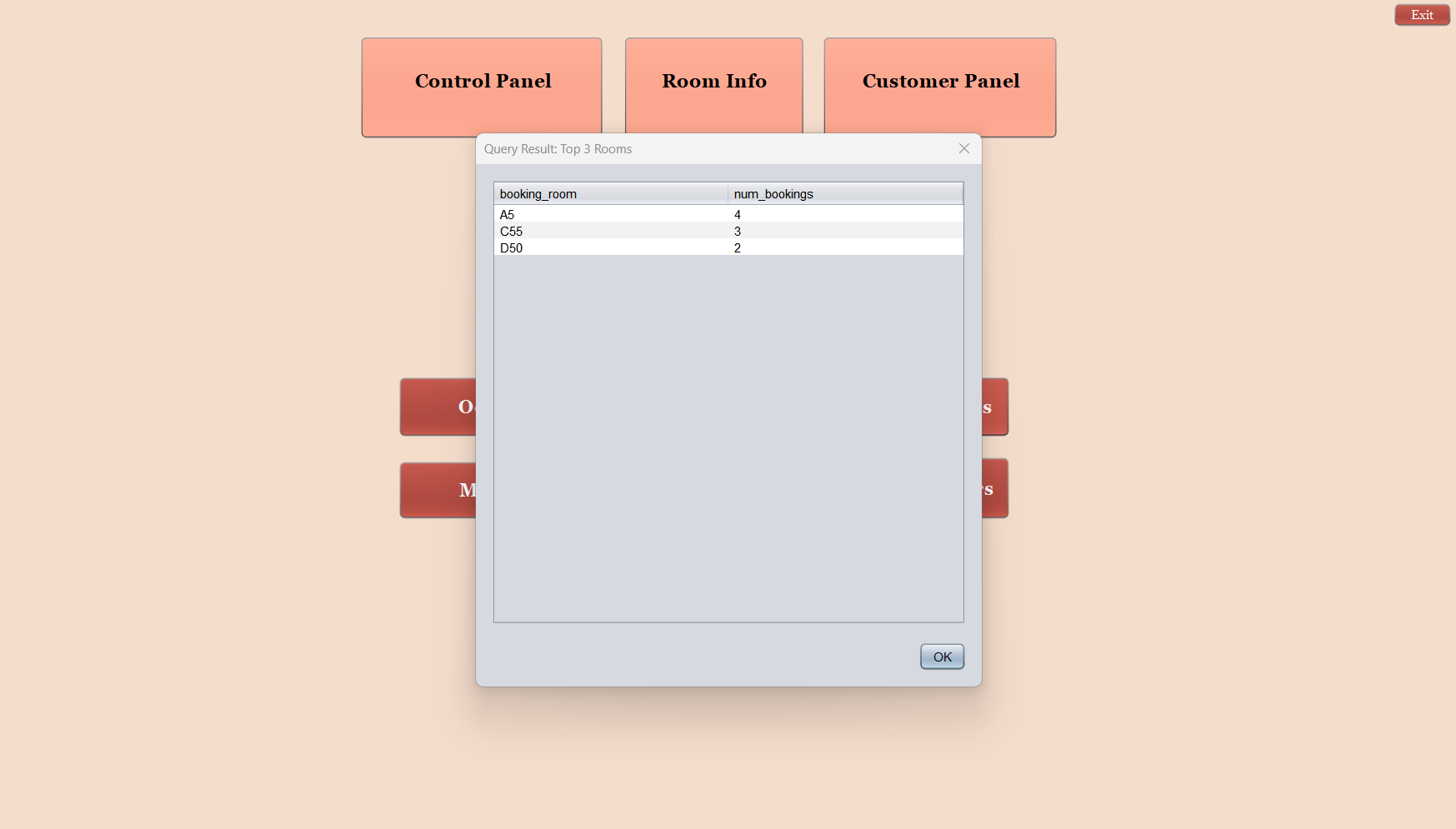
**" ORDER BY room.room\_no;")**

Please note that the variable called timestamp is today’s date in the type “timestamp.”

Date today = new Date();

Timestamp timestamp = new Timestamp(today.getTime());

### The most reserved rooms:



Another dialog will show up with the possibility of consulting the most liked rooms.

* This is the query used to show these results:

**"SELECT booking\_room, COUNT (\*) AS num\_bookings**

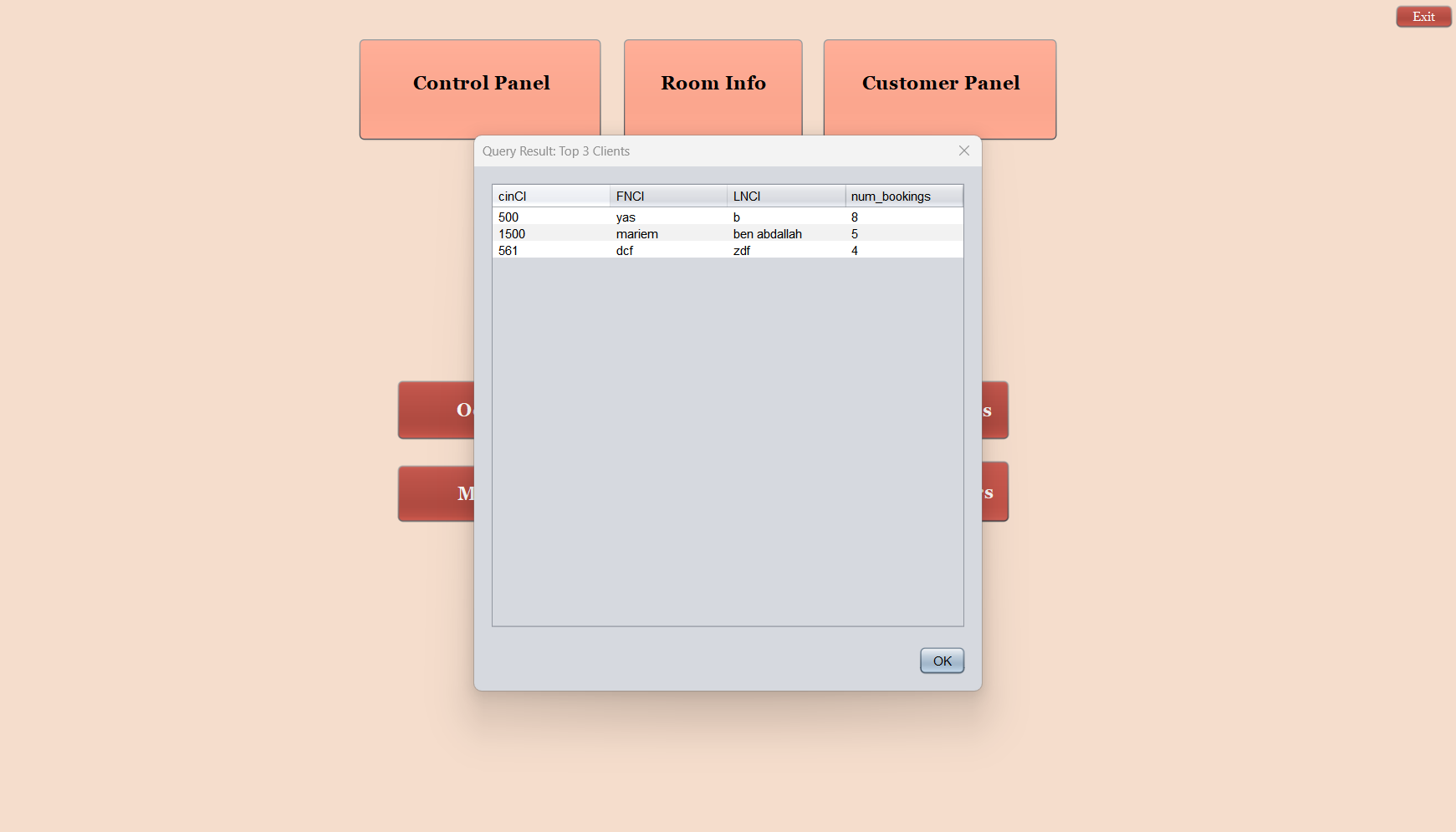
**FROM booking**

**GROUP BY booking\_room**

**ORDER BY num\_bookings DESC**

**LIMIT 3;")**

### Top 3 Clients:



This query’s result showed in a dialog the 3 most loyal customers who booked in the hotel more than the rest of the clients.

* This is the query used to show these results:

**"SELECT person.cin, person.FN, person.LN, COUNT(\*) AS num\_bookings**

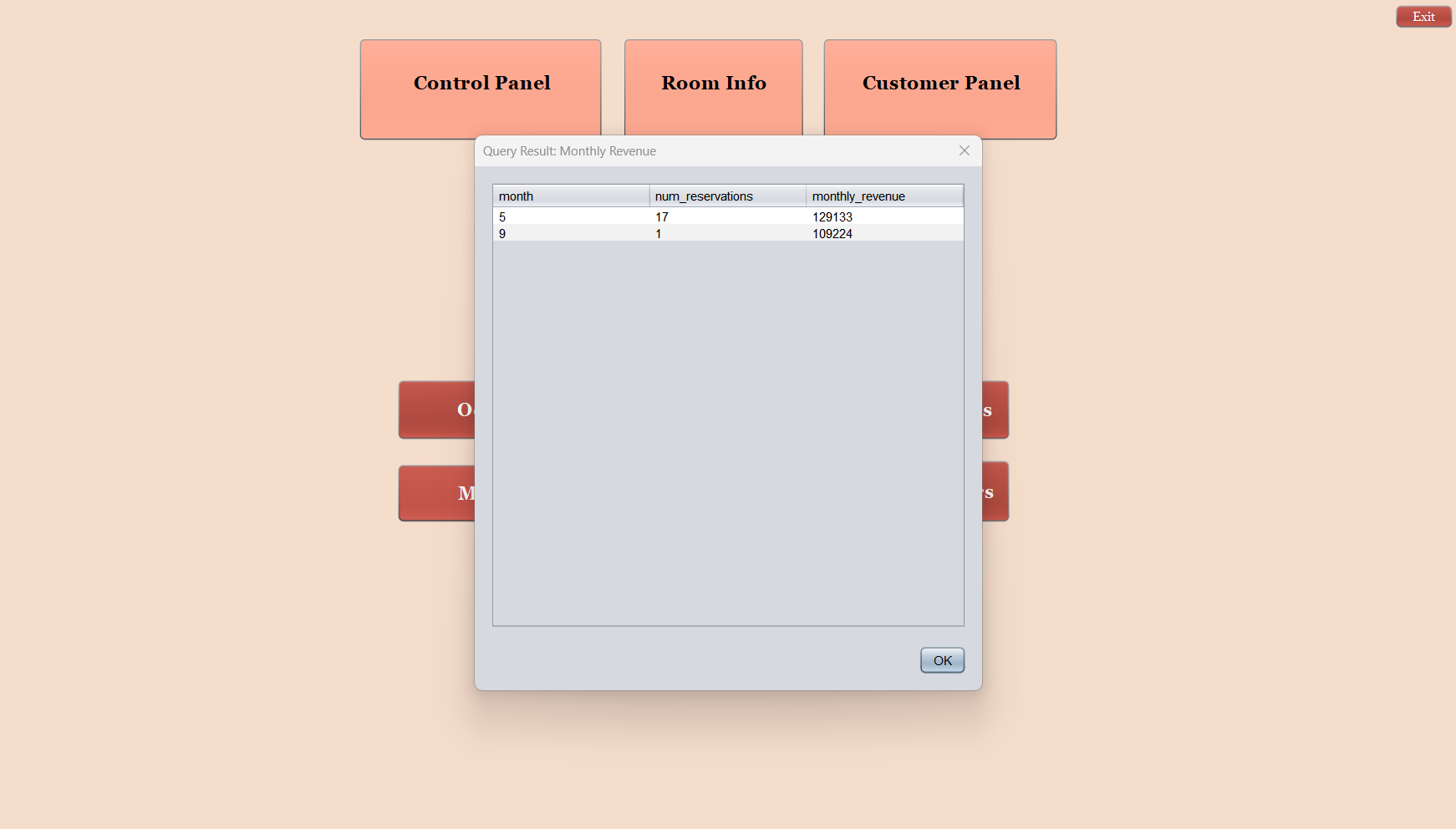
**FROM person JOIN booking ON person.cin = booking.customer\_id**

**GROUP BY person.cin**

**ORDER BY num\_bookings DESC**

**LIMIT 3;")**

### Monthly Revenue:



This stored procedure will show us the revenue of the hotel in each month.

* This is the query used to show these results:

**"call hotel.reservation\_stats()"**

reservation\_stats is a stored procedure that uses cursors to iterate through the result set of the query and row types to store the values of each row:

*Stored procedure in MySQL :*

USE hotel;

DROP PROCEDURE IF EXISTS reservation\_stats;

DELIMITER $$

CREATE PROCEDURE reservation\_stats()

BEGIN

DECLARE done INT DEFAULT FALSE;

DECLARE month\_val INT;

DECLARE num\_reservations\_val INT;

DECLARE monthly\_revenue\_val INT;

DECLARE cur CURSOR FOR

SELECT

MONTH(check\_in) AS month,

COUNT(\*) AS num\_reservations,

SUM(DATEDIFF(check\_out, check\_in) \* roomtype.price) AS monthly\_revenue

FROM booking

JOIN person ON booking.customer\_id = person.cin

JOIN room ON booking.booking\_room = room.room\_no

JOIN roomtype ON room.room\_class\_id = roomtype.room\_type\_id

GROUP BY month

ORDER BY month;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

DROP TABLE IF EXISTS reservation\_stats\_tbl;

CREATE TABLE reservation\_stats\_tbl (

month\_val INT,

num\_reservations\_val INT,

monthly\_revenue\_val INT

);

OPEN cur;

read\_loop: LOOP

FETCH cur INTO month\_val, num\_reservations\_val, monthly\_revenue\_val;

IF done THEN

LEAVE read\_loop;

END IF;

INSERT INTO reservation\_stats\_tbl VALUES (month\_val, num\_reservations\_val, monthly\_revenue\_val);

END LOOP;

CLOSE cur;

SELECT \* FROM reservation\_stats\_tbl;

END$$

DELIMITER ;

## Details:

Every interface has a reset button that can erase all the information entered in the forms, as well as a refresh button that the employee will click on each time there is an insertion in the database.

Every interface has a log out button that allows the employee to log out of his session with a message of confirmation as well as an exit button to exit the app with message of confirmation too.

All the delete queries are preceded by confirmation JDialog messages before they get executed.

# Conclusion:

Our hotel management system is designed to simplify the booking process for the employees. It offers an intuitive interface that allows easy navigation between different options and ensures the security of your data through a password-protected login system.

The app's ergonomic and minimalistic display makes it easy for employees to manage client reservations without much effort. It's user-friendly, even for those with limited technological knowledge.