SSN College of Engineering
Department of Computer Science and Engineering
UCS1313 – Object Oriented Programming using Java Laboratory
II Year CSE - A Section (III Semester)
Academic Year 2019-20
****MINI PROJECT****

Airport Management System

Members of the Project are: -

1. Akilan K (185001016)
2. Arunesh Kumar (185001024)

Project Description: -

This mini project will aim to create an Airport management system with the following features: -

1. Provide current flights departing and arriving in the airport
2. Updating of current departure and arrival list according to time constraints
3. Addition, Deletion and modification of new flights in the terminal
4. Separate access for administrator to edit all details
5. Contact page with all required contacts of the flight operators as well as feature to add more by the administrator

This will be a Graphic User Interface (GUI) based application, with database connectivity to access all data and save the data.

The class diagrams and Screenshots of the project files are: -

1. **Login_Page:**

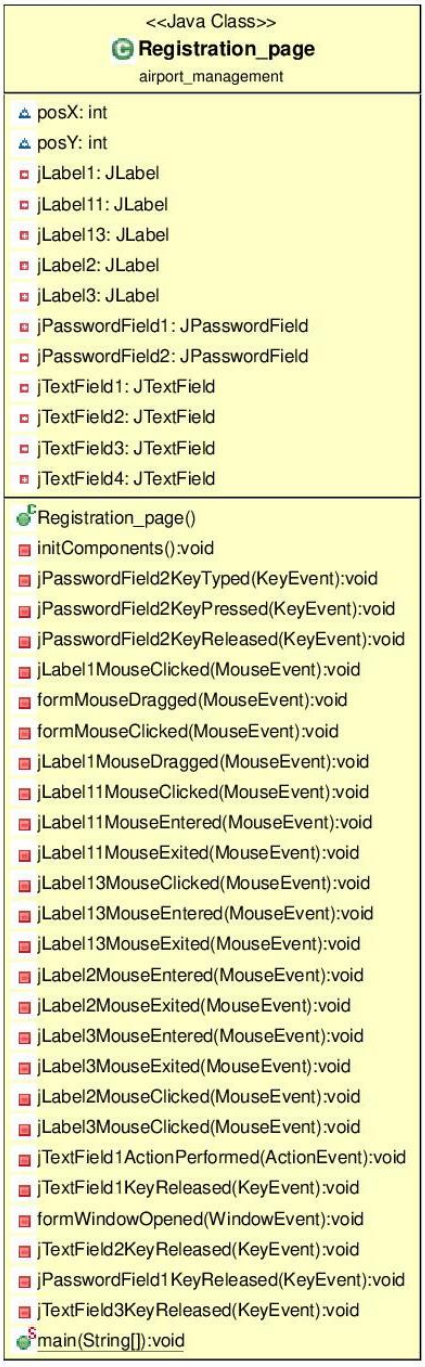
Class Diagram:

Screenshots:

- The login page asks for an ID and the password which is checked with the database and actions are taken.
- To create a new user the “NEW USER” button is used.

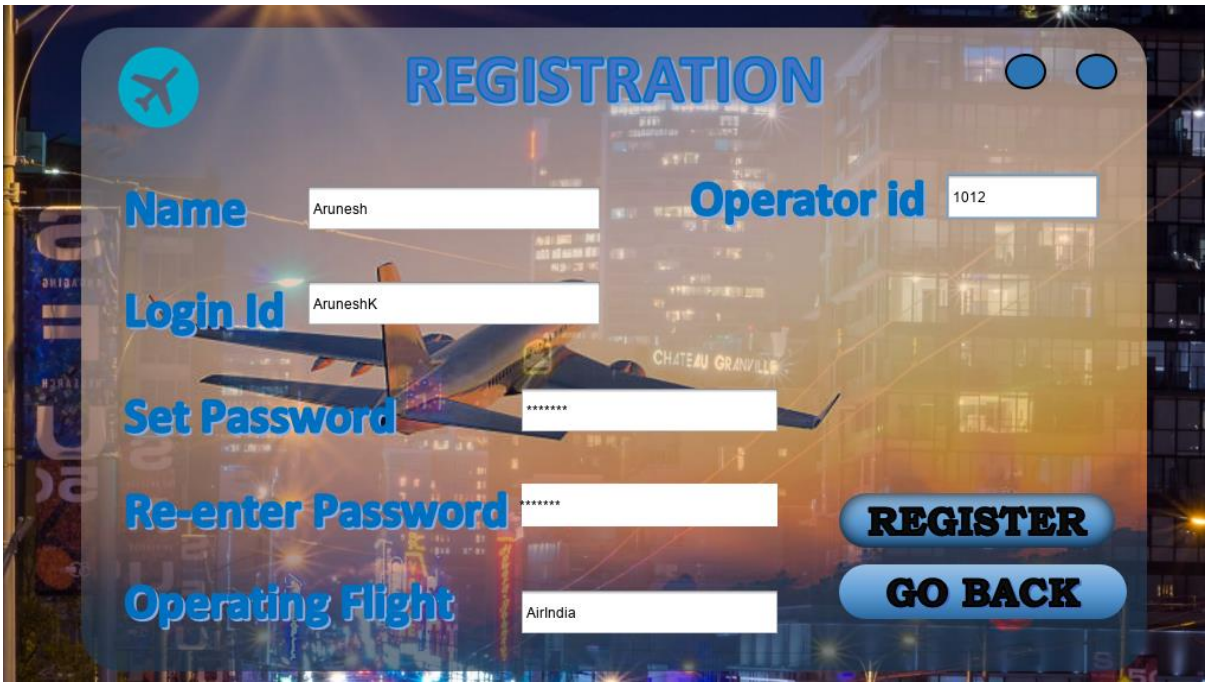
2. Registration_Page:

Class Diagram:

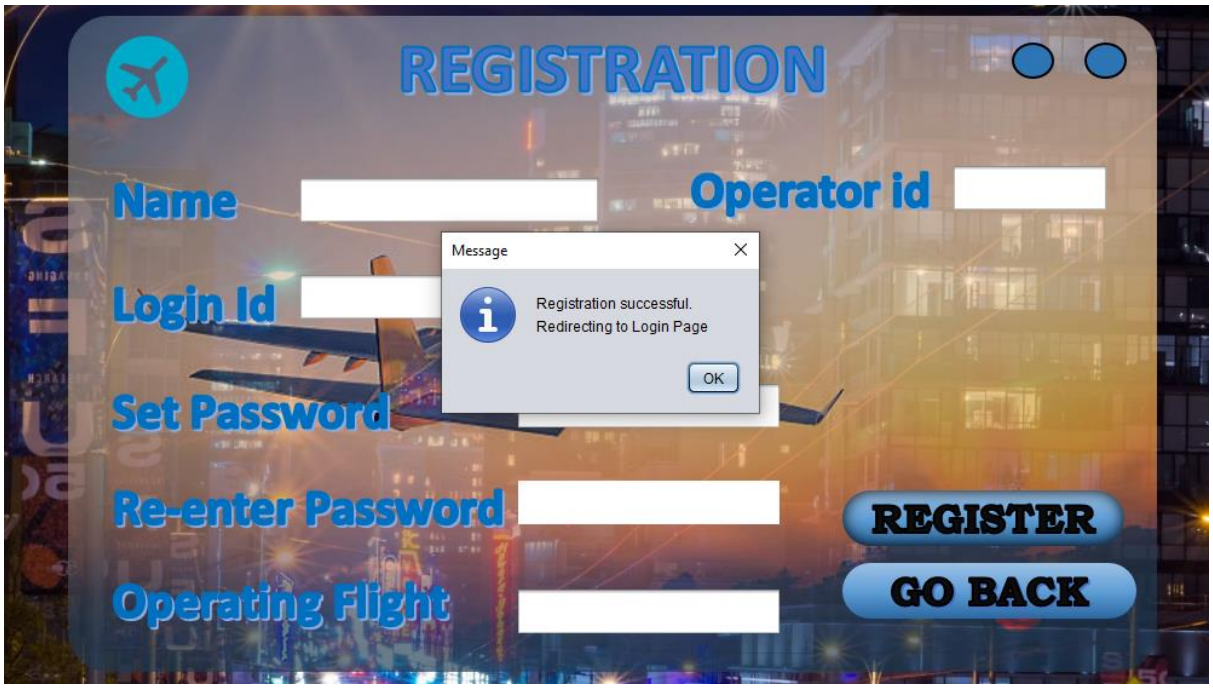


Screenshots:

- When no information is entered, the field remains red and doesn't accept it.



- The new user's information is entered.

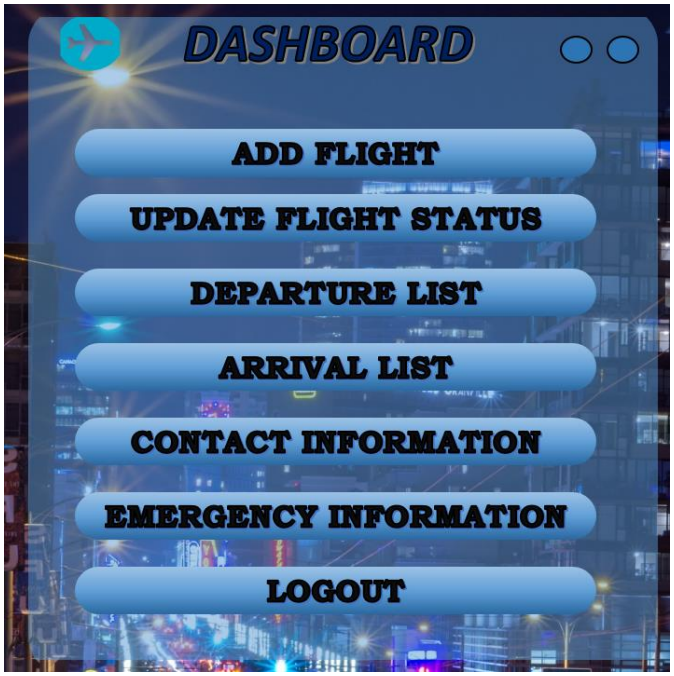


- Once the information is entered, it is stored in the database and the registration is done successfully.

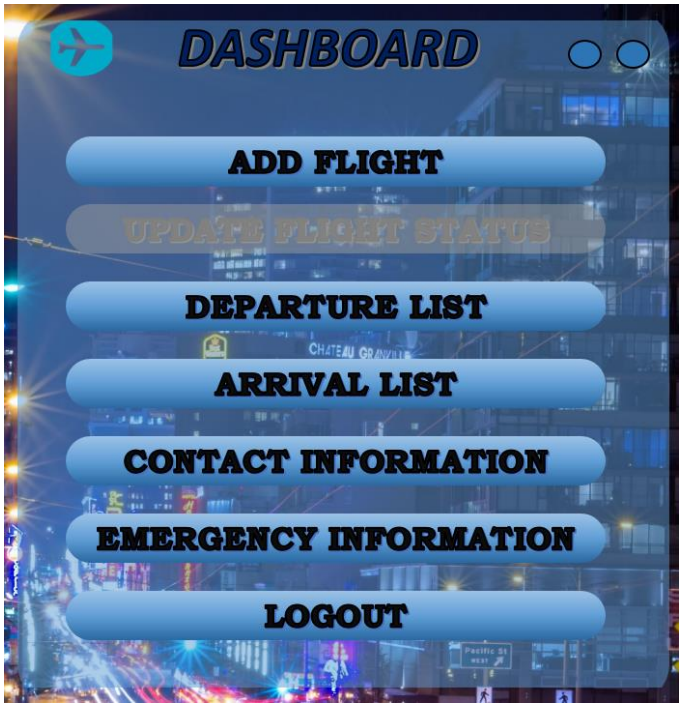
3. Dashboard:

Class Diagram:

Screenshots:



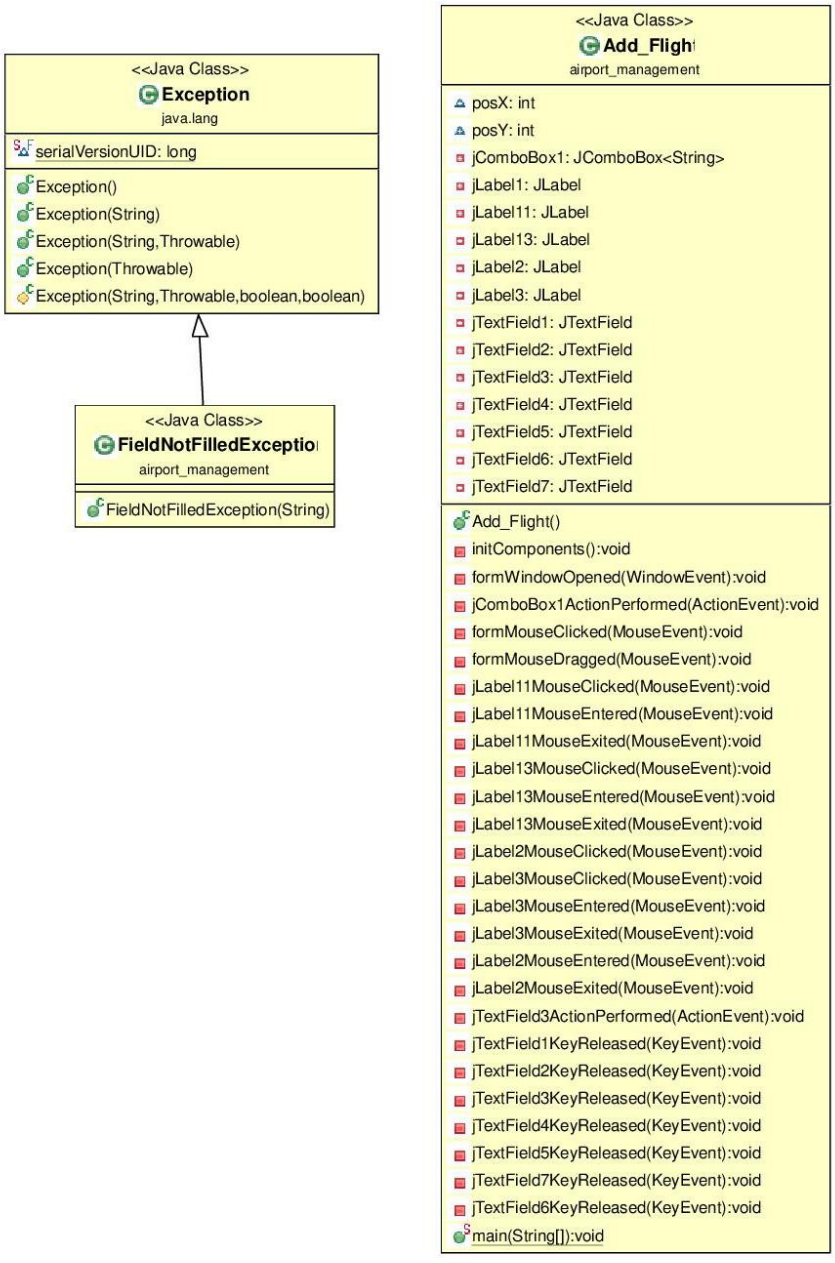
- The dashboard offers the above functionality.
- For the admins the “Update flight status” is enabled.



- For a normal user the “Update flight status” is greyed out.

4. Add_Flight:

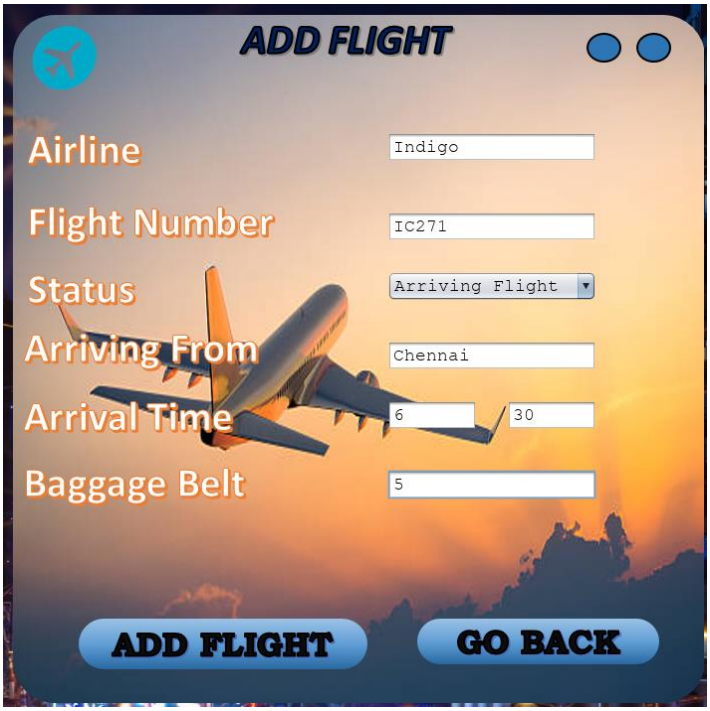
Class Diagram:



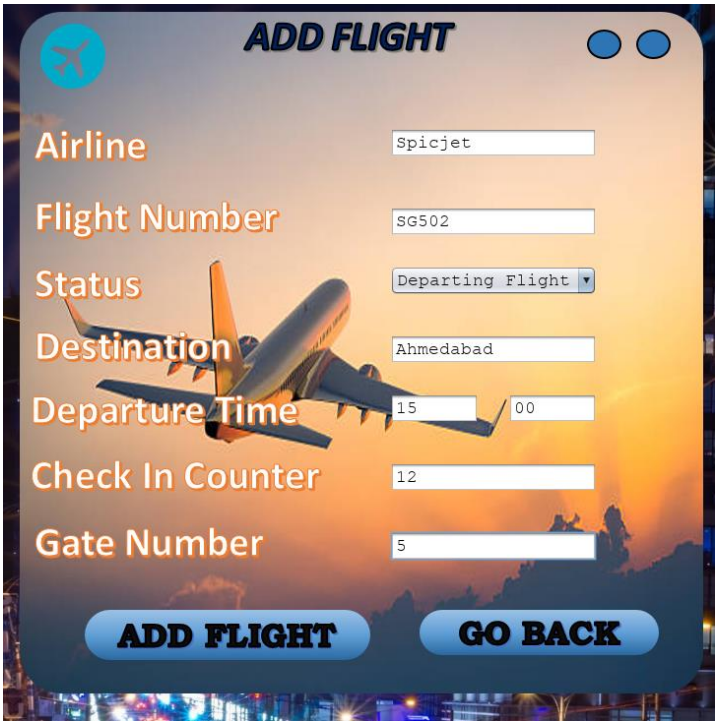
- The **FieldNotFilledException extends Exception** class and is used to show that a field is not filled.

Screenshots:

- When the flight details are not filled properly, a FieldNotFilledException is thrown which makes the text box red indicating that it is not properly entered.



- The flight's information is entered.
- Based on the status of the flight, different information is asked.
- Arriving Flight is selected using the drop-down menu.

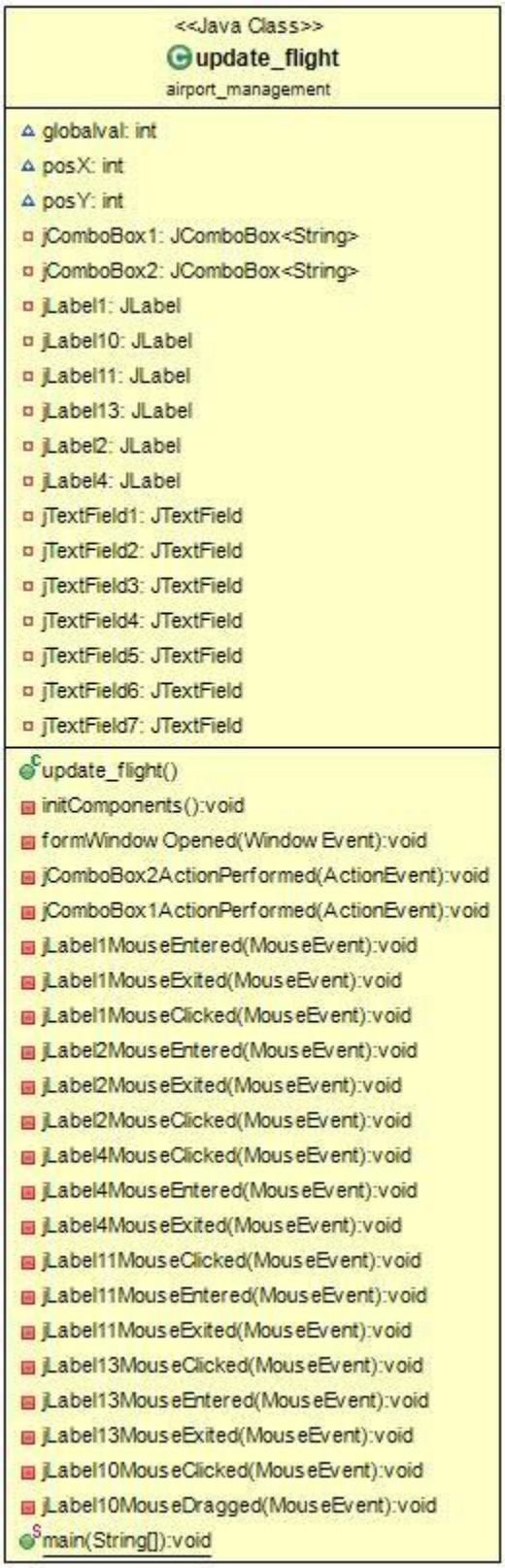


- Departing flight is selected using the drop-down menu.

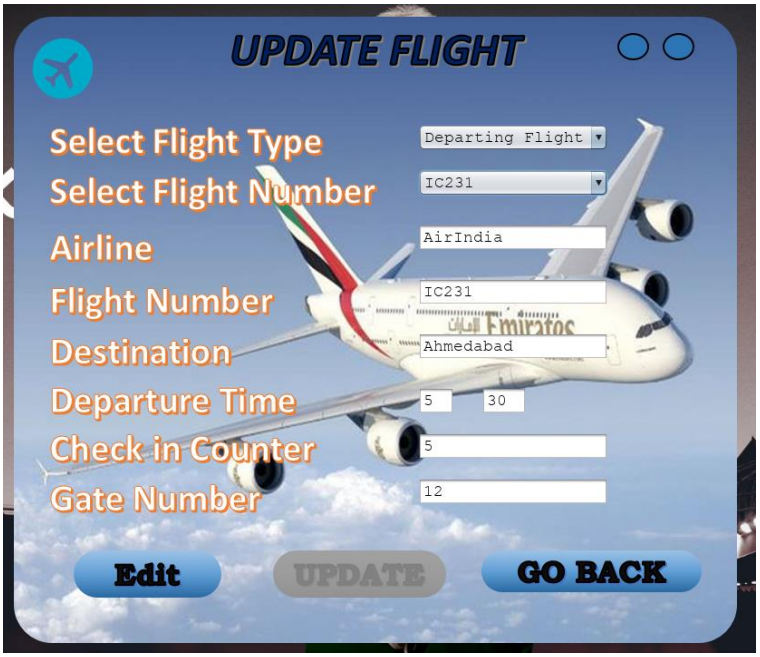
- After all the information is added, the “Add Flight” button is pushed which adds the flight to the database.

5. update_flight:

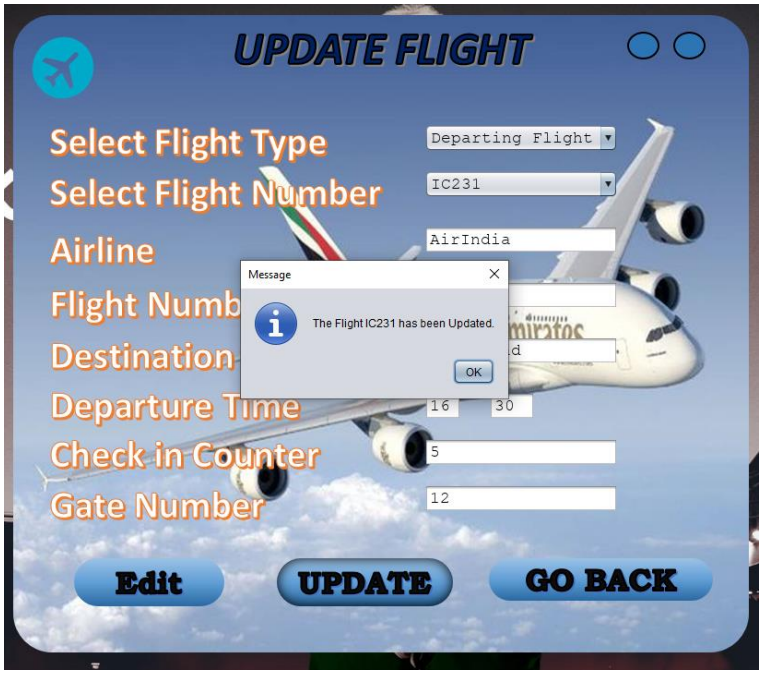
Class Diagram:



Screenshots:



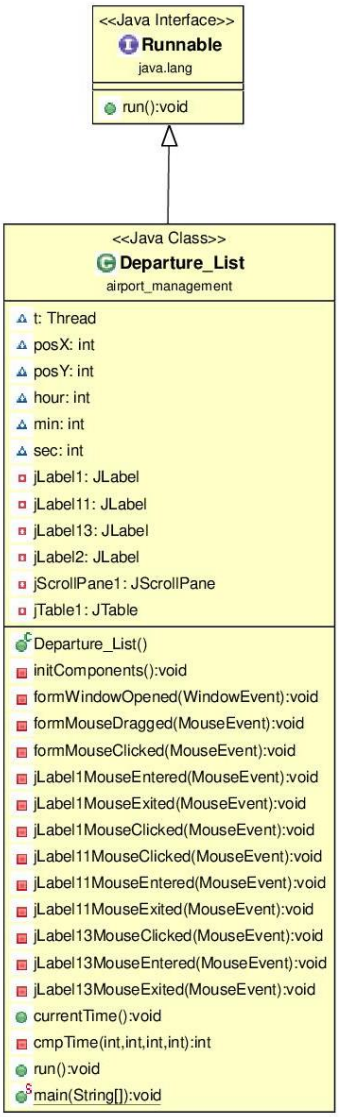
- The update flight function is only available for the admins.
- Here a flight is selected either arriving or departing using a drop-down menu.
- The necessary information is changed about the flight.



- After the information is entered, it is then updated in the database.

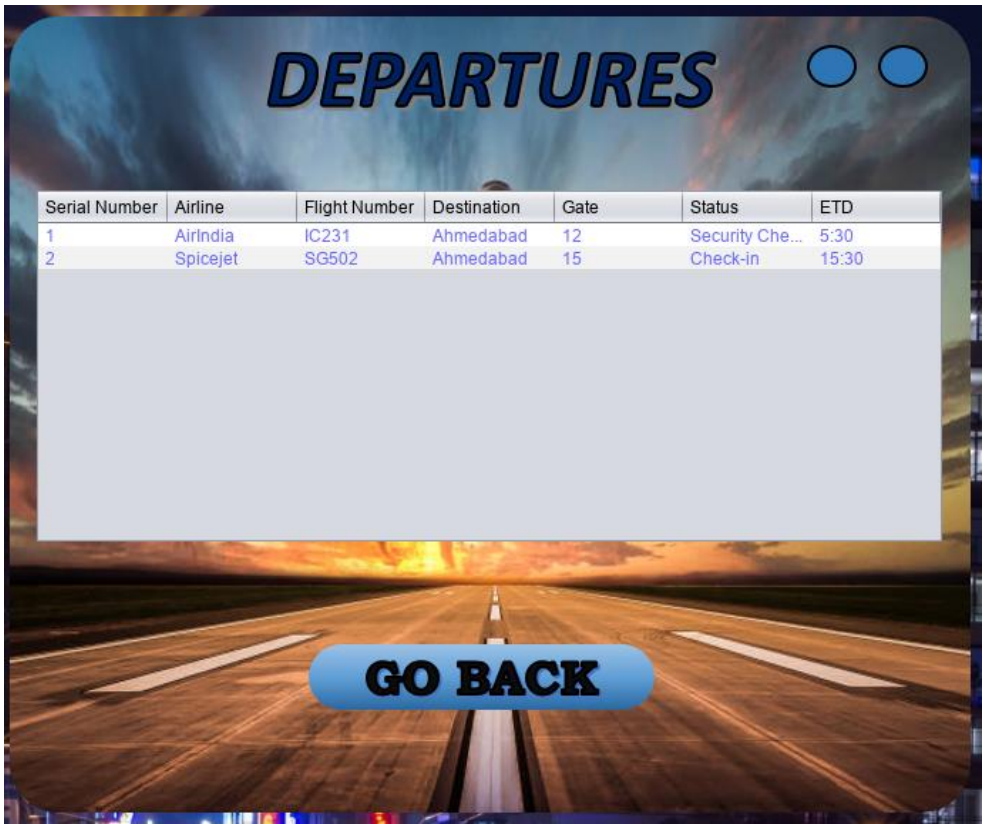
6. Departure_List:

Class Diagram:



- The **Departure_List implements Runnab**le interface to use the run function and to implement **parallel processing.**

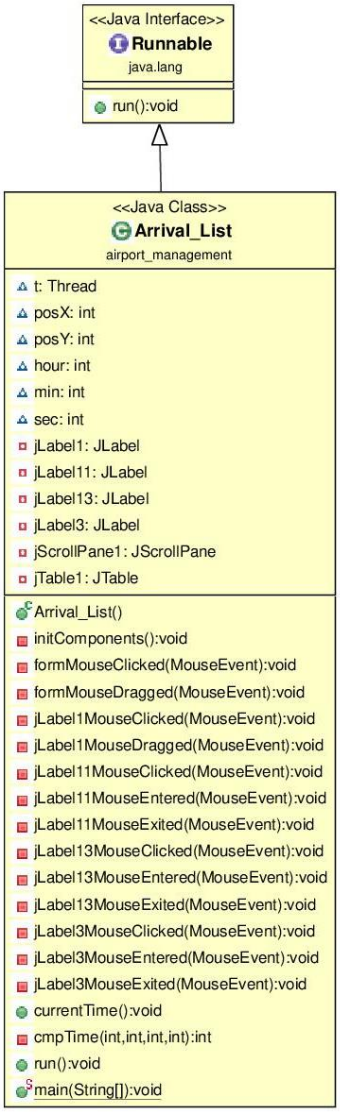
Screenshots:



- Used to display the information of the flights in the departing list.
- When the flights are departed, the records are automatically deleted based on time.

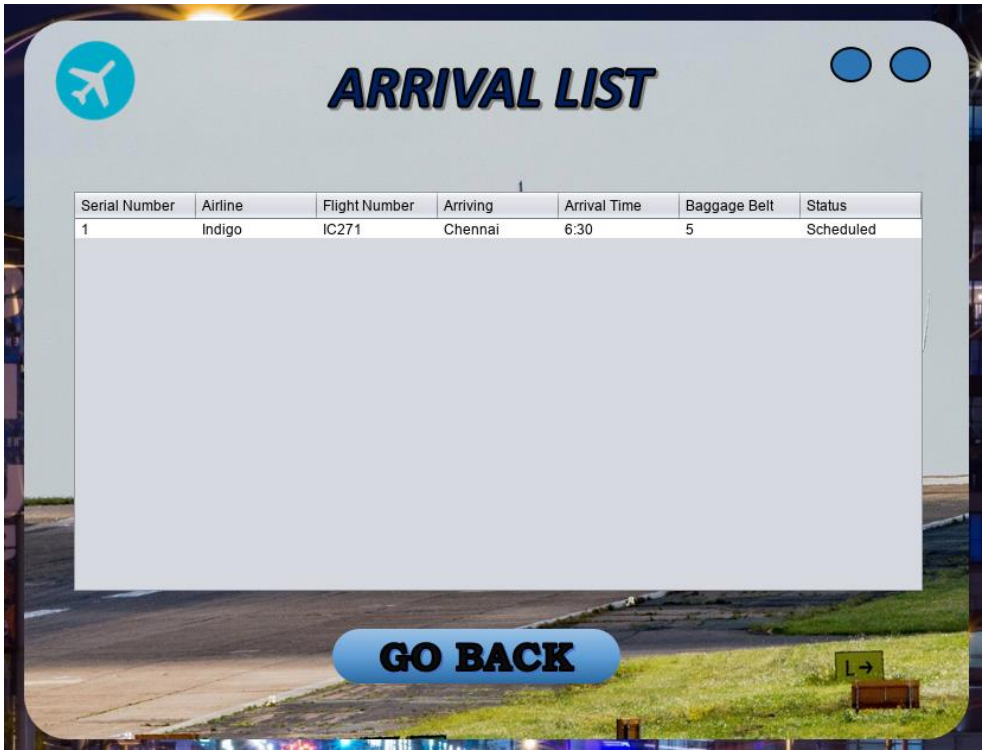
7. Arrival_List:

Class Diagram:



•The **Arrival_List implements Runnab**le interface to use the run function and to implement **parallel processing.**

Screenshots:

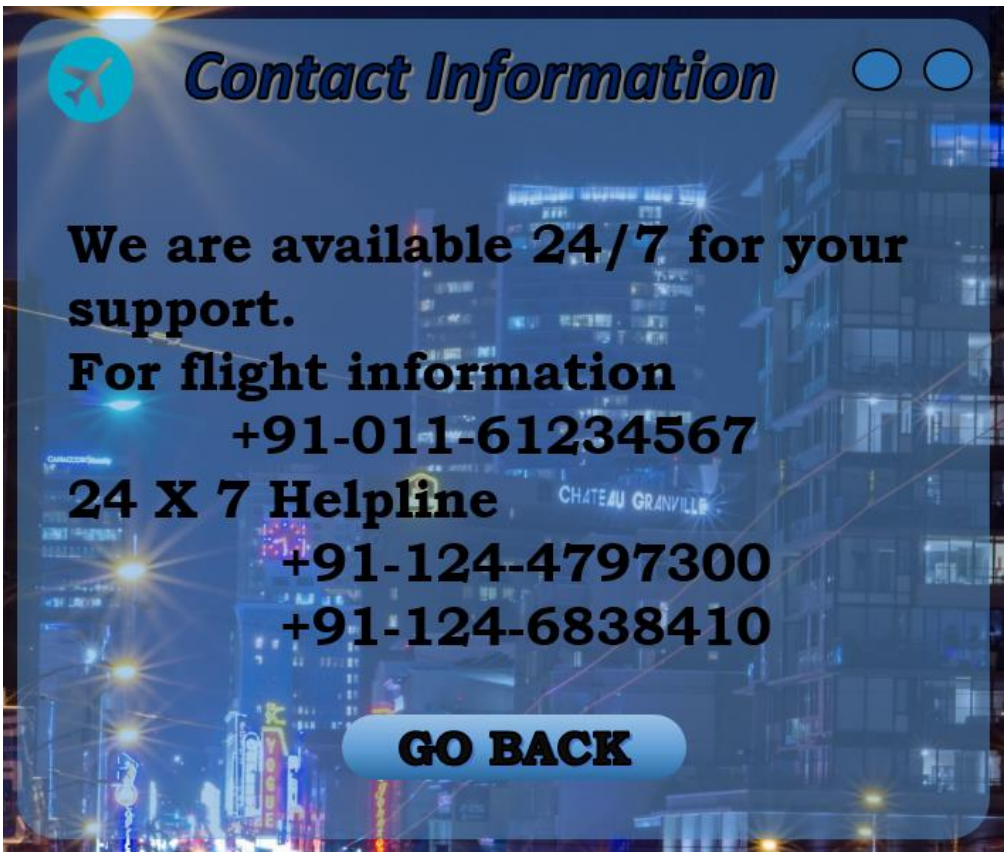


- The flights in the Arrival list are displayed.

8. **contact_page:**

Class Diagram:

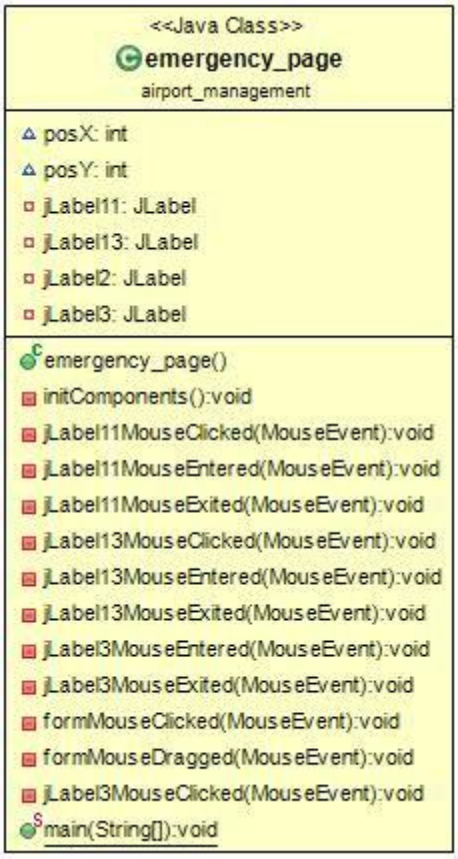
Screenshots:



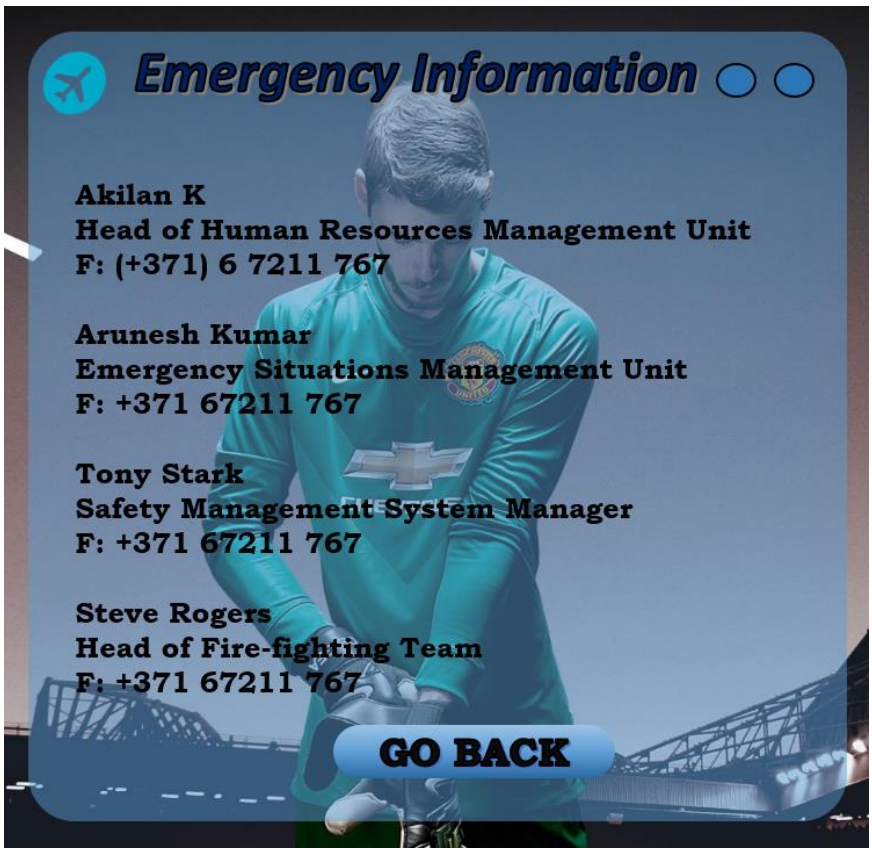
- Contact information is displayed.

9. **emergency_page:**

Class Diagram:



Screenshots:



- Emergency information is displayed.

Learning Outcome:

- Netbeans stuff
- Swing stuff
- Sql stuff
- We leant about exception handling.
- We learnt about the runnable interface and about parallel processing.