JavaFX- Hotel reservation prototype

In this project, you’ll use NetBeans and Maven to write a JavaFX application that calculates the number of nights and total price for a reservation based on the arrival and departure dates the user enters. You may notice this assignment is similar to a previous assignment in OOP last year. You may “borrow” some of your logic from that assignment for the calculations. Then you will write code that will insert the data into a MySQL database table. Finally, you will build a second screen that displays all available reservations.

A screenshot of a computer

Description automatically generated

1. Build the main form so it looks similar to the screenshot above. Use Netbeans with SceneBuilder to build your form.
   1. The Number of Nights and Total Price text fields must be disabled and non-editable.
   2. The form must have a title “Reservations”.
   3. The arrival date and departure date text fields must have yellow backgrounds.
2. Add code to get the arrival and departure dates the user enters when the user clicks the **Calculate** button. Then, calculate the number of days between those dates, calculate the total price based on a price per night of $120 and display the results. (HINT: use **LocalDate** for your date variables in the form controller)
3. The exit button will simply exit the application.
4. Add code to the click event of the “Book It!” button so that it will launch the popup form that you will create in the next step.

A screenshot of a computer

Description automatically generated

1. Build the **popup form** so that it looks like the screenshot above. Use a ListView control to display the list of reservations that will come from the database. Don’t worry about populating the ListView yet, just design the form. The “Close Window” button will simply close the current form and bring the user back to the main form.
2. Test the application to be sure it works correctly so far.

Enhance the way the forms work

1. Add an event handler for the **initialize** event of the main form. This event handler should get the current date and three days after the current date and assign these dates to the Arrival Date and Departure Date text boxes as default values. (HINT: you’ll have to implement the initializable interface to use the initialize method)
2. Test the application to be sure that the default dates are displayed correctly and that the totals are calculated correctly.
3. Add the following validation to the form. Display any error message in an alert box.
   1. Both arrival date and departure date must be non-empty and valid dates.
   2. Both dates must be **AFTER** the current date.
   3. The departure date must be **AFTER** the arrival date.

Setup the Database

1. Run the script file \_\_HotelDB\_\_.sql to create the MySQL database you will need for this assignment. It only contains 1 table. You should be able to see the database schema and table in Workbench. Ask if you need help setting this up.

Create 2 classes

1. Create a **HotelReservation** class with 4 properties:

* arrival\_date (LocalDate)
* departure\_date (LocalDate)
* numNights(int)
* price(double)
  1. This class must also contain appropriate getters, setters and constructors. No validation is required in this class because the validation should be handled at the form level.
  2. HotelReservation class must override the **toString**() method to display the properties (this will be used later in the popup form ListView)

1. Create a **HotelDB** class to handle database connections and methods. This class should contain at least these 2 methods. You may decide to create methods to connect and disconnect from the database. No validation is required in these methods either, since everything should already be validated on the form.
   1. **public static boolean AddReservation(HotelReservation hr)** – this method will accept a HotelReservation object as a parameter and insert it into the reservations table of the DB. You must use a prepared statement in this method.
   2. **public static List<HotelReservation> GetReservations()** – will run a select statement to get all the records from the reservations table of the DB. Then loop through the resultset to create a list of HotelReservation objects. This method will return the HotelReservation list so it can be displayed in the popup form

**Connect the GUI to the HotelDB class**

1. Add code to the “Book It!” button click event so it will create a **HotelReservation** object based on what the user entered. Then you must call the **AddReservation** method of the HotelDB class to insert the record. If the record gets inserted successfully, display the popup form. If any errors occur, display them in an **alert** box, and keep the user on the main form.
2. On the controller of the popup form, you’ll have to implement the **Initializable** interface so you can use the **initialize** method to call the **GetReservations** method of the HotelDB class. Then you can loop through the list of HotelReservation objects that come back, displaying them all in the ListView like the screenshot below.

A screenshot of a computer

Description automatically generated