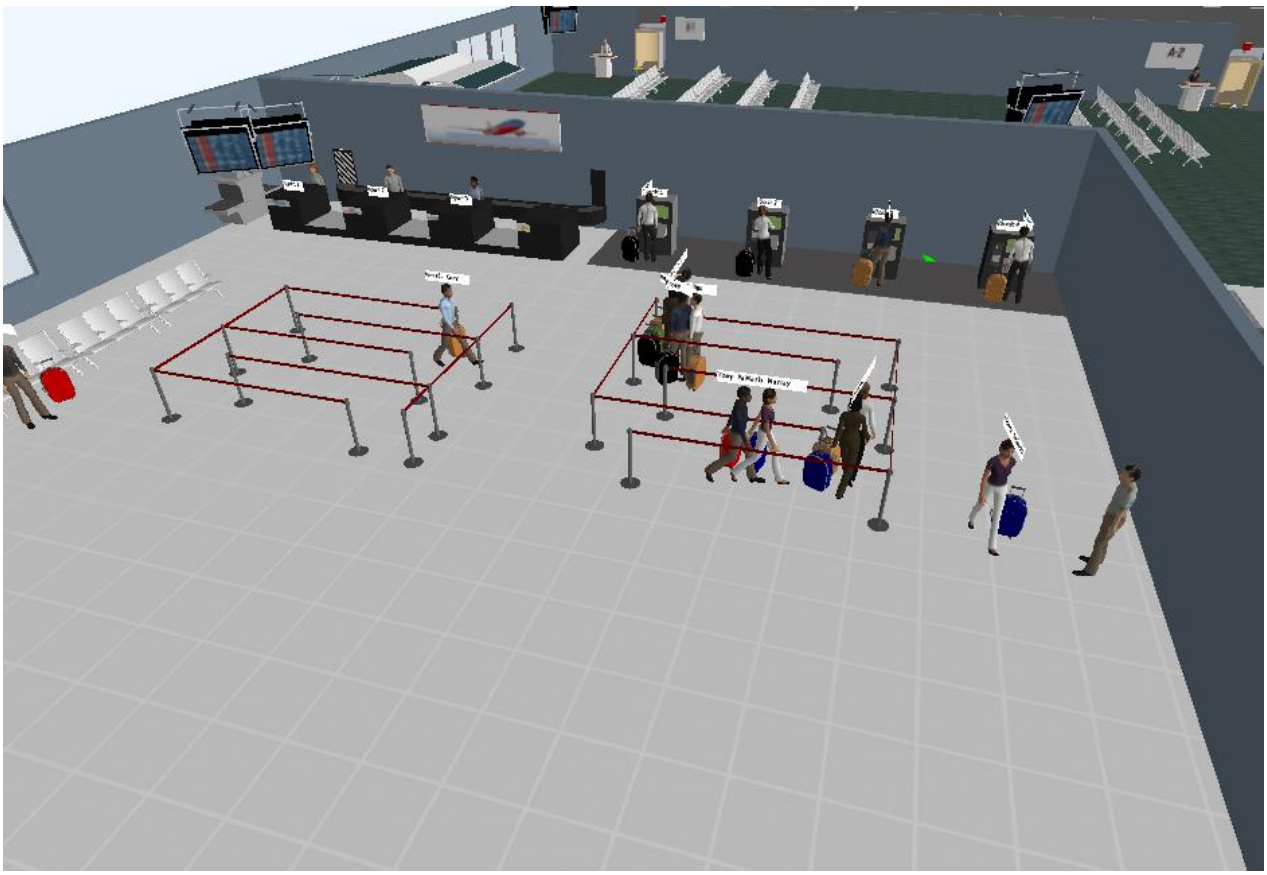
GUI Database Application Development of Airport Using Object Oriented Programming
PART (Question) 2

Time Estimate

Setting up the Project File	: 10 min
Creating the GUI	: 60 min
Developing the Controller for the GUI	: 80 min
Testing the Application with Examples from Simulation	: 30 min

GUI Database Application Development of Airport Using Object Oriented Programming (50 pts total)

The construction of a new airport terminal in a small town has recently been completed. The terminal has eight self-check-in machines, one clerk check-in, one ID check and two advanced imaging technology (AIT) units for scanning passengers and their luggage. You are hired as a software engineer to develop an information system using an object-oriented programming language. Your team is assigned to observe the airport location and develop a simple GUI that will allow users to perform CRUD functionalities. You will be provided two tables and their attributes. You will have to create the tables through flyway using SQL commands. The following figure shows a simulation model of the airport, which we will treat as the "real" system in this assignment.



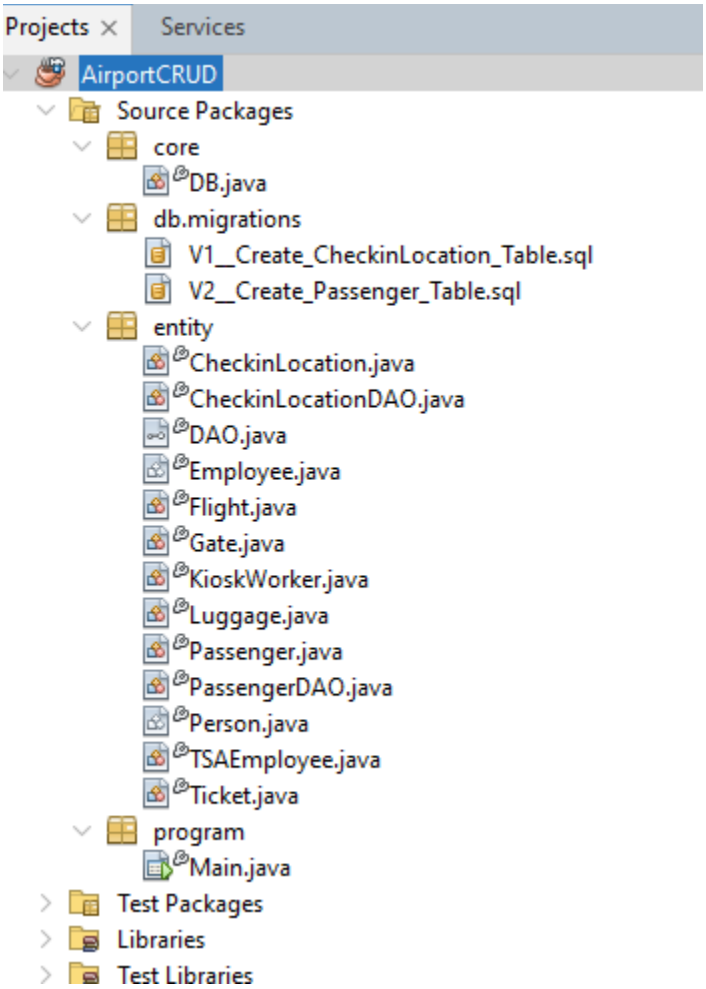
Your project export should include your existing database (created by flyway) and all the other necessary source code/libraries to compile and run the project successfully. Test your exported project zip file in another computer before submitting and see if it is working correctly on that

computer when you imported it. In your report, you must show your work with screenshots and descriptions/captions about those screenshots. If you do not show your work, you will not get that point (there is no partial credit).

starter project:

Part 1: Setting Up the Project File (2 pts)

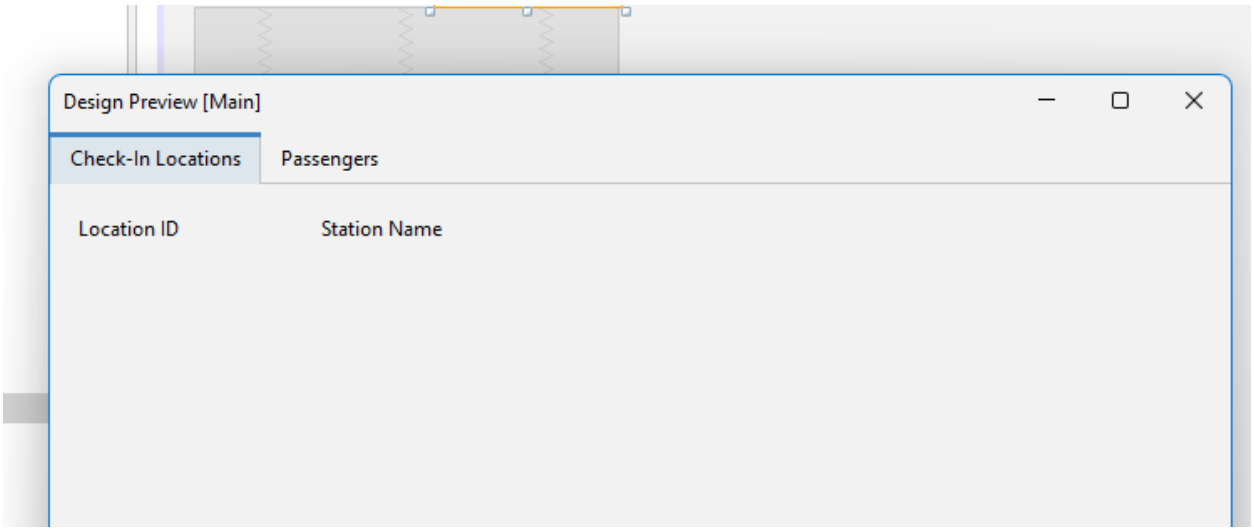
Download the starter project. Rename the project as AirportCRUD. Copy the project into your using NetBeans copy project function (right click to your AirportCRUD and select Copy, please do not copy it any other way). Now delete Main.java file and add a new Java Frame (called Main.java) into your program. Include screenshot of your folder directory.



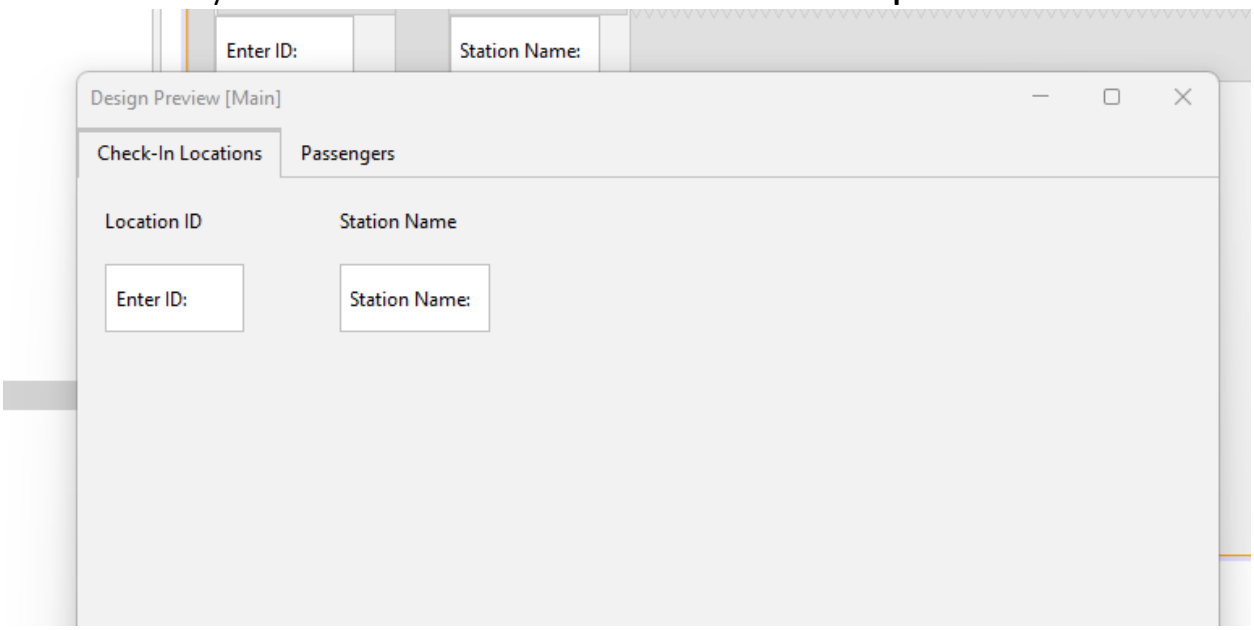
Part 2: Creating the GUI (10 pts)

Create a CRUD GUI application for Airport database. You can use split pane, tabbed pane, or scroll pane to layout your GUI palette. In your Main.java using Java Swing create a shell graphical interface.

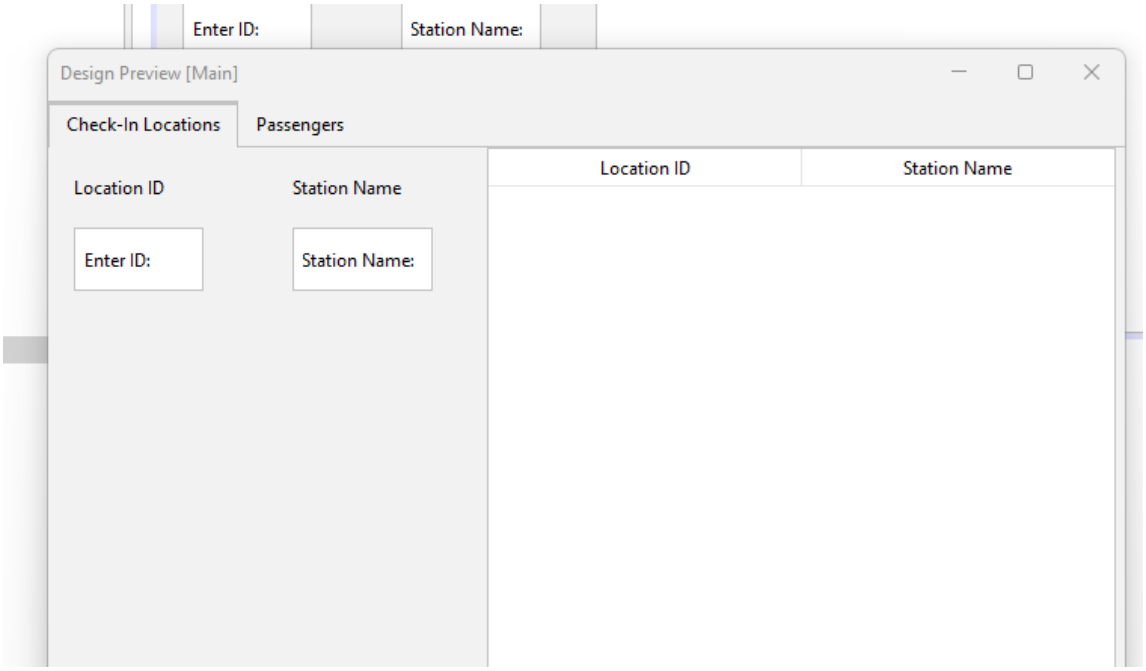
1. Create necessary labels for CheckInLocation with screenshots. **1 point.**



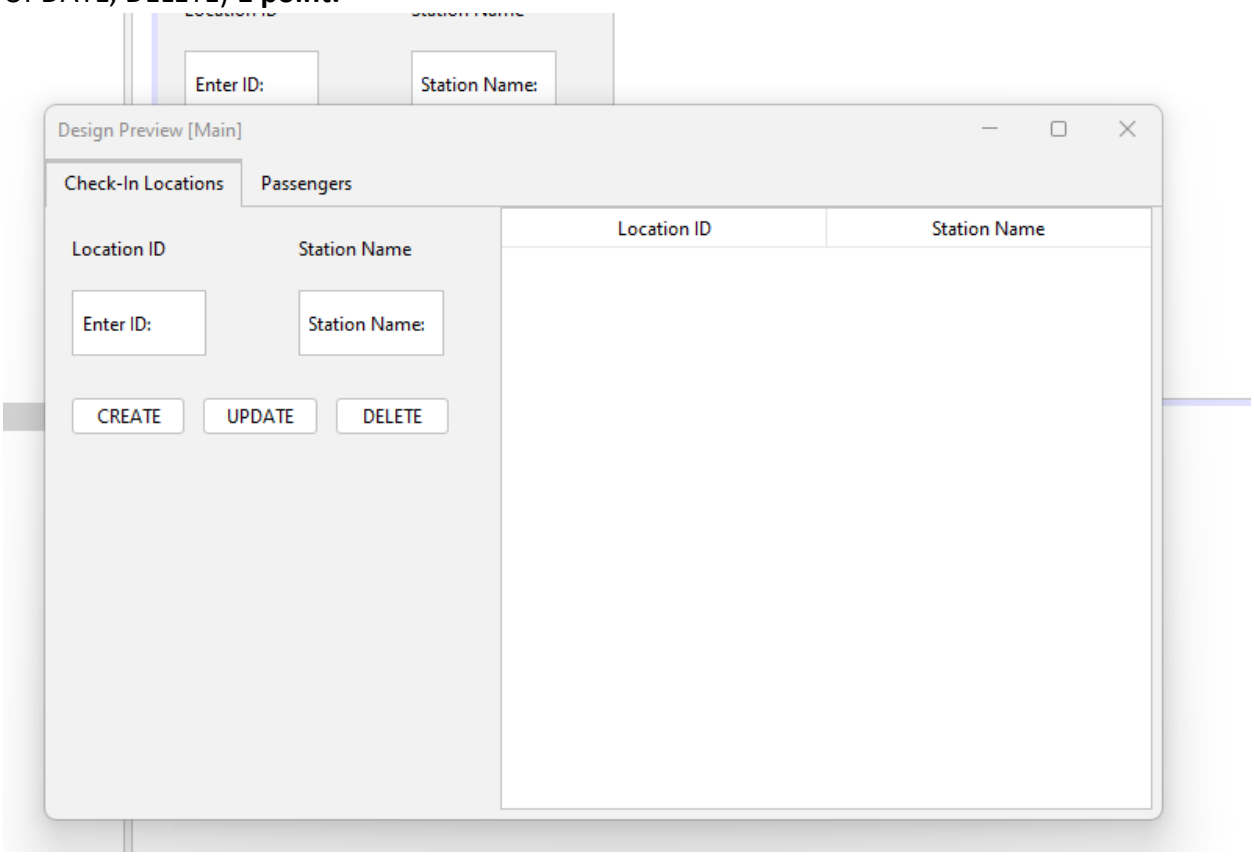
2. Create necessary text fields for CheckInLocation with screenshots. **1 point.**



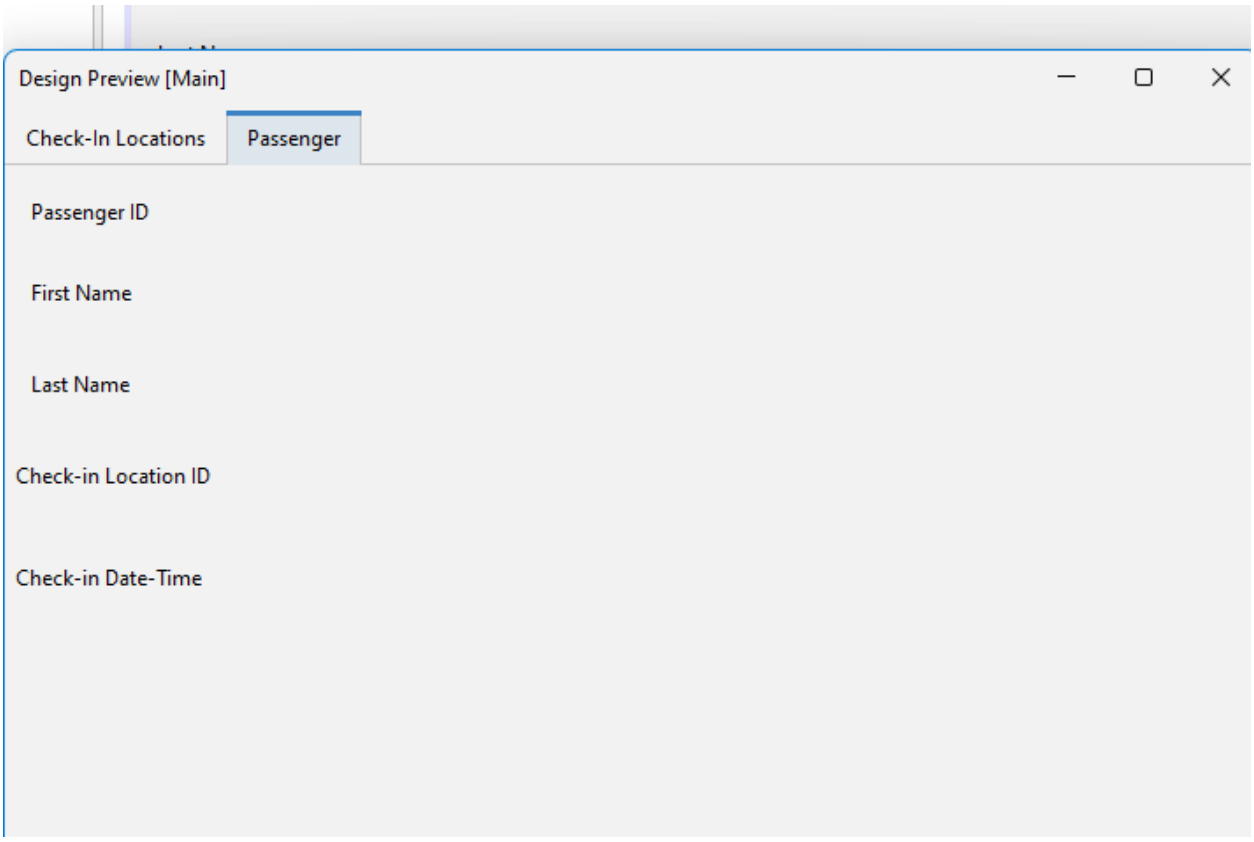
3. Create/Edit the JTable to include attributes from the CheckInLocation table. Provide screenshots. **1 point.**



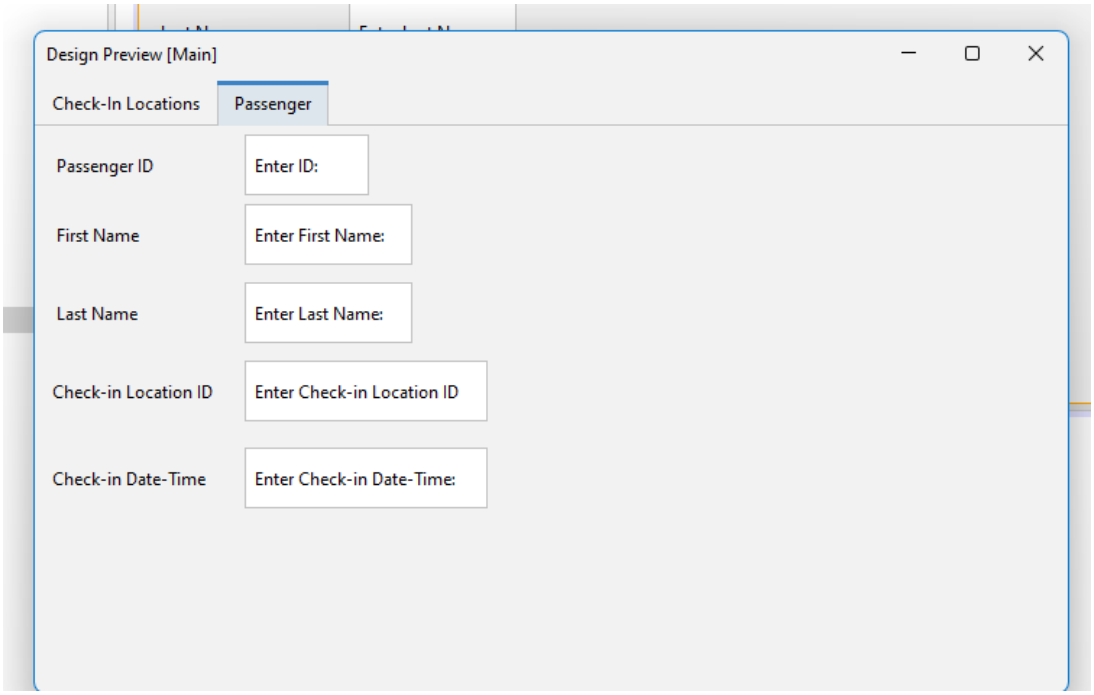
4. Create buttons for CheckInLocation CRUD functionalities with screenshots. (*i.e.*, CREATE, UPDATE, DELETE) **1 point.**



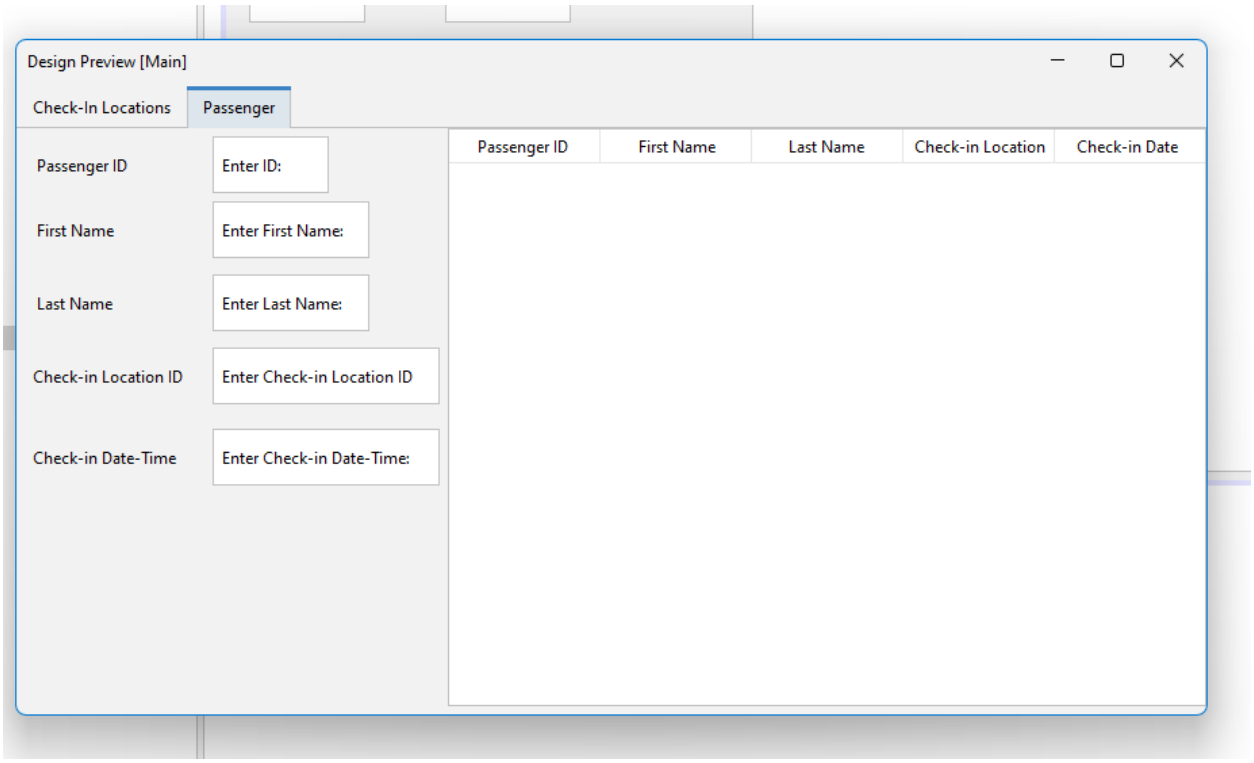
5. Create necessary labels for Passenger with screenshots. **1 point.**



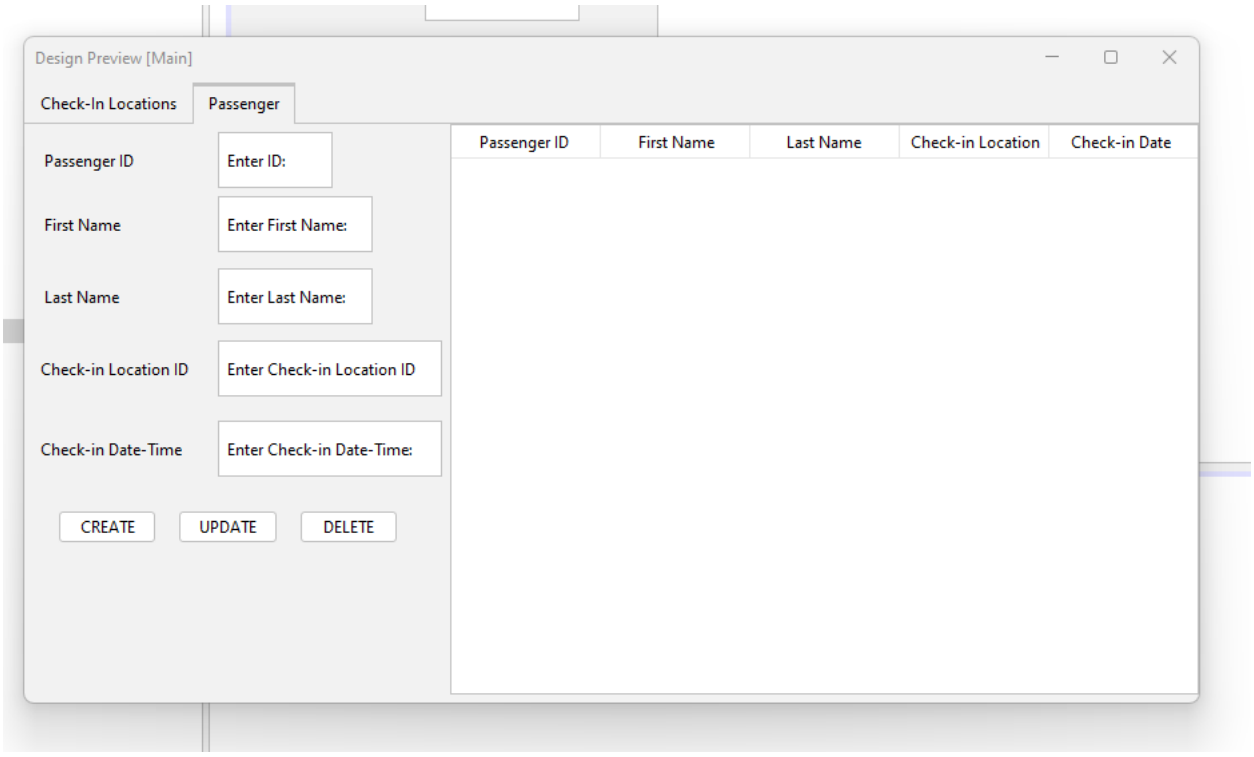
6. Create necessary text fields for Passenger with screenshots. **1 point.**



7. Create/Edit the JTable to include attributes from the Passenger table. Provide screenshots. **1 point.**



8. Create buttons for Passenger CRUD functionalities with screenshots. (*i.e.*, CREATE, UPDATE, DELETE) **1 point.**



9. Explain the layout used in you GUI and why you choose to use such layout. (150 words max) **2 points.**

So, I chose to have 2 tabs one for passenger, and one for check-in location so it clean and organized in those tabs I chose to have labels next to the inputs and at the bottom I placed the buttons to keep them all together, then to the right I put the table so that its neat and organized. I chose to do it this way because it looks the most user friendly and organized.

Part 3: Developing the Controller for the GUI (8 pts)

In your Main.java, write the necessary methods and action events for your GUI interface to function. Be sure to use comments to label what each method does and output.

1. Provide screenshot for each CRUD function methods for CheckInLocation. **4 points.**

Create:

```
private void CreateActionPerformed(java.awt.event.ActionEvent evt) {
    String ID = LocationId.getText().trim();
    String stationName = StationName.getText().trim();
    // check to see its not empty
    if(ID.isEmpty() || stationName.isEmpty()){
        JOptionPane.showMessageDialog(this, "ID and Station Name cannot be empty", "Insert Error", JOptionPane.ERROR_MESSAGE);
        return;
    }
    int id = Integer.parseInt(ID);
    // Check if the ID already exists
    if (checkIfLocationExists(id)) {
    JOptionPane.showMessageDialog(this, "Check-in Location ID already exists! Cannot insert duplicate.", "Primary Key Violation", JOption
    return;
    }
    //refresh and insert
    CheckinLocation newLocation = new CheckinLocation(id, stationName);
    checkinLocationDAO.insert(newLocation);
    refreshCheckInTable();
    JOptionPane.showMessageDialog(this, "Location added successfully!");
}
```

```
    }
/ check to see if it exists
    private boolean checkIfLocationExists(int id) {
    Optional<CheckinLocation> existingLocation = checkinLocationDAO.get(id);
    return existingLocation.isPresent(); // Returns true if the location exists

/table selecting a row in the JTable
```

Read: this is technically my read I'm bring my data from my db table and pasting into my jtable

```java
private void refreshCheckInTable() {
        // Refresh the JTable with the latest locations from the database
        List<CheckinLocation> locations = checkinLocationDAO.getAll();
        DefaultTableModel model = (DefaultTableModel) CheckInTable.getModel();
        model.setRowCount(0);// Clear existing rows
        for (CheckinLocation location : locations) {
            Object[] row = new Object[2];
            row[0] = location.getCheckInLocationID();
            row[1] = location.getStationName();
            model.addRow(row);
        }
    }
//update
```

Update:

```java
private void UpdateActionPerformed(java.awt.event.ActionEvent evt) {
    String ID = LocationId.getText().trim();
    String stationName = StationName.getText().trim();
     // Ensure ID and Station Name are not empty
    if (ID.isEmpty() || stationName.isEmpty()) {
        JOptionPane.showMessageDialog(this, "ID and Station Name cannot be empty", "Update Error", JOptionPane.ERROR_MESSAGE);
    }
    int id = Integer.parseInt(ID);
    Optional<CheckinLocation> locationOpt = checkinLocationDAO.get(id);
    if(locationOpt.isPresent()){
        //update
        checkinLocationDAO.update(new CheckinLocation(id, stationName));
    // Refresh the table to show the new location
        refreshCheckInTable();
        JOptionPane.showMessageDialog(this, "Location updated successfully!");
    } else {
        // If the location does not exist, show an error message
        JOptionPane.showMessageDialog(this, "Location with ID " + id + " does not exist.", "Update Error", JOptionPane.ERROR_MESSAGE);
    }
}
```

delete

Delete:

```java
private void DeleteActionPerformed(java.awt.event.ActionEvent evt) {
    String ID = LocationId.getText().trim();
    //check to see its empty
    if (ID.isEmpty()) {
        JOptionPane.showMessageDialog(this, "ID cannot be empty", "Delete Error", JOptionPane.ERROR_MESSAGE);
    }
    int id = Integer.parseInt(ID);
    Optional<CheckinLocation> locationOpt = checkinLocationDAO.get(id);
    //delete
    if (locationOpt.isPresent()) {
    int option = JOptionPane.showConfirmDialog(this, "Are you sure you want to delete this location?", "Delete Confirmation", JOptionPane.YES_NO_OPTION);
        if (option == JOptionPane.YES_OPTION) {
            checkinLocationDAO.delete(locationOpt.get());  // Call DAO to delete the location
            refreshCheckInTable(); // Refresh the table
            JOptionPane.showMessageDialog(this, "Location deleted successfully!");
        }
    } else {
        // If the location doesn't exist, show an error message
        JOptionPane.showMessageDialog(this, "Location with ID " + id + " does not exist.", "Delete Error", JOptionPane.ERROR_MESSAGE);
    }
}
:eate
```

2. Provide screenshot for each CRUD function methods for Passenger. **4 points.**

Create:

```
private void CreatePassengerActionPerformed(java.awt.event.ActionEvent evt) {
    String ID = PassengerID.getText().trim();
    String firstName = FirstName.getText().trim();
    String lastName = LastName.getText().trim();
    String checkinID = CheckinID.getText().trim();
    String checkInTimeStr  = checkINTime.getText().trim();
    // check to see if anything is empty
    if(ID.isEmpty() || firstName.isEmpty()|| lastName.isEmpty()|| checkinID.isEmpty()|| checkInTimeStr.isEmpty()){
        JOptionPane.showMessageDialog(this, "You can't keep any field empty", "Insert Error", JOptionPane.ERROR_MESSAGE);
        return;
    }
    int id = Integer.parseInt(ID);
    int checkinIDs = Integer.parseInt(checkinID);
    // Convert checkInTimeStr (String) to Timestamp
alDateTime checkInDateTime = null;
 {
LocalDateTime checkInDateTime = null;
try {
    // Assume the format of checkInTimeStr is "yyyy-MM-dd HH:mm:ss"
    checkInDateTime = LocalDateTime.parse(checkInTimeStr.replace(" ", "T")); // Adjust the format as necessary
} catch (DateTimeParseException e) {
    JOptionPane.showMessageDialog(this, "Invalid date/time format. Please use 'YYYY-MM-DD HH:MM:SS'.", "Insert Error", JOpti
    return; // Don't proceed if there's an invalid date format
}
// Check if the CheckInLocationID exists in the CheckinLocation table
CheckinLocationDAO checkinLocationDAO = new CheckinLocationDAO();
Optional<CheckinLocation> checkinLocation = checkinLocationDAO.get(checkinIDs);
if (!checkinLocation.isPresent()) {
    JOptionPane.showMessageDialog(this, "Invalid Check-In Location ID! This location does not exist.", "Foreign Key Violatio
    return; // Don't proceed if the CheckInLocationID is invalid
}
// Now create a new Passenger object
PassengerDAO passengerDAO = new PassengerDAO();
Passenger newPassenger = new Passenger(id, firstName, lastName, checkinIDs, checkInDateTime); // Pass Timestamp here
passengerDAO.insert(newPassenger);
        refreshPassengerTable();
        JOptionPane.showMessageDialog(this, "Passenger added successfully!");
    }
```

Read: this is technically my read I'm bring my data from my db table and pasting into my jtable

```
private void refreshPassengerTable() {
        // Refresh the JTable with the latest locations from the database
        List<Passenger> allPassengers = passengerDAO.getAll();
        DefaultTableModel model = (DefaultTableModel) passengerTable.getModel();
        model.setRowCount(0);// Clear existing rows
        for (Passenger passenger : allPassengers) {
            Object[] row = new Object[5];
            row[0] = passenger.getPassengerID();
            row[1] = passenger.getFirstName();
            row[2] = passenger.getLastName();
            row[3] = passenger.getCheckInLocationID();
            row[4] = passenger.getCheckInDateTime();
            model.addRow(row);
        }
}
```

Update:

```
//update
    private void updatePassengerActionPerformed(java.awt.event.ActionEvent evt) {
    String ID = PassengerID.getText().trim();
    String firstName = FirstName.getText().trim();
    String lastName = LastName.getText().trim();
    String checkinID = CheckinID.getText().trim();
    String checkInTimeStr = checkINTime.getText().trim();

if (ID.isEmpty() || firstName.isEmpty() || lastName.isEmpty() || checkinID.isEmpty() || checkInTimeStr.isEmpty()) {
    JOptionPane.showMessageDialog(this, "You can't keep any field empty", "Update Error", JOptionPane.ERROR_MESSAGE);
    return;
}
int id = Integer.parseInt(ID);
int checkinIDs = Integer.parseInt(checkinID);
LocalDateTime checkInDateTime = null;
try {
    checkInDateTime = LocalDateTime.parse(checkInTimeStr.replace(" ", "T"));  // Convert to LocalDateTime
} catch (DateTimeParseException e) {
    JOptionPane.showMessageDialog(this, "Invalid date/time format. Please use 'YYYY-MM-DD HH:MM:SS'.", "Update Error", JOptior
    return; // Don't proceed if there's an invalid date format
}

// Check if the Passenger ID exists in the database (Primary Key Check)
PassengerDAO passengerDAO = new PassengerDAO();
Optional<Passenger> existingPassenger = passengerDAO.get(id);
if (!existingPassenger.isPresent()) {
    JOptionPane.showMessageDialog(this, "Passenger ID not found! Cannot update.", "Primary Key Violation", JOptionPane.ERROR_M
    return; // Exit if Passenger doesn't exist
}

// Check if the CheckinLocationID exists in the CheckinLocation table (Foreign Key Check)
```

// Check if the CheckinLocationID exists in the CheckinLocation table (Foreign Key Check)

```
CheckinLocationDAO checkinLocationDAO = new CheckinLocationDAO();
Optional<CheckinLocation> checkinLocation = checkinLocationDAO.get(checkinIDs);
if (!checkinLocation.isPresent()) {
    JOptionPane.showMessageDialog(this, "Invalid Check-In Location ID! This location does not exist.", "Foreign Key Violation", JOptionPane.ERROR_MESSAGE);
    return; // Exit if the CheckInLocationID is invalid (foreign key violation)
}
Passenger updatedPassenger = new Passenger(id, firstName, lastName, checkinIDs, checkInDateTime);
passengerDAO.update(updatedPassenger);
refreshPassengerTable();
JOptionPane.showMessageDialog(this, "Passenger updated successfully!");

    }
//create
```

Delete:

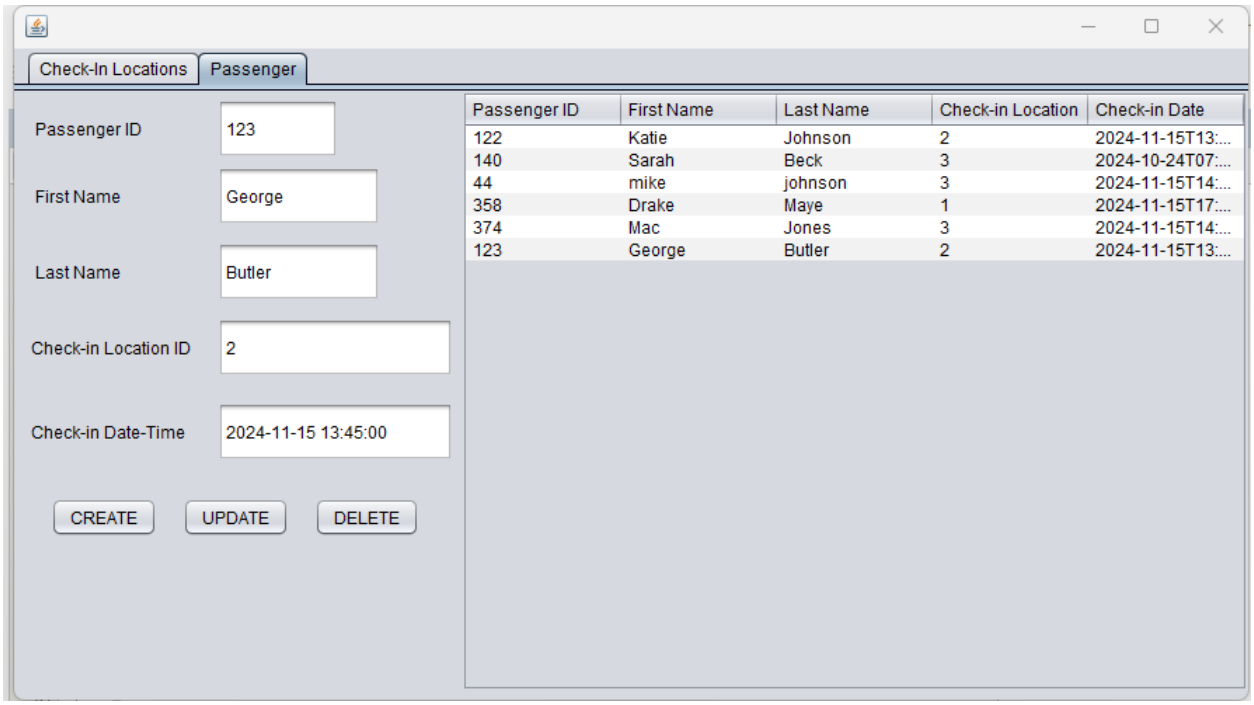
```java
private void deletePassengerActionPerformed(java.awt.event.ActionEvent evt) {
    String ID = PassengerID.getText().trim();
// Check to see if the ID is empty
if (ID.isEmpty()) {
    JOptionPane.showMessageDialog(this, "Passenger ID cannot be empty", "Delete Error", JOptionPane.ERROR_MESSAGE);
    return;
}
int id = Integer.parseInt(ID);
// Check if the Passenger ID exists (Primary Key Check)
PassengerDAO passengerDAO = new PassengerDAO();
Optional<Passenger> existingPassenger = passengerDAO.get(id);
if (!existingPassenger.isPresent()) {
    JOptionPane.showMessageDialog(this, "Passenger ID not found! Cannot delete.", "Primary Key Violation", JOptionPane.ERROR_MESSAGE);
    return; // Exit if Passenger doesn't exist
}
passengerDAO.delete(existingPassenger.get());

// Refresh the table and show success message
refreshPassengerTable();
JOptionPane.showMessageDialog(this, "Passenger deleted successfully!");
}
```

Part 4: Testing the Application with Simulation Data (30 pts)

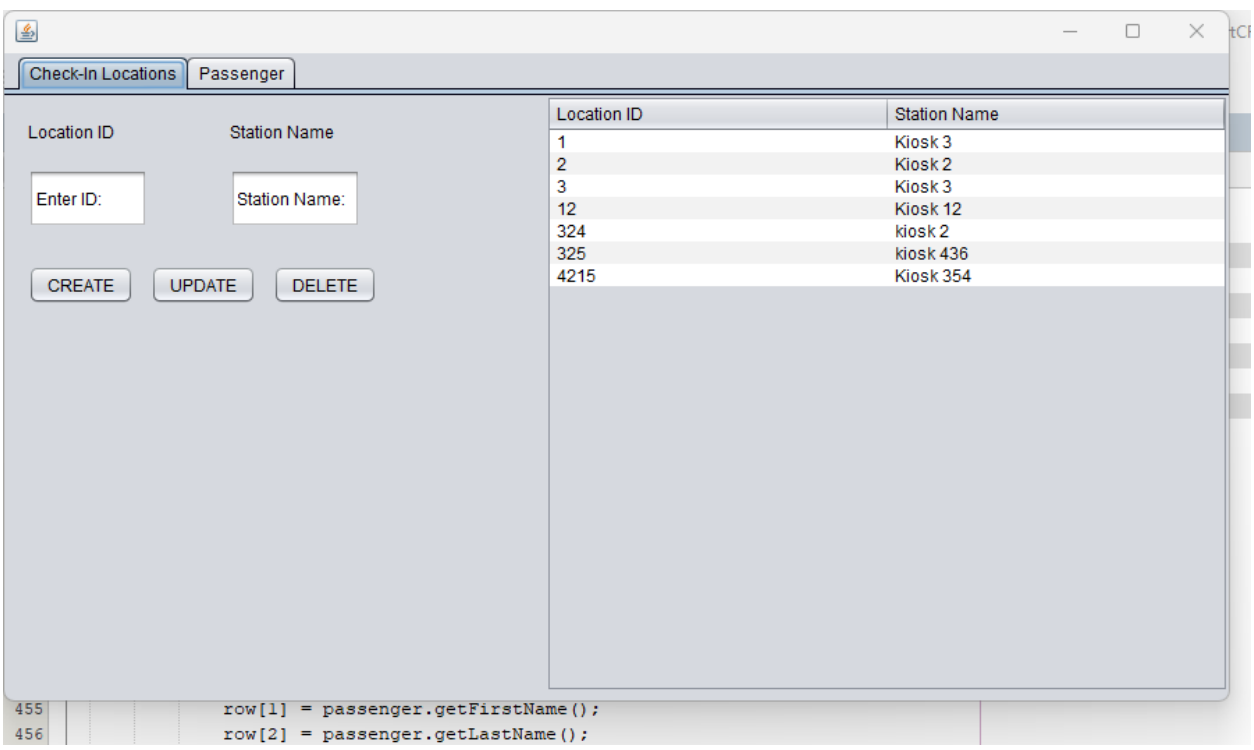
Passenger ID (Given in Simulation)	**First Name**	**Last Name**	**Check in Location**	**Check-in Date Time**
122	Katie	Johnson	7:01 am	Kiosk 5
123	George	Butler	7:01 am	Kiosk 7
140	Sarah	Beck	7:02 am	Kiosk 6

1. Pick two Passengers and record their information and Check-In Location data into the table above. Passenger ID can be collected by pausing the simulation and hovering the cursor over the passenger object. (e.g., Passenger.129 has Passenger ID 129) **2 points.**
   I added 2 from abvove and already had a few in my table:

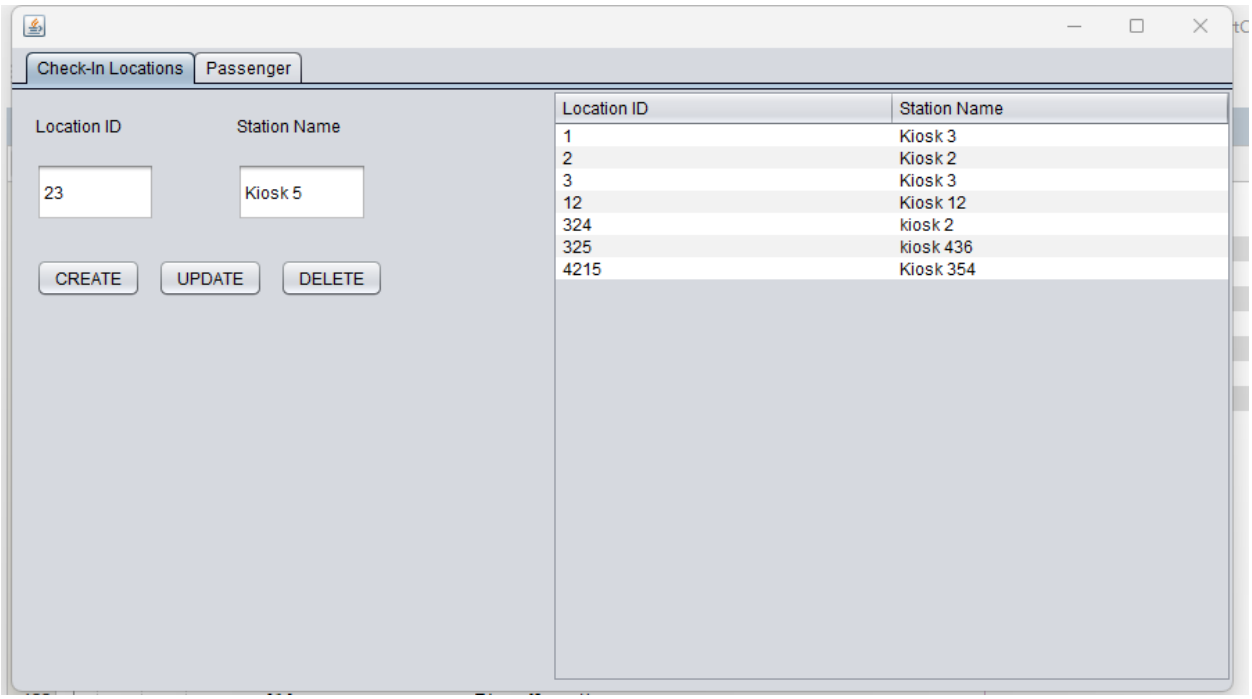


2. Show how you add a new CheckInLocation with screenshots. **3 points.**

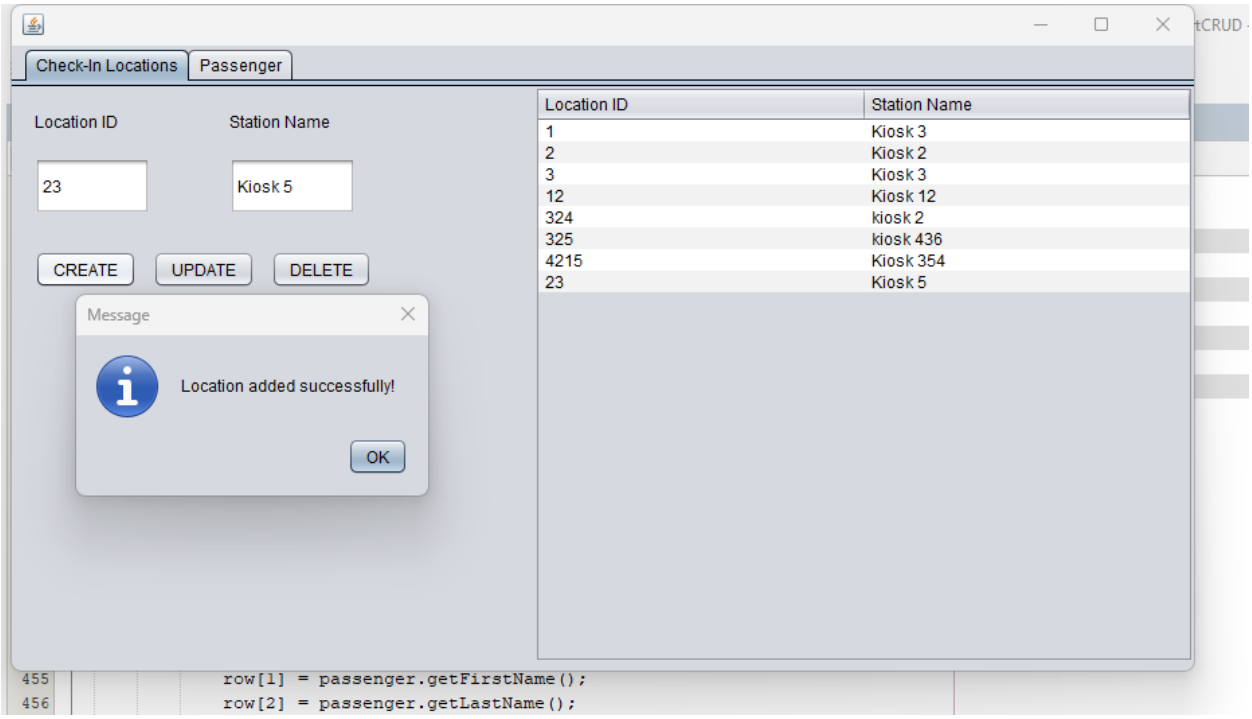
Before:



During:

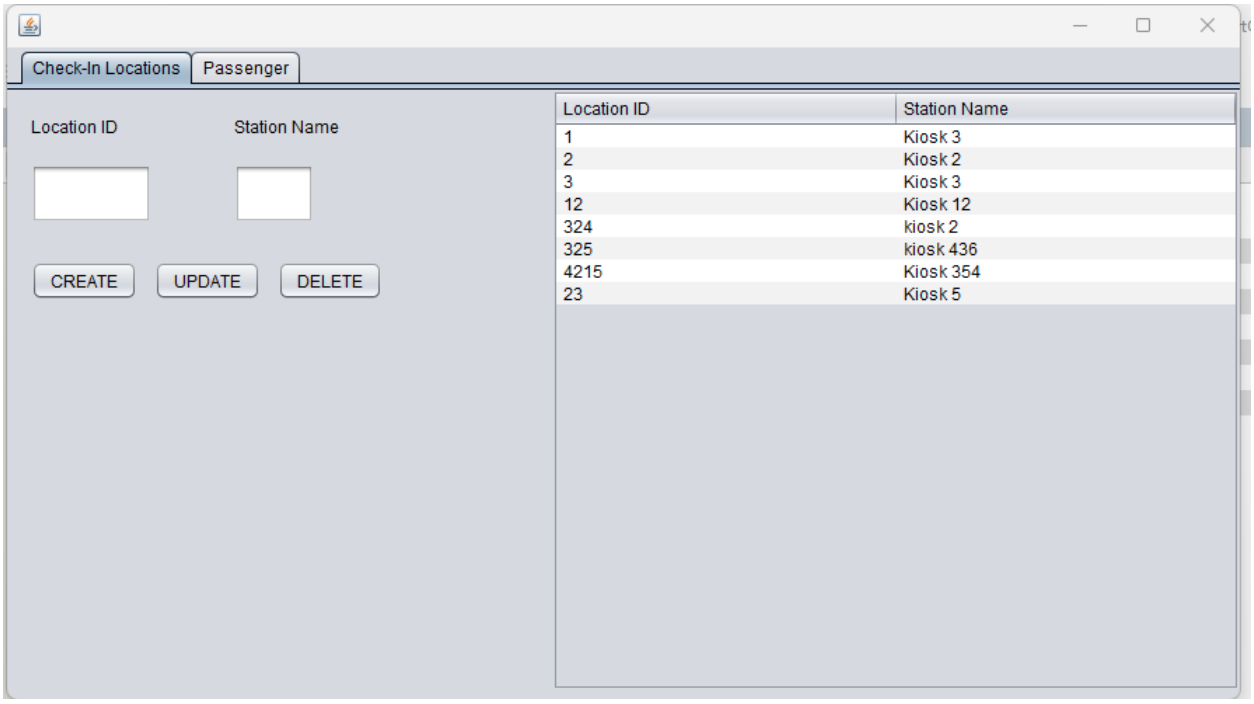


After:

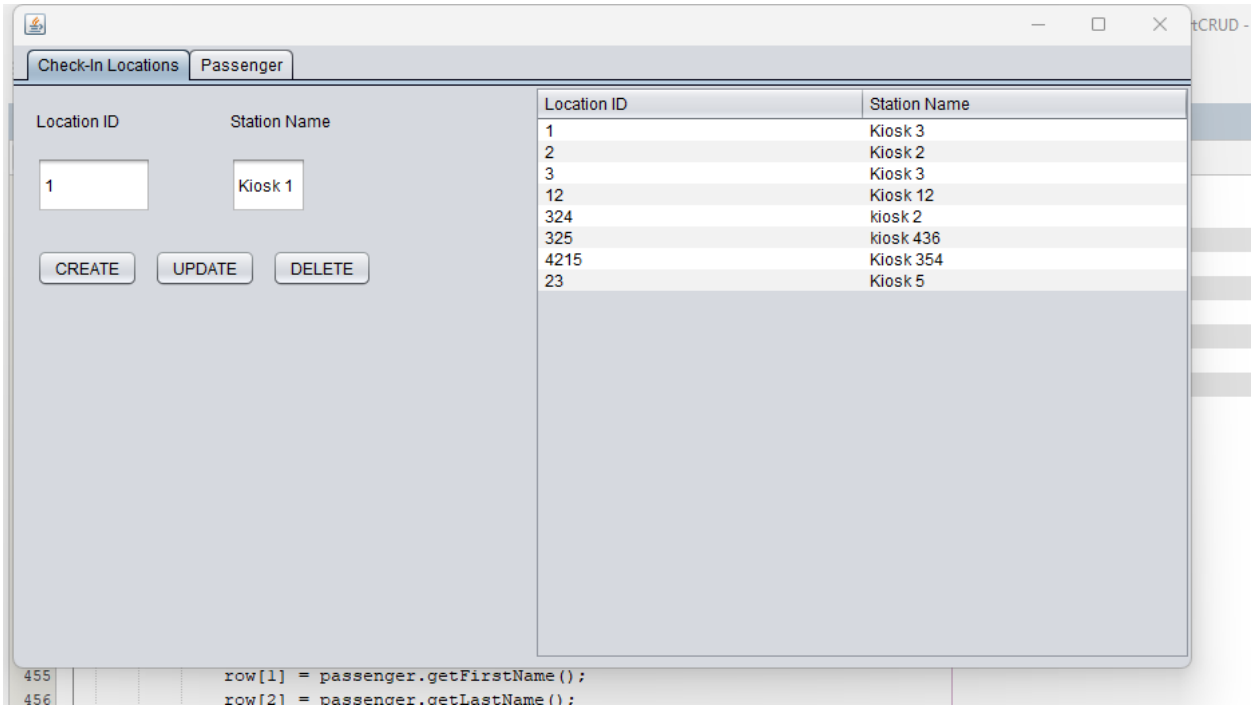


3. Show how you update an existing CheckInLocation with screenshots. **3 points.**

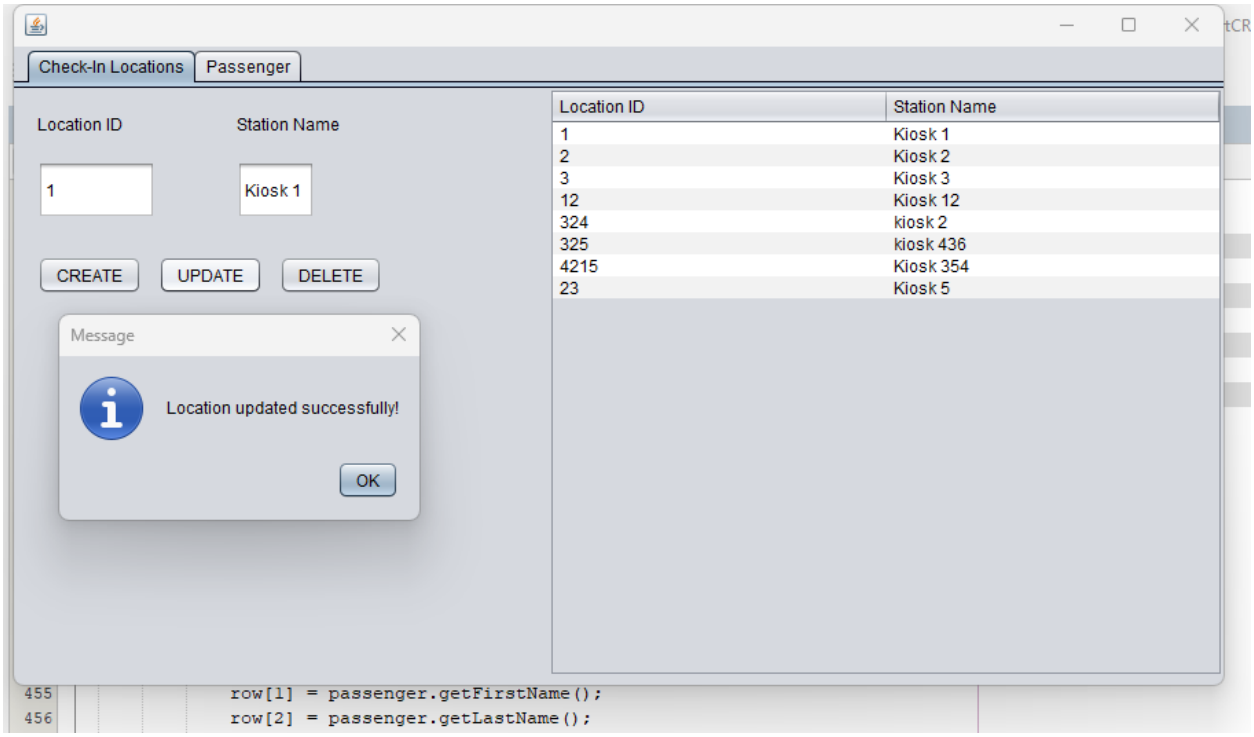
Before:



During:



After:



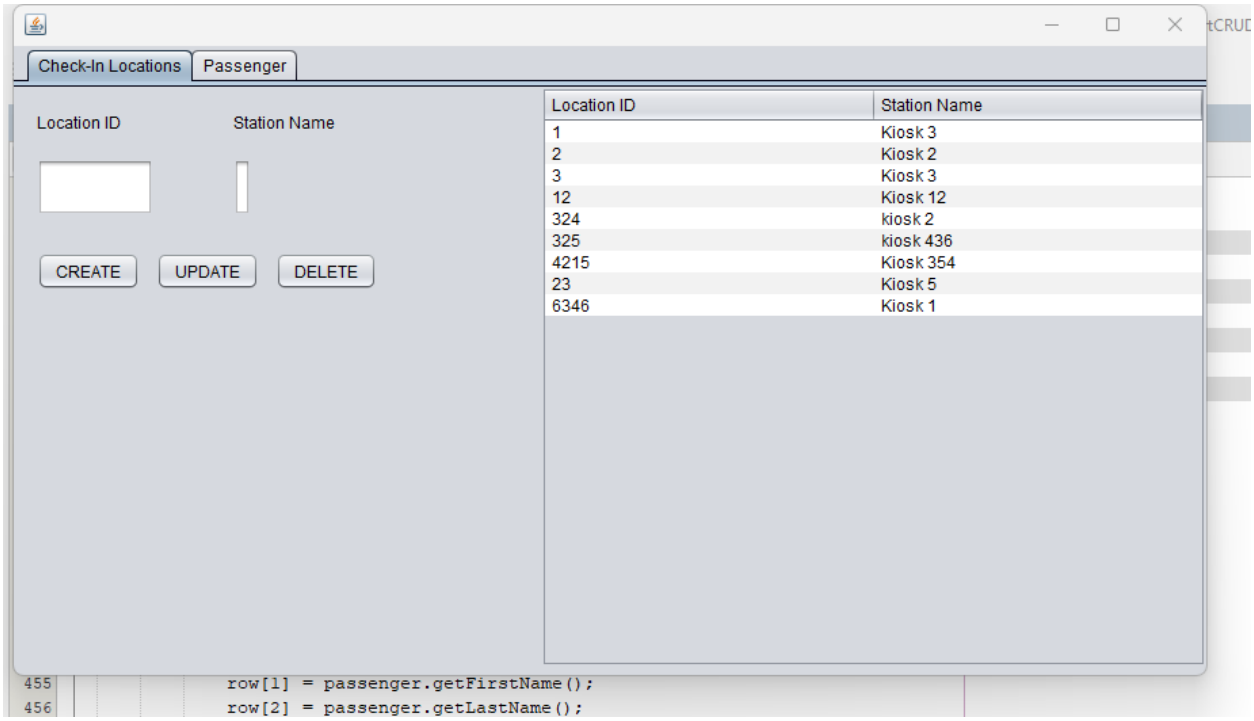
4. Show how you print/list CheckInLocations with screenshots. **3 points.**

   So, for this question I showed the code earlier for my read, and i have the table outputting each time a button is hit so you can see the changes here is a screenshot of it:

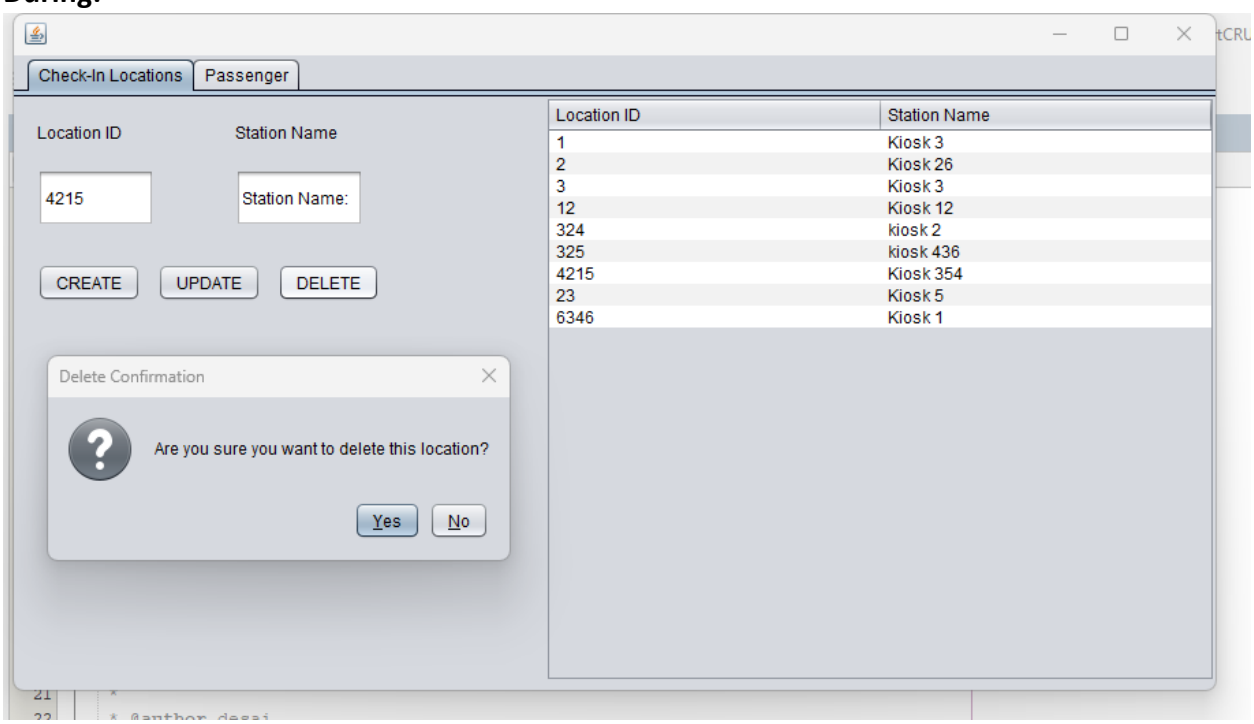
Location ID	Station Name
1	Kiosk 3
2	Kiosk 2
3	Kiosk 3
12	Kiosk 12
324	kiosk 2
325	kiosk 436
4215	Kiosk 354
23	Kiosk 5
6346	Kiosk 1

5. Show how you delete an existing CheckInLocation with screenshots. **3 points.**

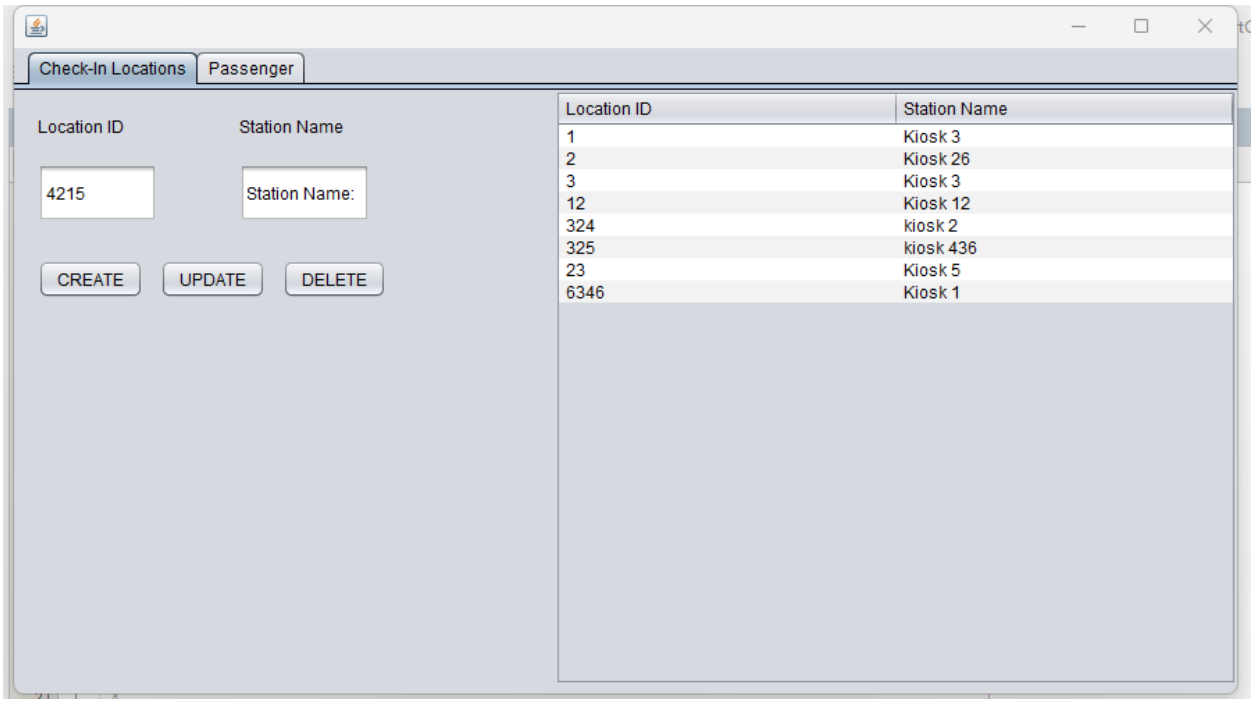
Before:



During:

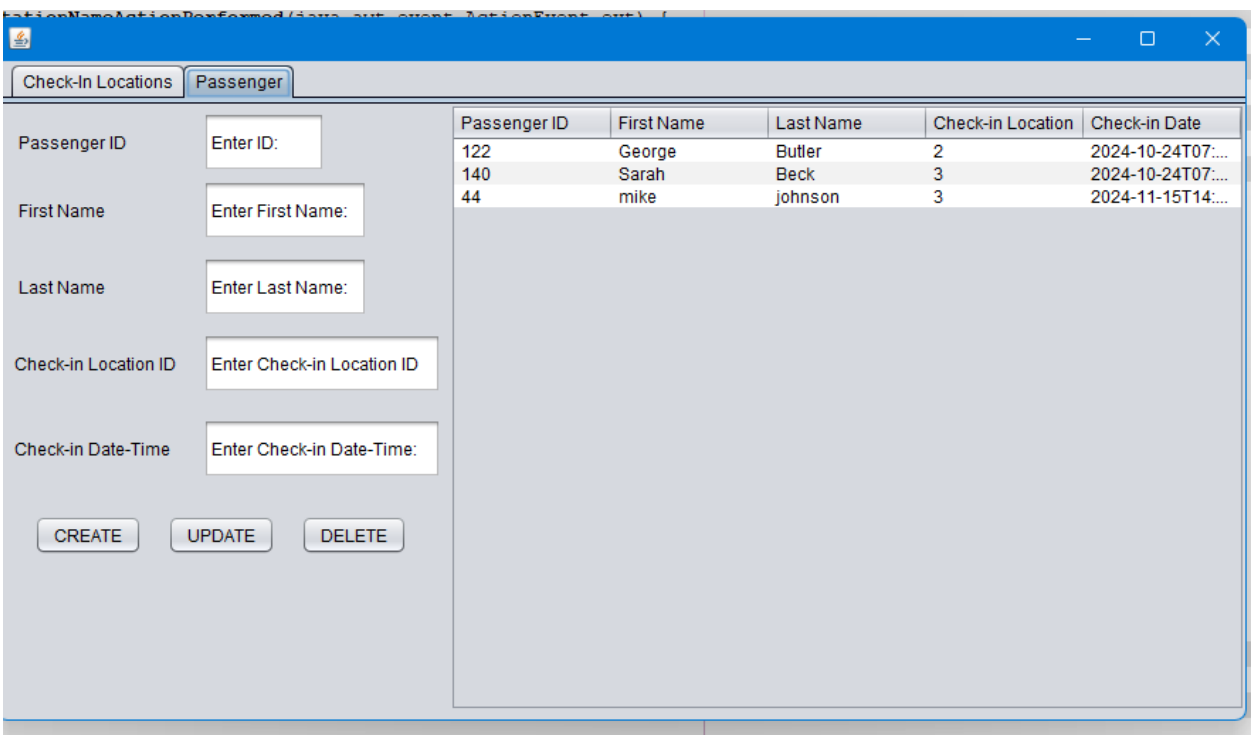


After:

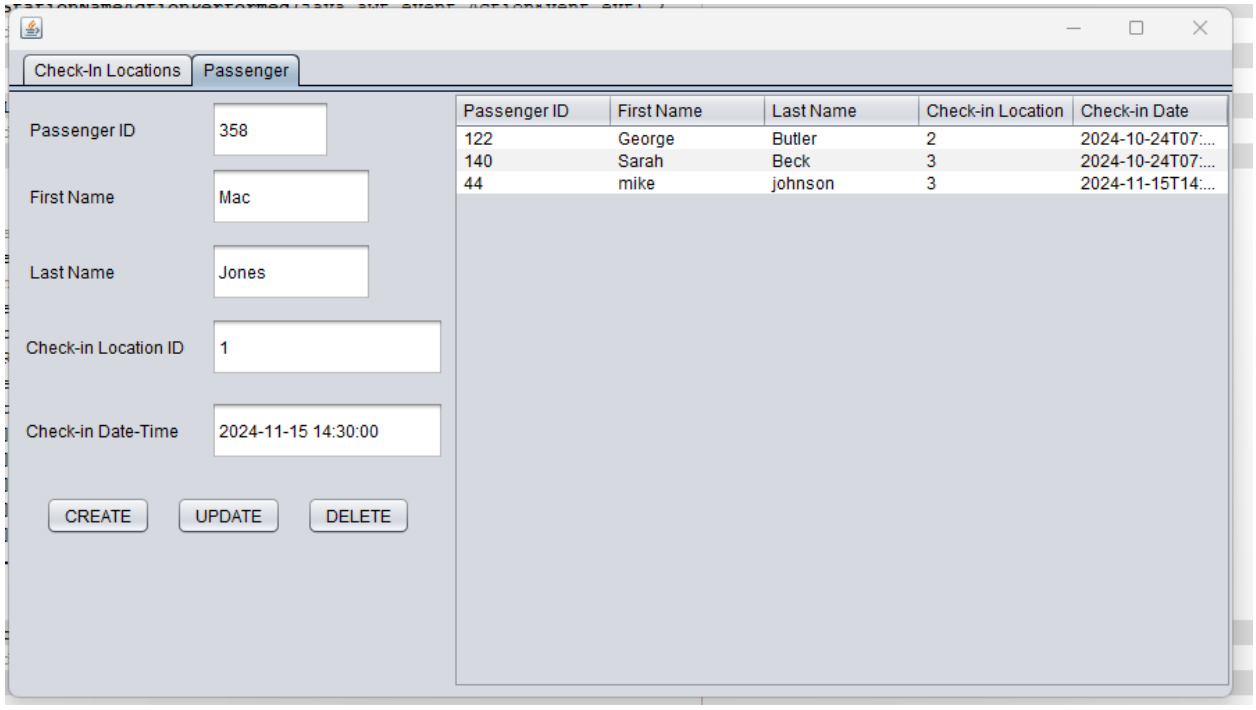


6. Show how you add a new Passenger with screenshots. **3 points.**

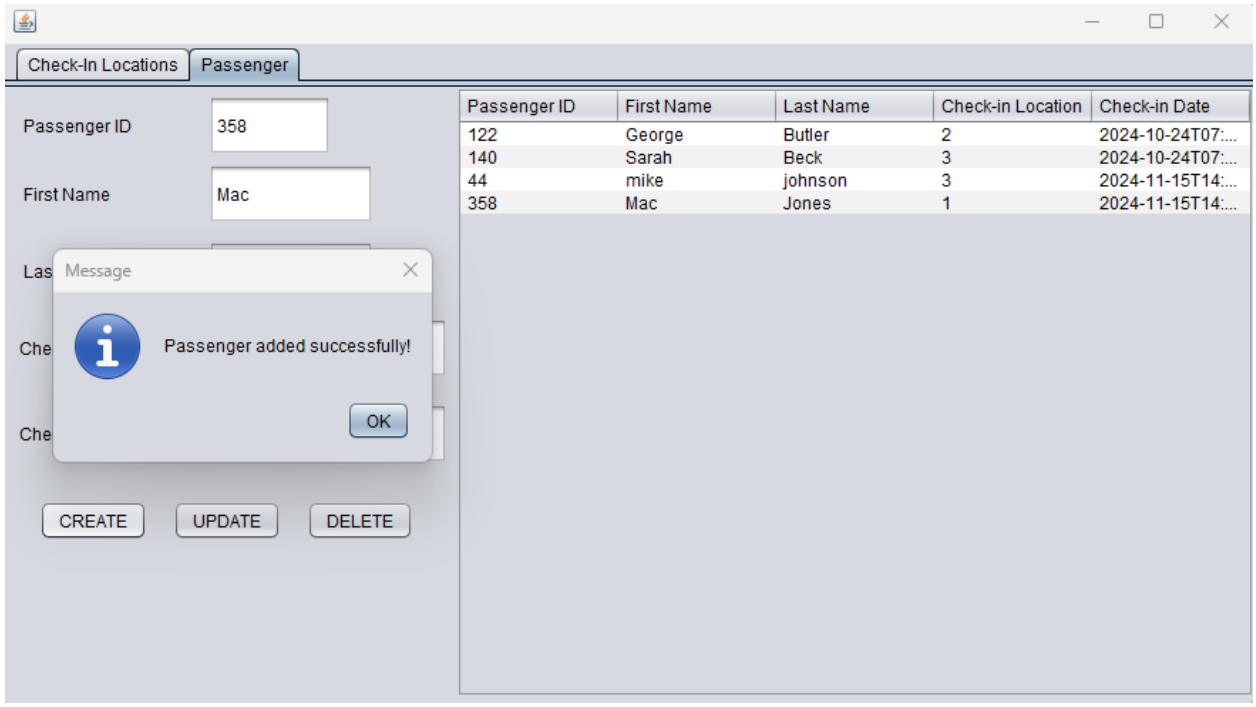
Before:



During:

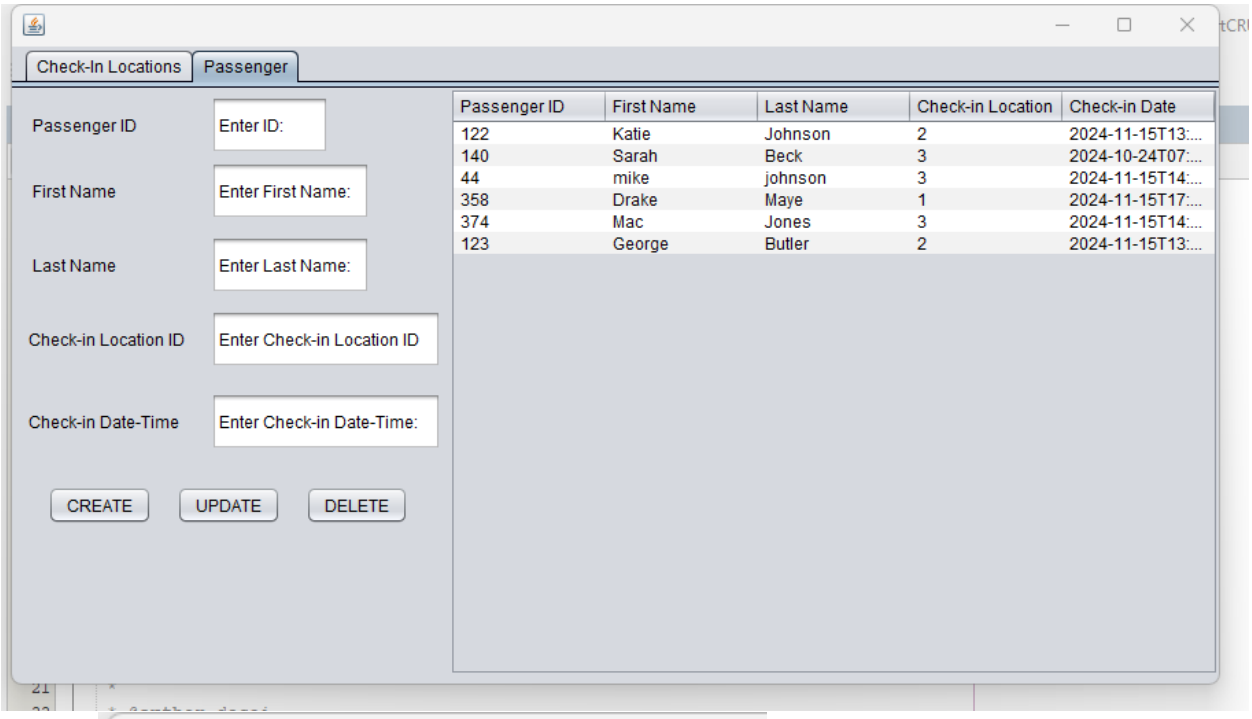


After:

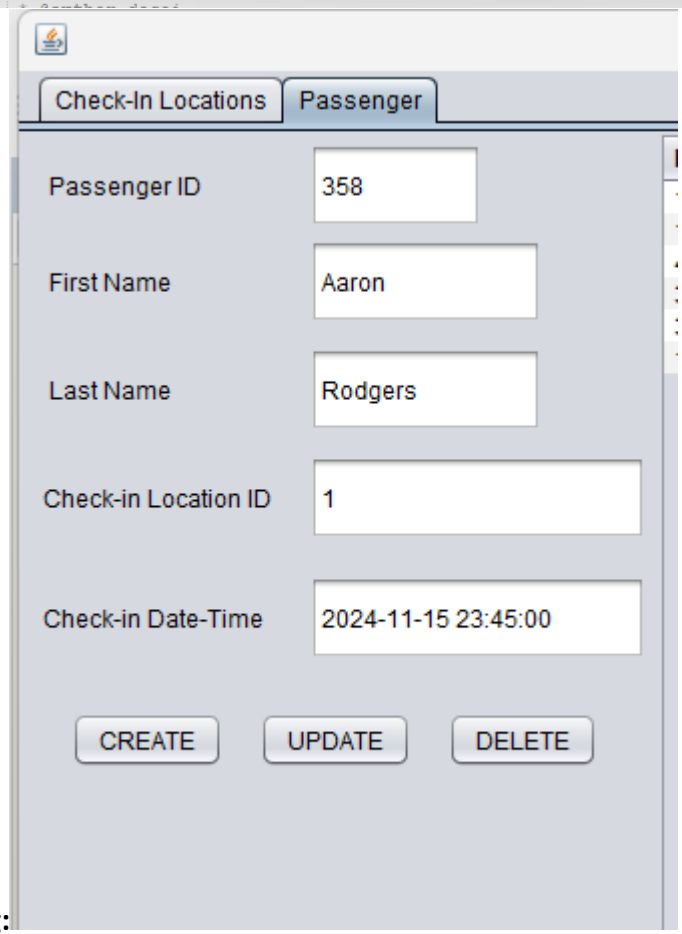


7. Show how you update an existing Passenger with screenshots. **3 points.**

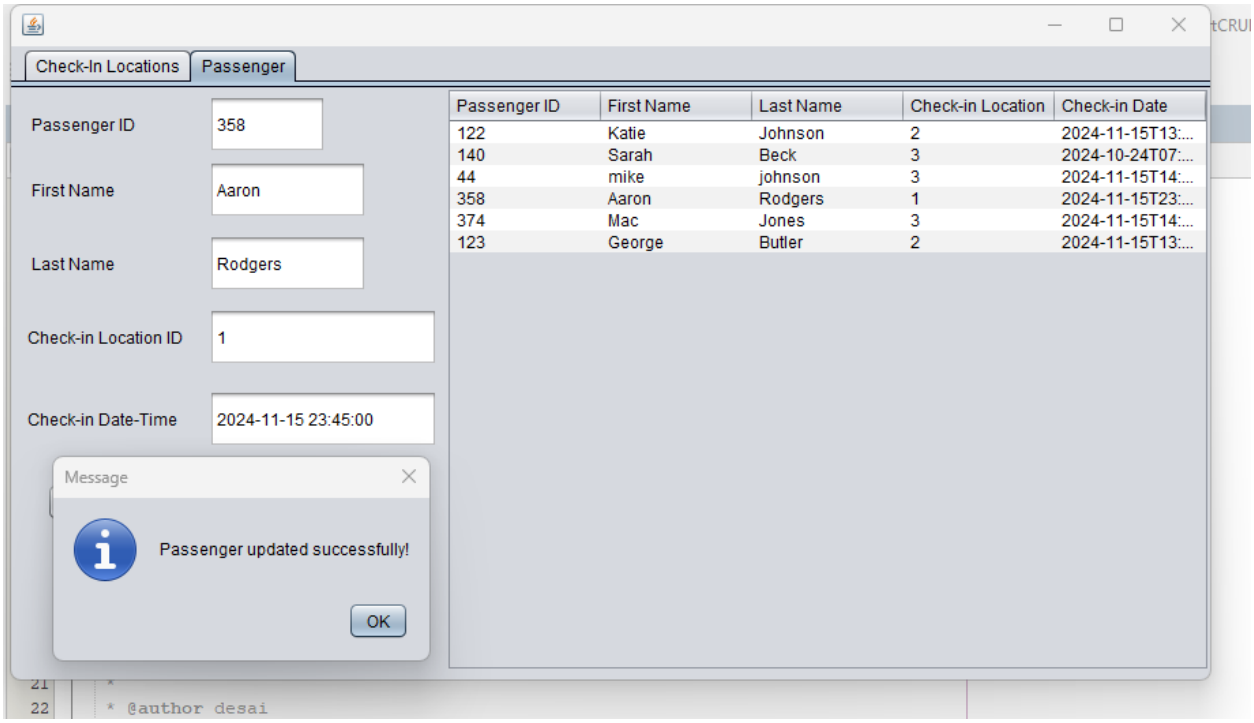
Before:



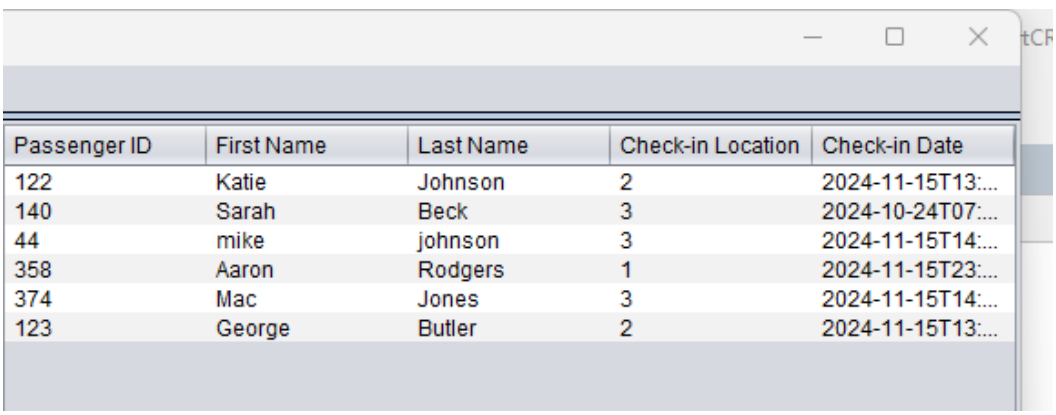
During:



After:



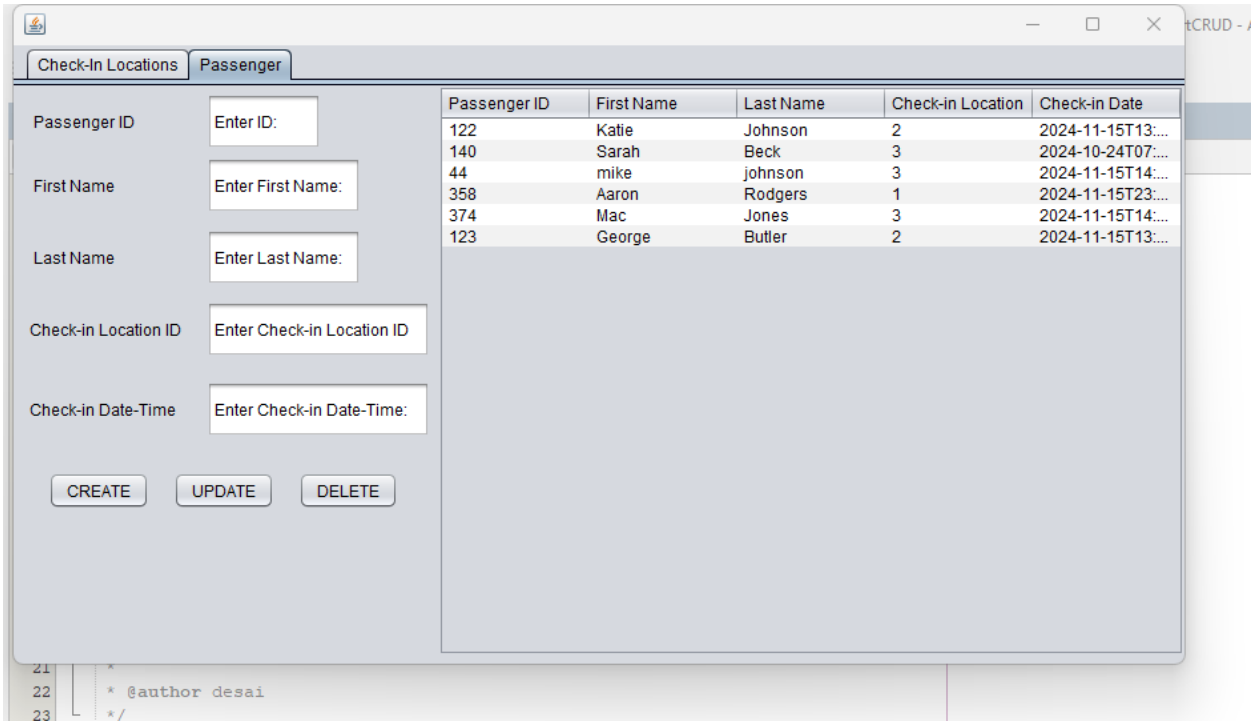
8. Show how you print/list Passengers with screenshots. **3 points.**
   Again, my code prints everything in a table as soon as you run the program and updates every time and actions is done.



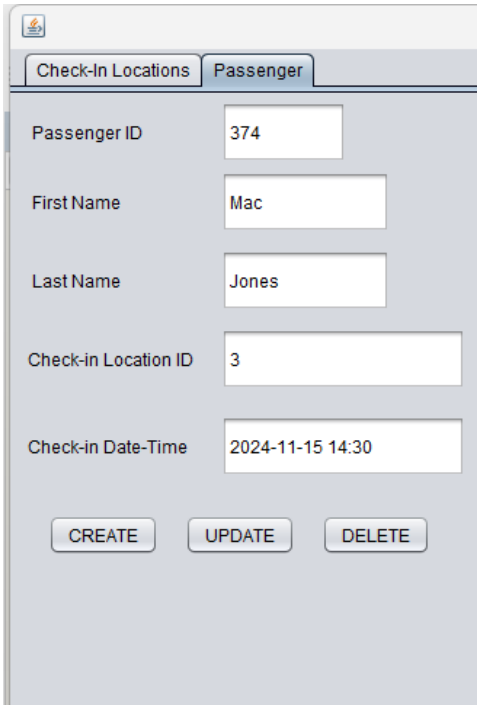
Passenger ID	First Name	Last Name	Check-in Location	Check-in Date
122	Katie	Johnson	2	2024-11-15T13:...
140	Sarah	Beck	3	2024-10-24T07:...
44	mike	johnson	3	2024-11-15T14:...
358	Aaron	Rodgers	1	2024-11-15T23:...
374	Mac	Jones	3	2024-11-15T14:...
123	George	Butler	2	2024-11-15T13:...

9. Show how you delete an existing Passenger with screenshots. **3 points.**

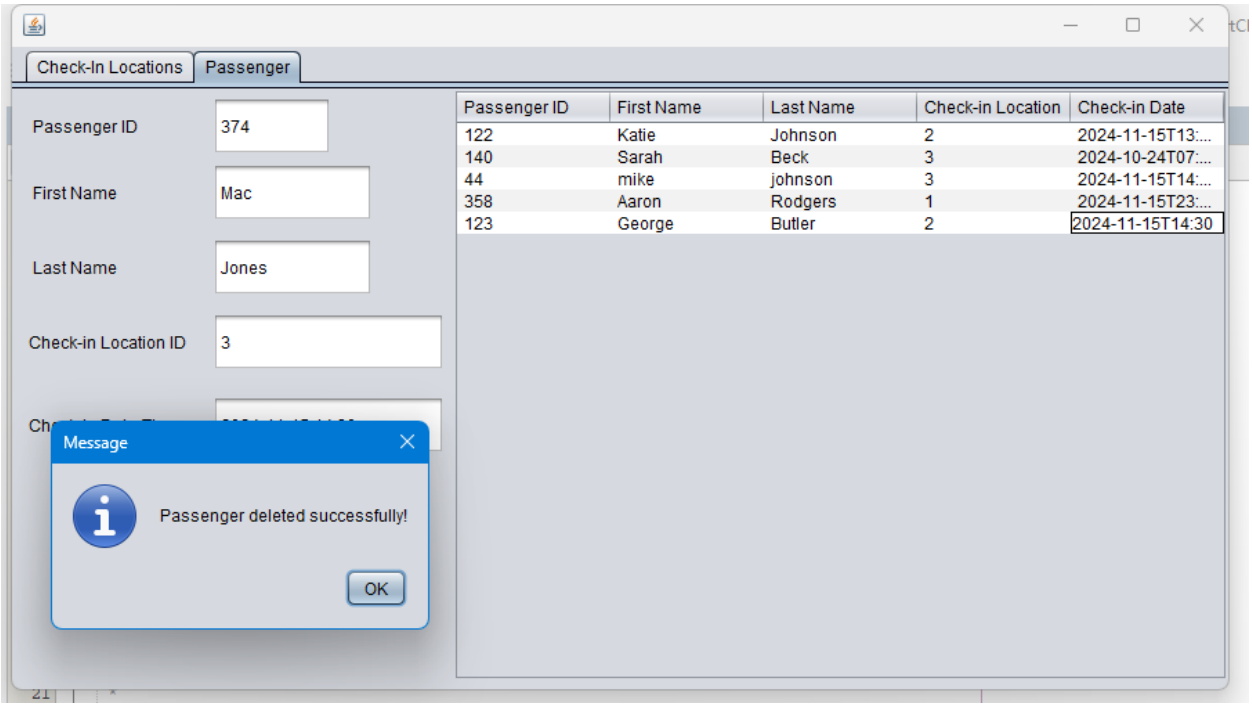
Before:



During:



After:

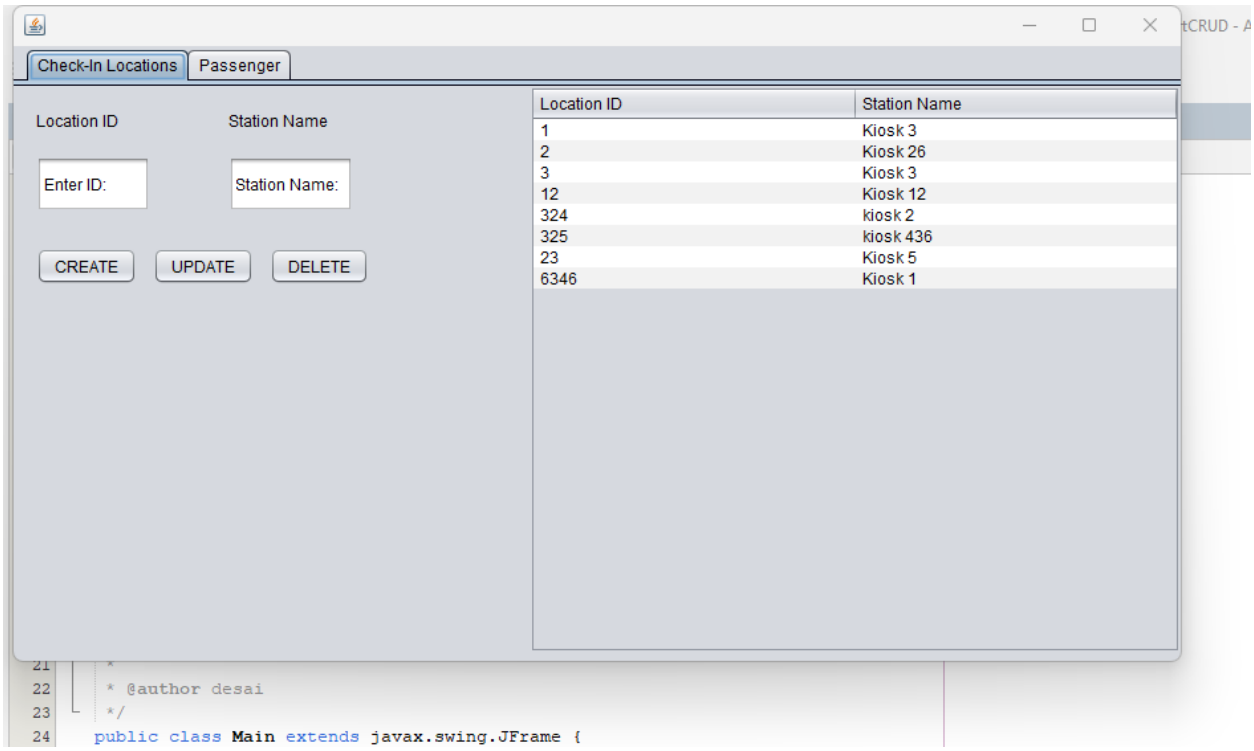


10. Show a primary key, how you print the error to the screen with screenshots. You must use ShowDialog to print the error or exception. Command line errors are not accepted. **2 points.**

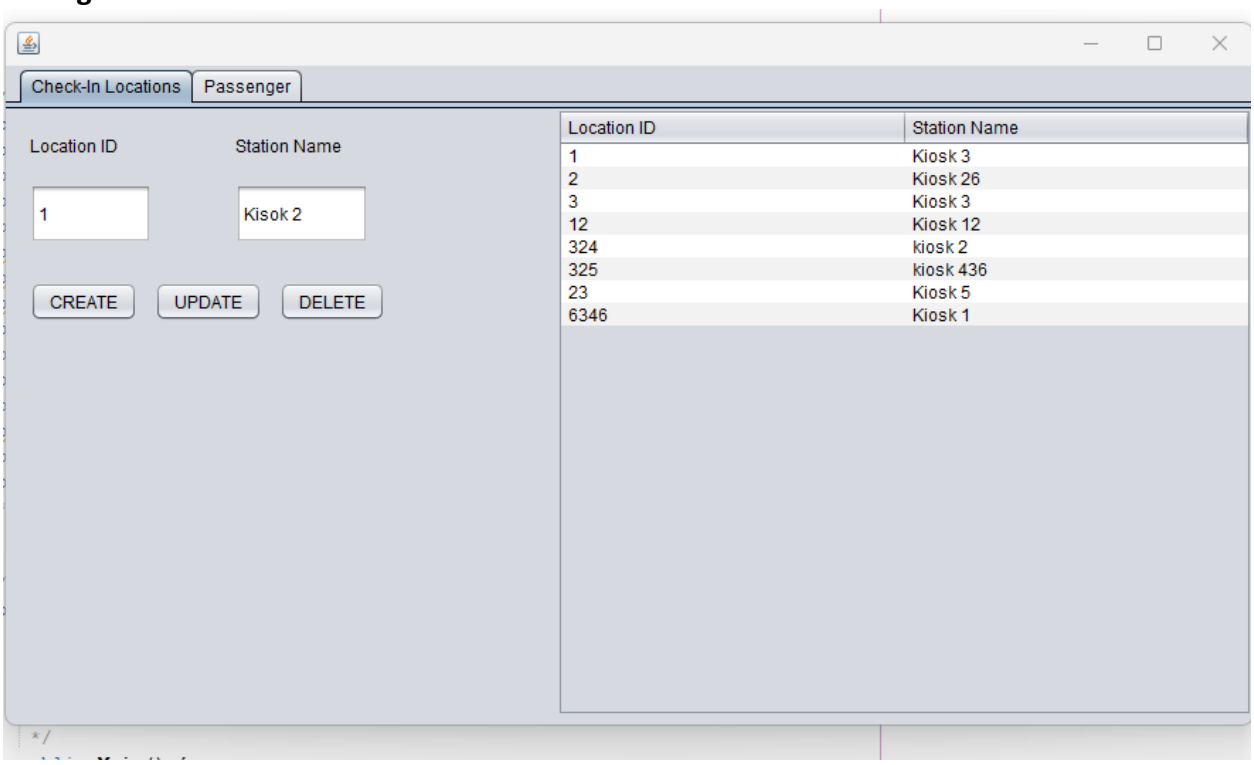
CheckIn Location PK Violation:

My example shows when I try to create a check in location that already exists.

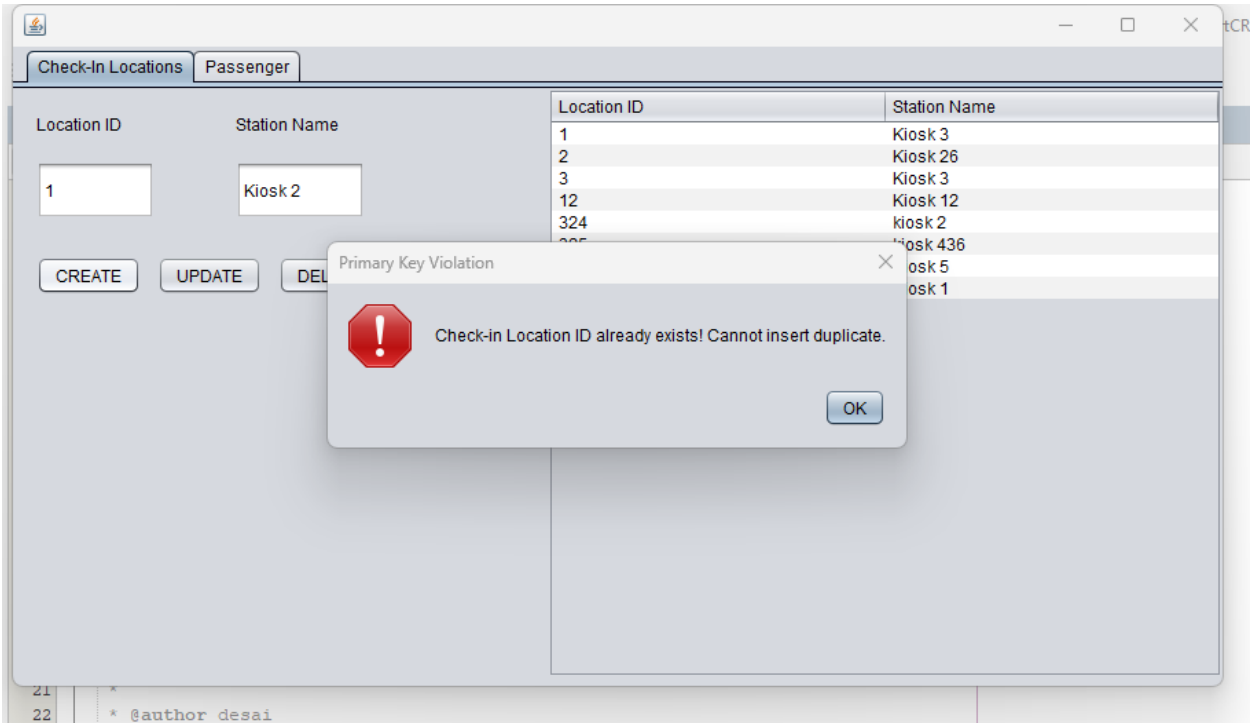
Before:


Check-In Locations
Passenger
Location ID
Station Name
Enter ID:
Station Name:
CREATE
UPDATE
DELETE
Location ID
Station Name
1
Kiosk 3
2
Kiosk 26
3
Kiosk 3
12
Kiosk 12
324
kiosk 2
325
kiosk 436
23
Kiosk 5
6346
Kiosk 1
* @author desai
*/
public class Main extends javax.swing.JFrame {

During:

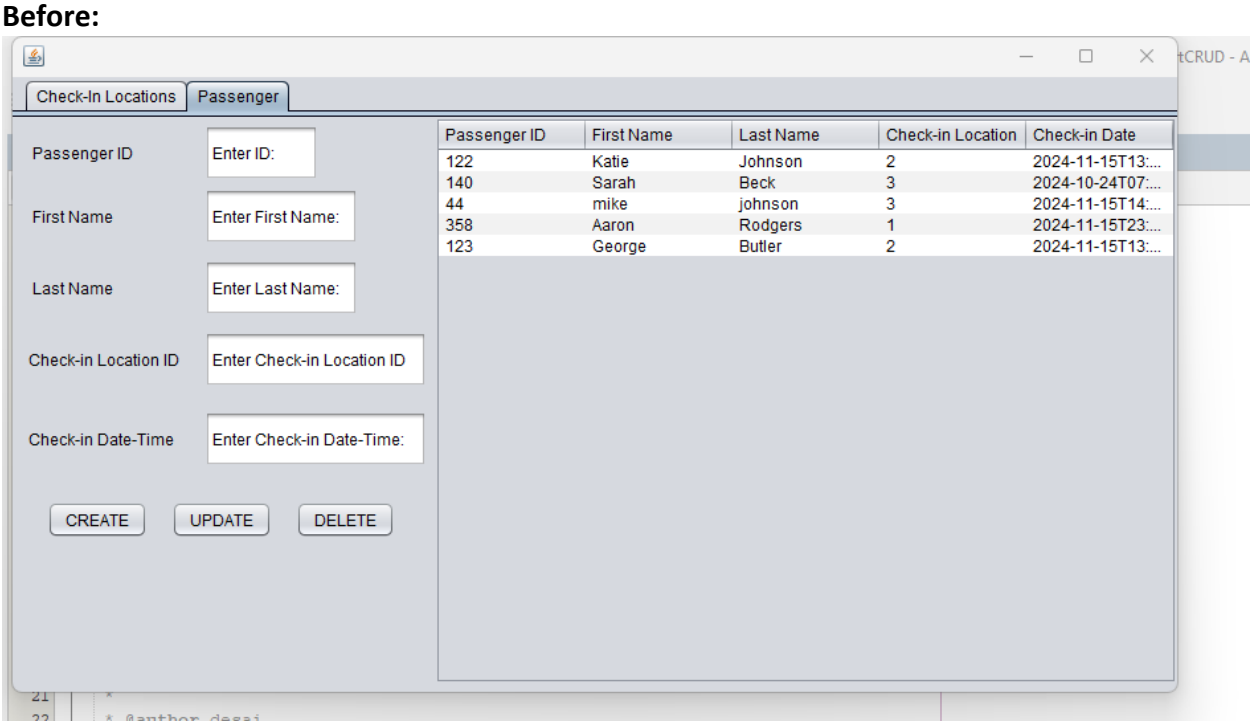

Check-In Locations
Passenger
Location ID
Station Name
1
Kisok 2
CREATE
UPDATE
DELETE
Location ID
Station Name
1
Kiosk 3
2
Kiosk 26
3
Kiosk 3
12
Kiosk 12
324
kiosk 2
325
kiosk 436
23
Kiosk 5
6346
Kiosk 1
*/

After:

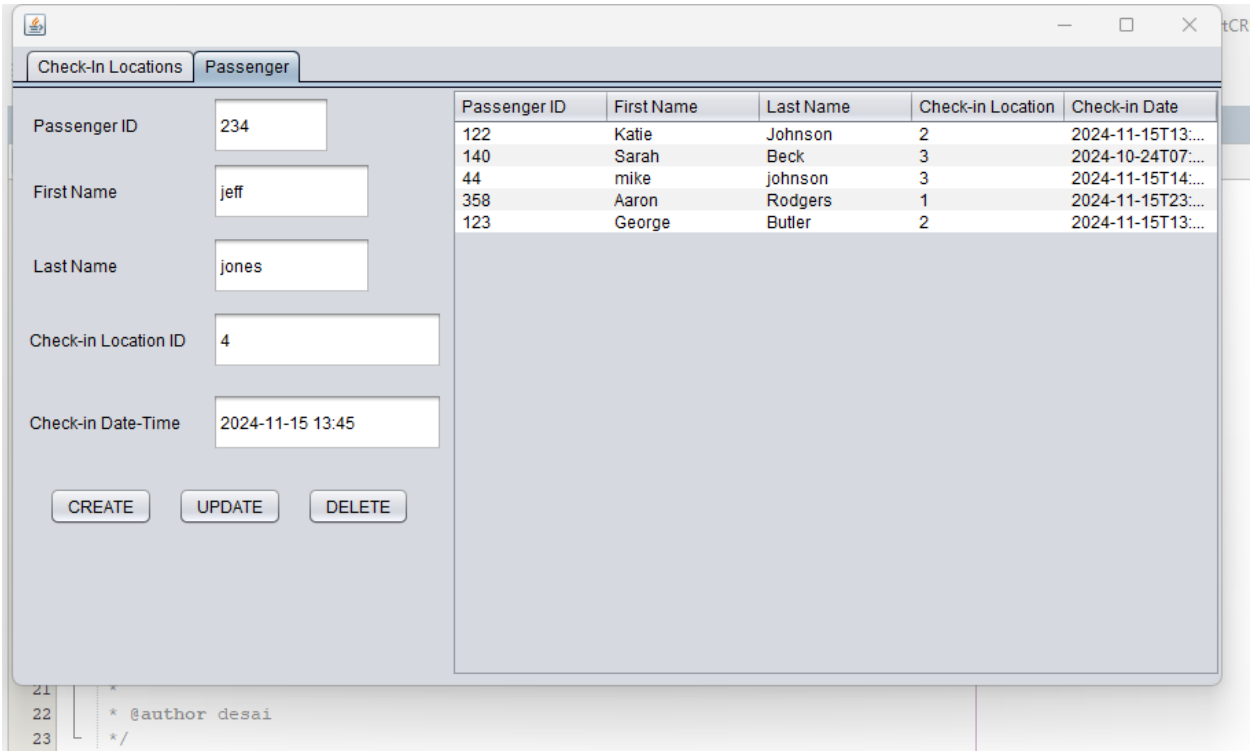


Passenger PK Violation (I used update this time, when that id doens't exist)

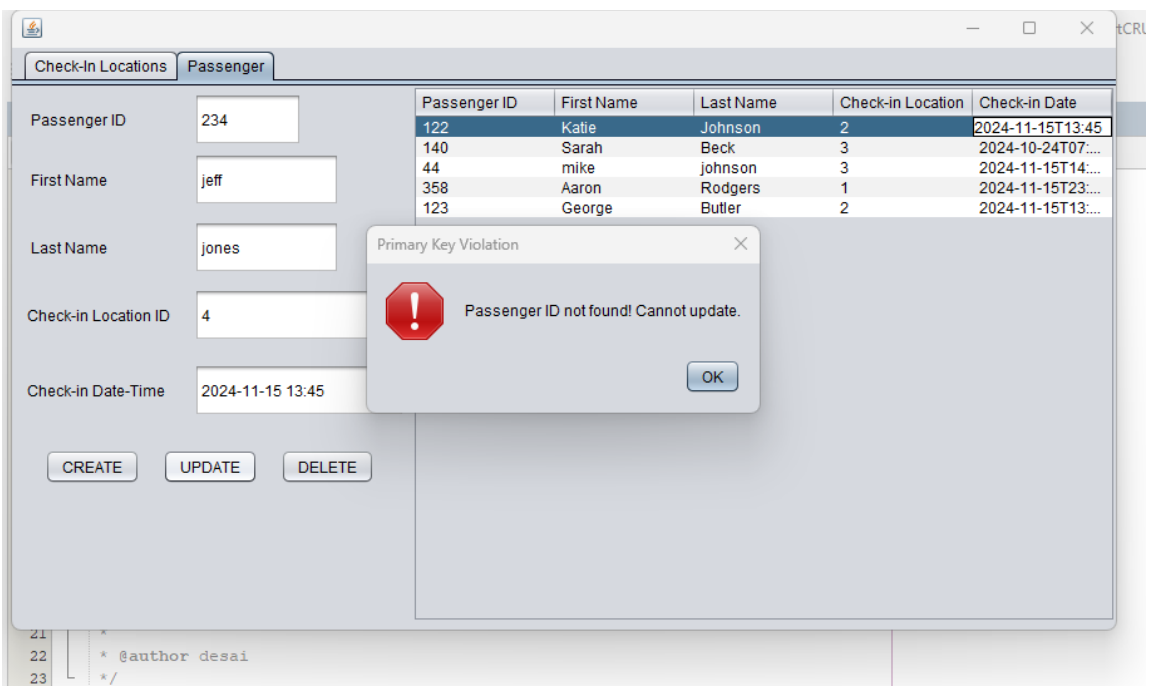
Before:



During:



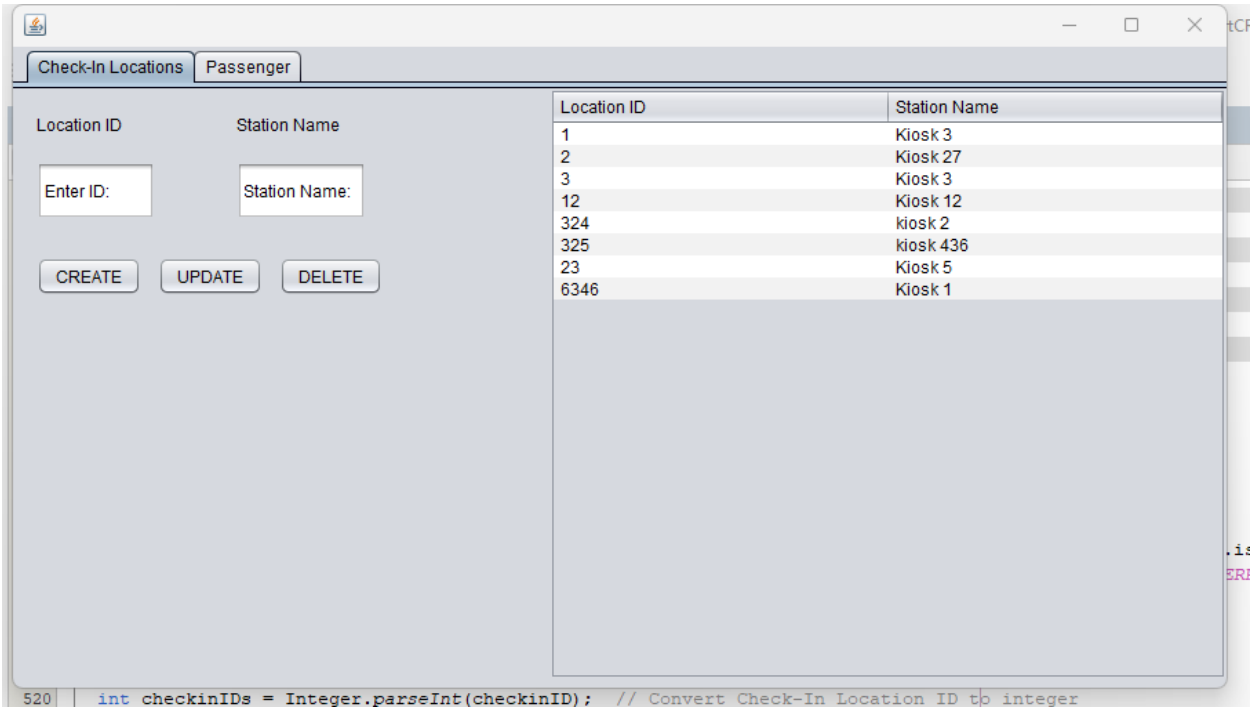
After:



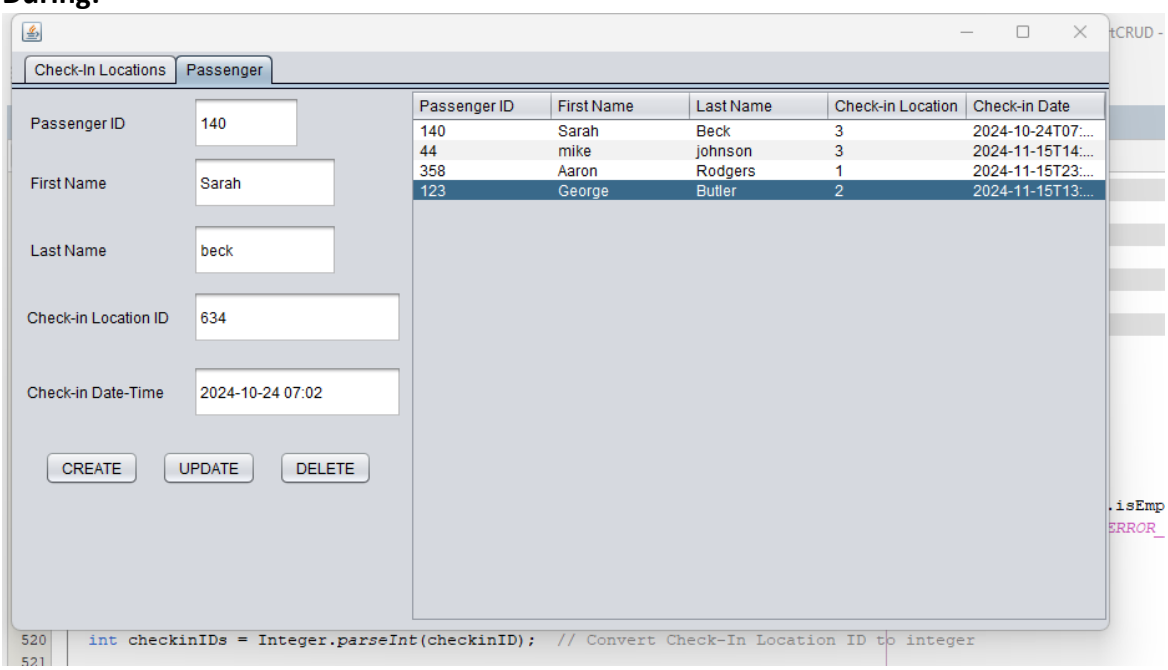
11. Show a foreign key violation, how you print the error to the screen with screenshots. You must use ShowDialog to print the error or exception. Command line errors are not accepted. **2 points.**

Passenger FK Violation:

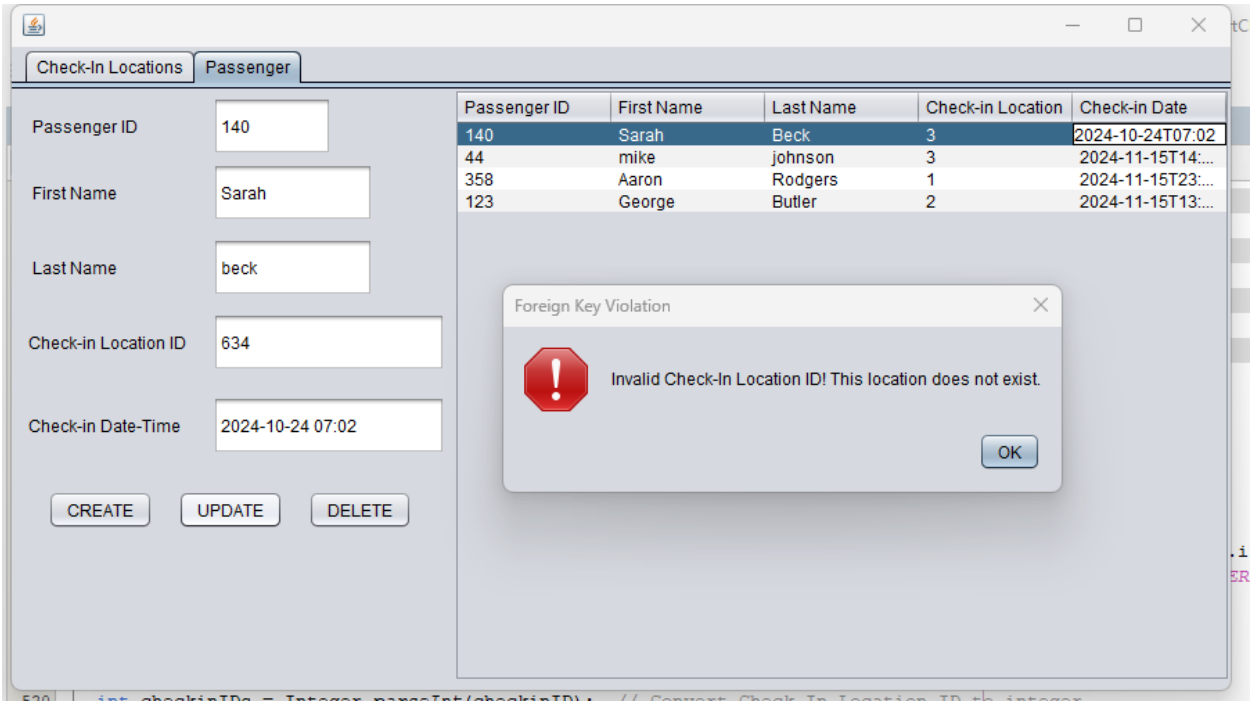
Before:



During:



After:


Check-In Locations
Passenger
Passenger ID
140
First Name
Sarah
Last Name
beck
Check-in Location ID
634
Check-in Date-Time
2024-10-24 07:02
CREATE
UPDATE
DELETE
Passenger ID
First Name
Last Name
Check-in Location
Check-in Date
140
Sarah
Beck
3
2024-10-24T07:02
44
mike
johnson
3
2024-11-15T14:...
358
Aaron
Rodgers
1
2024-11-15T23:...
123
George
Butler
2
2024-11-15T13:...
Foreign Key Violation
Invalid Check-In Location ID! This location does not exist.
OK